Intel's Unnati Program

PS 04: Introduction to GenAI and Simple LLM Inference on CPU and finetuning of LLM Model to create a custom chatbot.

Name: Agamjot Kaur Choudhary

Register No: RA2311026010681

Course: CSE(AIML)

Year: 2

Semester: 3

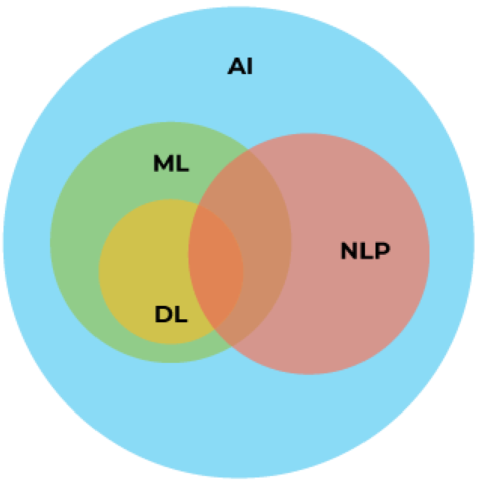
Table Of Contents

Introduction

My journey into Artificial Intelligence (AI) began with practical projects in computer vision and natural language processing (NLP), where I developed an image classifier and worked on a fake news detection model. These projects helped me understand data manipulation, supervised learning techniques, and their applications in specialised domains. During the Unnati Program, I delved deeper into these topics and programmed several different types of Bots. I have worked on a total of 3 Bots: FAQ Answering Bot, Storytelling Bot, ChatBot

Visual Representation of AI Technologies:

1. **Artificial Intelligence (AI)**: At the core, AI represents the broader field of creating intelligent systems capable of performing tasks that typically require human intelligence. It encompasses various subfields such as ML, Deep Learning, and NLP.
2. **Machine Learning (ML)**: ML involves algorithms that learn patterns and make decisions from data, without explicit programming. It includes supervised learning, unsupervised learning, and reinforcement learning.
3. **Deep Learning (DL)**: A subset of ML, deep learning uses neural networks with many layers to learn representations of data. It excels in tasks like image and speech recognition.
4. **Natural Language Processing (NLP)**: NLP focuses on enabling machines to understand and generate human language. It includes tasks like text classification, language translation, and sentiment analysis.



LLM Models and Transformer Architecture

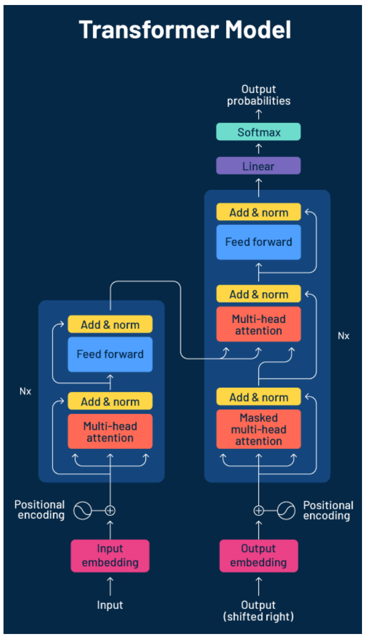
Language Model (LLM) models are pivotal in advancing natural language understanding and generation tasks. These models utilise Transformer architectures, which employ self-attention mechanisms to capture complex relationships within data. Transformers facilitate effective information processing across sequences, enabling models like LLaMA-2 to comprehend context and generate human-like responses.

Transformer Architecture: Flow of Information

The Transformer architecture revolutionises Natural Language Processing (NLP) by leveraging self-attention mechanisms to capture contextual dependencies efficiently. Here's how it works:

1. Input Embeddings:
   - Input sequences are embedded into vectors, combining token embeddings (word meanings) and positional encodings (sequence order).
2. Encoder Layers:
   - Each encoder layer processes the input embeddings through two main steps:

        - Self-Attention Mechanism: Computes attention scores to weigh the importance of each token in relation to others in the sequence.
        - Feed-Forward Neural Network: Applies transformations to each token independently, enhancing the model's ability to capture complex patterns.



3. Decoder Layers:
    - Decoder layers also operate in two stages:
        - Self-Attention: Helps the model focus on different parts of the input sequence during decoding.
        - Encoder-Decoder Attention: Allows the decoder to use the encoded representations to generate the output sequence.
4. Output Generation:
    - The final output embeddings are decoded into the desired format, such as translated text or generated responses.

Fine-Tuning Techniques: Optimising Model Performance

Fine-tuning plays a crucial role in adapting pre-trained LLM models like LLaMA-2 to specific tasks or domains. This process involves adjusting model parameters and training on domain-specific datasets to improve performance and task-specific outcomes. Several fine-tuning techniques have emerged to enhance model capabilities:

- **QLoRA (Quantized Long Range Arena)**: QLoRA optimises model efficiency by employing quantization techniques, reducing computational overhead while maintaining performance integrity.
- **LoRA (Long Range Arena)**: LoRA addresses challenges associated with long-range dependencies in data processing. It enhances model capacity to capture distant relationships within sequences, improving overall predictive accuracy.
- **RaG (Retrieval-augmented Generation)**: RaG integrates retrieval-based strategies into model training, enabling enhanced context understanding and response generation. By incorporating external knowledge sources during generation, RaG enhances the relevance and accuracy of generated outputs.

Intel Tools Explored

- Intel Developer Cloud Console: Gen AI Essentials
    - Simple LLM Inference
    - Build Chatbot on SPR
    - Fine-tuning on Single Node Xeon SPR.
    - Text to Image with Stable Diffusion
    - Simple RaG
    - Fine-tuning Google's GEMMA Model on Intel Max GPU Series

Project 1: FAQ Answering Bot

Overview:

This Python script demonstrates the creation of an FAQ answering bot using the GPT-2 model, fine-tuned on a custom retail FAQ dataset. It showcases model training and interaction using the Hugging Face transformers and datasets libraries, providing a practical example of how to build and deploy an AI-powered FAQ bot. Initially, there were random responses, however, with more training, the response quality improved.

Applications:

- Customer Support: Enhances customer service by providing instant and accurate responses to frequently asked questions.
- Educational Tool: Facilitates learning about NLP, transformer models, and fine-tuning techniques through practical implementation.
- Prototyping: Serves as a prototype for developing AI applications that require automatic question-answering capabilities.

Components Utilised:

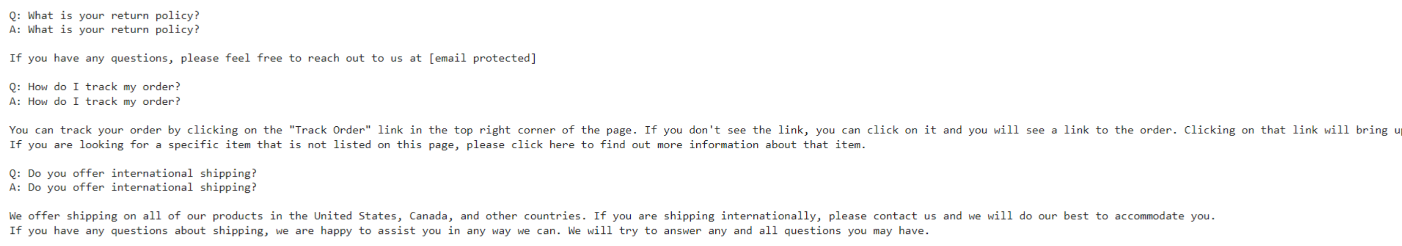
- Libraries: datasets and transformers from Hugging Face for dataset handling and model training.
- GPT-2 Model: Utilises the GPT-2 language model architecture for generating responses to FAQs.
- Tokenizer: AutoTokenizer from Hugging Face's transformers library for tokenizing input texts.
- Trainer: Trainer class from transformers for configuring and executing model training with specified training arguments.
- Dataset: A retail FAQ dataset from Hugging Face's dataset repository for training the model to understand and answer common retail-related questions.
- Fine-Tuning Method: Supervised learning is used for fine-tuning the GPT-2 model. This approach involves training the model on pairs of questions and answers to improve its ability to generate relevant responses.

Justification:

The script employs supervised learning via fine-tuning on the retail FAQ dataset to adapt the pre-trained GPT-2 model for question-answering tasks. This approach enhances model performance by leveraging transfer learning, enabling the FAQ bot to produce accurate and contextually relevant responses based on user questions. Supervised learning is chosen for its effectiveness in training the model to understand specific patterns in question-answer pairs.

GitHub Link

Sample Inputs and Outputs:



```
Q: What is your return policy?
A: What is your return policy?

If you have any questions, please feel free to reach out to us at [email protected]

Q: How do I track my order?
A: How do I track my order?

You can track your order by clicking on the "Track Order" link in the top right corner of the page. If you don't see the link, you can click on it and you will see a link to the order. Clicking on that link will bring u
If you are looking for a specific item that is not listed on this page, please click here to find out more information about that item.

Q: Do you offer international shipping?
A: Do you offer international shipping?

We offer shipping on all of our products in the United States, Canada, and other countries. If you are shipping internationally, please contact us and we will do our best to accommodate you.
If you have any questions about shipping, we are happy to assist you in any way we can. We will try to answer any and all questions you may have.
```

Project 2: Storytelling Bot

Overview:

This project implements a storytelling bot using the GPT-2 (Generative Pre-trained Transformer 2) model from the transformers library. The bot generates creative stories based on user-provided prompts, designed to be engaging and fun for kids.

Applications:

- Creative Writing Assistance: Provides inspiration and generates stories for young writers and content creators.
- Educational Tool: Enhances learning through interactive storytelling experiences.
- Entertainment: Offers an enjoyable and imaginative activity for children.

Components Utilised:

- GPT2 Tokenizer: Tokenizes input text for model processing.
- GPT2LMHeadModel: Implements the GPT-2 model for language modeling and story generation.
- Fine-Tuning Technique: The model is fine-tuned using supervised learning, specifically transfer learning. This approach adapts a pre-trained GPT-2 model on a dataset curated for children's stories and imaginative prompts, enabling the bot to generate coherent and age-appropriate content that resonates with young audiences.

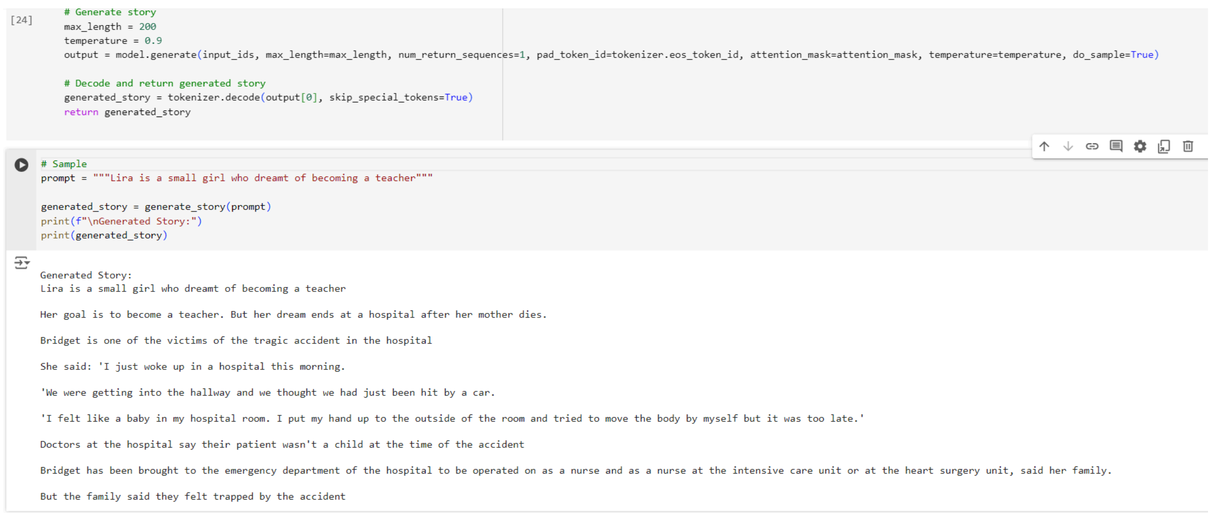
Justification:

Supervised learning through transfer learning ensures the model learns from labeled examples (children's stories), enhancing its ability to generate stories that align with the stylistic and thematic preferences typical in children's literature. This method not only improves customization but also ensures the generated stories are engaging and suitable for educational purposes.

GitHub Link

Sample Inputs and Outputs:

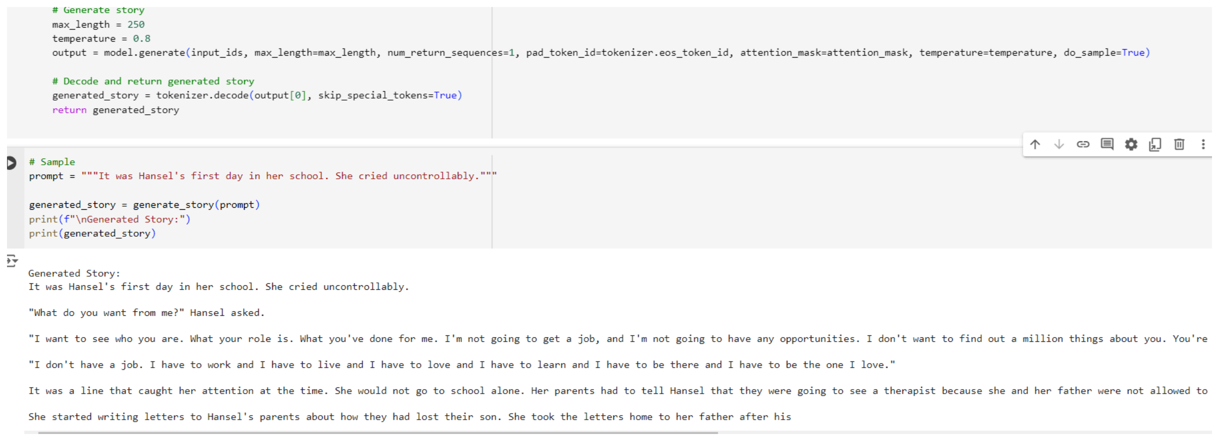
Through the following inputs and outputs we can see the difference in the story created using two different temperature values, 0.8 and 0.9. When the temperature parameter increases, the creativity of the model also increases. However, we notice that the output created at 0.9 is much more logical than that at 0.8. Since it is a storytelling bot, it requires creativity, thus we are using such high temperature parameters.



```
[24]    # Generate story
        max_length = 200
        temperature = 0.9
        output = model.generate(input_ids, max_length=max_length, num_return_sequences=1, pad_token_id=tokenizer.eos_token_id, attention_mask=attention_mask, temperature=temperature, do_sample=True)

        # Decode and return generated story
        generated_story = tokenizer.decode(output[0], skip_special_tokens=True)
        return generated_story
```

```
# Sample
prompt = """Lira is a small girl who dreamt of becoming a teacher"""

generated_story = generate_story(prompt)
print(f"\nGenerated Story:")
print(generated_story)
```

```
Generated Story:
Lira is a small girl who dreamt of becoming a teacher

Her goal is to become a teacher. But her dream ends at a hospital after her mother dies.

Bridget is one of the victims of the tragic accident in the hospital

She said: 'I just woke up in a hospital this morning.

'We were getting into the hallway and we thought we had just been hit by a car.

'I felt like a baby in my hospital room. I put my hand up to the outside of the room and tried to move the body by myself but it was too late.'

Doctors at the hospital say their patient wasn't a child at the time of the accident

Bridget has been brought to the emergency department of the hospital to be operated on as a nurse and as a nurse at the intensive care unit or at the heart surgery unit, said her family.

But the family said they felt trapped by the accident
```



```
    # Generate story
    max_length = 250
    temperature = 0.8
    output = model.generate(input_ids, max_length=max_length, num_return_sequences=1, pad_token_id=tokenizer.eos_token_id, attention_mask=attention_mask, temperature=temperature, do_sample=True)

    # Decode and return generated story
    generated_story = tokenizer.decode(output[0], skip_special_tokens=True)
    return generated_story
```

```
# Sample
prompt = """It was Hansel's first day in her school. She cried uncontrollably."""

generated_story = generate_story(prompt)
print(f"\nGenerated Story:")
print(generated_story)
```

```
Generated Story:
It was Hansel's first day in her school. She cried uncontrollably.

"What do you want from me?" Hansel asked.

"I want to see who you are. What your role is. What you've done for me. I'm not going to get a job, and I'm not going to have any opportunities. I don't want to find out a million things about you. You're

"I don't have a job. I have to work and I have to live and I have to love and I have to learn and I have to be there and I have to be the one I love."

It was a line that caught her attention at the time. She would not go to school alone. Her parents had to tell Hansel that they were going to see a therapist because she and her father were not allowed to

She started writing letters to Hansel's parents about how they had lost their son. She took the letters home to her father after his
```

Project 3: ChatBot

Overview:

This Python script demonstrates how to train a GPT-2 (Generative Pre-trained Transformer 2) model on a custom dataset and deploy it as an interactive AI chatbot for text generation. It utilises the transformers library for model training and datasets library for dataset handling, offering insights into tokenization, model training configuration, and interactive text generation.

Applications:

- Interactive Chatbot: Enables users to interact with an AI-powered chatbot for generating responses based on user inputs. This can be particularly enjoyable for kids, providing fun and engaging stories.
- Educational Tool: Facilitates learning about natural language processing and transformer-based models through practical implementation.
- Prototyping: Serves as a prototype for developing AI applications that require natural language understanding and generation capabilities.

Components Utilised:

- Libraries: datasets, transformers, accelerate, and matplotlib for dataset management, model training, performance optimization, and visualisation.
- GPT-2 Model: Utilises the GPT-2 language model architecture for natural language generation tasks.
- Tokenizer: GPT2 Tokenizer from Hugging Face's transformers library for tokenizing input texts.
- Trainer: Trainer class from transformers for configuring and executing model training with specified training arguments.
- Text Generation Pipeline: pipeline function from transformers for setting up an interactive text generation pipeline with the trained GPT-2 model.
- Dataset: The Alpaca dataset from the AlpacaDataCleaned repository is used for fine-tuning the GPT-2 model, providing a diverse set of instructions and outputs to improve the model's conversational abilities.
- Fine-Tuning Method: Low-Rank Adaptation (LoRA) is used for fine-tuning the GPT-2 model. LoRA efficiently adapts the pre-trained model by adjusting only a subset of the model parameters, leading to faster training times and reduced computational resources.

Justification:

The script employs supervised learning via fine-tuning on the Alpaca dataset to adapt the pre-trained GPT-2 model for specific text generation tasks. This approach enhances model performance by leveraging transfer learning, enabling the chatbot to produce coherent and contextually relevant responses based on user inputs. The LoRA fine-tuning technique is chosen for its efficiency in training, requiring fewer resources while maintaining high performance.

GitHub Link

Dataset Details:

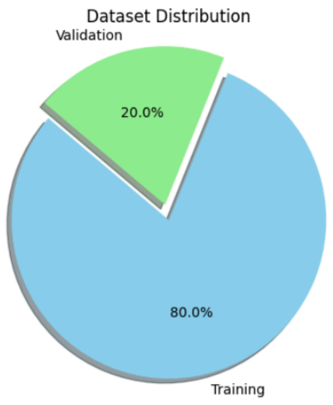
Total tokenized examples: 51,760

Training examples: 41,408

Validation examples: 10,352

Pie Chart:

The pie chart above illustrates the distribution of the dataset into training and validation sets.



Sample Inputs and Outputs:

You: Tell me a story about a magical forest.
Chatbot: Once upon a time, in a magical forest filled with shimmering trees and sparkling rivers, there lived a wise old wizard named Merlin...

You: Explain the concept of black holes in space.
Chatbot: Black holes are fascinating astronomical objects with gravitational forces so strong that nothing, not even light, can escape from them...

You: What is your favourite book?
Chatbot: My favourite book is "The Adventures of Sherlock Holmes" by Sir Arthur Conan Doyle…