**PowerPoint Presentation: Mastering Large Language Models (LLMs)**

### **Slide 1: Title Slide**

* **Title:** Mastering Large Language Models (LLMs)
* **Subtitle:** From Basics to Deployment
* **Presented by:** [Your Name/Organization]

### **Slide 2: Course Overview**

* 8 Modules covering NLP, Transformers, LLMs, and Deployment
* Hands-on with tools like Hugging Face, LangChain, OpenAI API
* Real-world projects: summarization, chatbots, RAG-based apps

### **Slide 3: Prerequisites**

* Python Programming
* Basics of Machine Learning
* Intro to Neural Networks
* Optional: NLP Basics

### **Slide 4: Module 1 - Introduction to NLP**

* What is NLP?
* Key tasks: Tokenization, POS tagging, NER
* Rule-based vs Machine Learning approaches

### **Slide 5: Module 2 - Neural Networks for Language**

* Word Embeddings: Word2Vec, GloVe
* RNNs, LSTM, GRU
* Encoder-Decoder architectures

### **Slide 6: Module 3 - Transformers and Attention**

* Problem with RNNs
* Self-attention mechanism
* Transformer architecture
* Positional encoding, Multi-head attention

### **Slide 7: Module 4 - Understanding LLMs**

* What are Language Models?
* BERT, GPT, T5, PaLM, LLaMA
* Pretraining vs Fine-tuning
* Inference and Use Cases

### **Slide 8: Module 5 - Hands-on with LLMs**

* Hugging Face Transformers
* GPT-2/3 for text generation
* Prompt Engineering: Zero-shot, Few-shot
* Fine-tuning small models

### **Slide 9: Module 6 - Retrieval-Augmented Generation (RAG)**

* Concept of RAG
* FAISS/Chroma for vector stores
* LangChain + LLM for QA bots
* Build your custom LLM-based chatbot

### **Slide 10: Module 7 - Ethics, Safety, and Limitations**

* Bias and fairness in LLMs
* Hallucinations and content moderation
* Guardrails and responsible use

### **Slide 11: Module 8 - Production Deployment**

* LLMs via API (OpenAI, Cohere)
* Local hosting & optimization
* LangChain, Streamlit for app interfaces
* LLMOps: Monitoring and Cost Management

### **Slide 12: Projects & Tools**

* Projects:
  + Blog Generator
  + Summarizer
  + RAG Chatbot
* Tools:
  + Hugging Face, LangChain
  + OpenAI, Streamlit, Gradio

### **Slide 13: Recommended Free Courses**

* Hugging Face: <https://huggingface.co/learn/nlp-course>
* DeepLearning.AI: <https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/>
* Coursera NLP Specialization
* Fast.ai, Karpathy’s Zero to Hero

### **Slide 14: Final Slide**

* **Thank You!**
* Questions?
* Contact: [Your Contact Info]

### **Bonus: Real-World Projects with Hugging Face Pipelines**

#### 1. **Text Generation - Blog Idea Expander**

from transformers import pipeline  
  
pipe = pipeline('text-generation', model='gpt2')  
prompt = "The role of AI in personalized education"  
print(pipe(prompt, max\_length=100)[0]['generated\_text'])

#### 2. **Text Classification - News Sentiment Analyzer**

from transformers import pipeline  
classifier = pipeline("text-classification")  
print(classifier("Stock market is expected to rise sharply next week."))

#### 3. **Summarization - Article Summarizer**

from transformers import pipeline  
summarizer = pipeline("summarization")  
text = """(insert long news article text here)"""  
print(summarizer(text, max\_length=100, min\_length=30, do\_sample=False))

#### 4. **Translation - Language Localizer**

from transformers import pipeline  
translator = pipeline("translation\_en\_to\_fr")  
print(translator("Machine learning will change the future of healthcare."))

#### 5. **Zero-shot Classification - Resume Skill Matching**

from transformers import pipeline  
classifier = pipeline("zero-shot-classification")  
sequence = "He has 5 years of experience in Python and DevOps."  
candidate\_labels = ["Software Development", "DevOps", "Data Science"]  
print(classifier(sequence, candidate\_labels))

#### 6. **Feature Extraction - Semantic Search**

from transformers import pipeline  
extractor = pipeline("feature-extraction")  
features = extractor("OpenAI has transformed the AI ecosystem.")  
print(features[0][0][:10]) # Show first 10 values of first token

#### 7. **Image-to-Text - Caption a Product Photo**

from transformers import pipeline  
captioner = pipeline("image-to-text", model="nlpconnect/vit-gpt2-image-captioning")  
from PIL import Image  
img = Image.open("product.jpg")  
print(captioner(img))

#### 8. **Image Classification - Quality Control for Manufacturing**

from transformers import pipeline  
classifier = pipeline("image-classification")  
img = Image.open("sample.jpg")  
print(classifier(img))

#### 9. **Object Detection - Detect Tools on Factory Floor**

from transformers import pipeline  
from PIL import Image  
img = Image.open("factory.jpg")  
detector = pipeline("object-detection")  
print(detector(img))

#### 10. **Automatic Speech Recognition - Meeting Transcriber**

from transformers import pipeline  
asr = pipeline("automatic-speech-recognition")  
print(asr("meeting\_audio.wav"))

#### 11. **Audio Classification - Detect Alarms in Audio**

from transformers import pipeline  
classifier = pipeline("audio-classification")  
print(classifier("alarm.wav"))

#### 12. **Text-to-Speech - Voice-Enable a Chatbot**

from TTS.api import TTS  
tts = TTS(model\_name="tts\_models/en/ljspeech/tacotron2-DDC", progress\_bar=False, gpu=False)  
tts.tts\_to\_file(text="Hello, your order has been placed successfully!", file\_path="output.wav")

#### 13. **Image+Text to Text - Visual Q&A System**

from transformers import pipeline  
pipe = pipeline("image-to-text", model="Salesforce/blip-image-captioning-base")  
img = Image.open("context\_image.jpg")  
print(pipe(img))