Noakhali Science & Technology University
Department of Computer Science & Telecommunication Engineering

Assignment On:

Large Language Model (LLM)

Course Code: CSTE 3211

Submitted By

Mohammad Borhan Uddin
ID: ASH2101008M

Submitted To

Iftekhar Mahmud Tawhid
Assistant Professor
Department of Computer Science & Telecommunication Engineering

Date : 18th September, 2025

Contents

1 Introduction

A Large Language Model (LLM) is a type of artificial intelligence program built upon deep learning architectures and pre-trained on massive collections of text. Its primary purpose is to understand, generate, and manipulate human language with a high degree of coherence and relevance. Examples of widely known LLMs include GPT and BERT.

The development of LLMs began with simpler language models based on statistical methods and n-grams. The first generation of neural network-based language models appeared in the early 2000s, with models like the feedforward neural language model introduced by Bengio et al. in 2003. These early models demonstrated the potential of deep learning to capture linguistic patterns more effectively than traditional statistical approaches. Subsequent advances, such as recurrent neural networks and transformers, paved the way for modern LLMs capable of understanding context over long sequences and generating highly coherent text.

At its core, an LLM operates based on probability. Its fundamental objective is to predict the most likely next word or token in a sequence, given all preceding words. Although this task may appear simple, when executed at a large scale with millions or billions of parameters, it enables the model to perform a wide range of sophisticated language tasks. By learning statistical patterns, grammar, factual knowledge, and reasoning structures from its training data, an LLM can generate text that is both grammatically correct and contextually appropriate.

This capability opens the door to a variety of applications, including:

- **Creative Writing:** Generating stories, poetry, song lyrics, and marketing content.
- **Summarization:** Condensing long articles, reports, or transcripts into concise summaries.
- **Translation:** Translating text between numerous languages.
- **Question Answering:** Serving as a knowledge resource capable of answering questions across diverse topics.
- **Code Generation and Assistance:** Writing, explaining, and debugging code in multiple programming languages.
- **Conversational Agents:** Powering advanced chatbots and virtual assistants for customer support, tutoring, or companionship.
- **Content Moderation:** Detecting and filtering harmful or inappropriate online content.

2 Historical Development of LLM

2.1 The Seq2Seq Framewor

The development of modern language models began with the Encoder-Decoder architecture, also known as Sequence-to-Sequence (Seq2Seq) learning. Prior to this, handling tasks where the input and output were sequences of different lengths—such as machine translation—posed a significant challenge. The seminal work by Sutskever, Vinyals, and Le (2014) proposed a solution: an **encoder** neural network compresses the entire input sequence into a single, fixed-length context vector, which a **decoder** network then uses to generate the output sequence[1]. While this framework demonstrated that neural networks could perform complex sequence transduction, it also introduced a critical limitation: the fixed-size context vector became an **information bottleneck**. For long sequences, the encoder struggled to retain all information, causing the model to forget important details from the beginning of the input, which significantly degraded performance.

2.2 The Attention Mechanism

The Seq2Seq bottleneck was overcome with the introduction of the **Attention Mechanism**. Bahdanau, Cho, and Bengio (2014) revolutionized the architecture by allowing the decoder to access all encoder states directly, rather than relying solely on a single compressed vector[2]. At each step of generating an output, the decoder learns to assign dynamic weights to the input sequence, effectively attending to the most relevant words. This innovation dramatically improved long-sequence processing and maintained coherent understanding of the entire input context.

2.3 The Transformer Model

Despite the power of attention, early models were still based on Recurrent Neural Networks (RNNs), which process data sequentially. Sequential processing limited parallel computation and slowed training. Vaswani et al. (2017) introduced the **Transformer** architecture, which eliminated recurrence entirely[3]. The core innovation was **self-attention**, enabling the model to weigh the importance of all

[1]Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *Advances in Neural Information Processing Systems, 27.* `https://papers.nips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf`

[2]Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473.* `https://arxiv.org/abs/1409.0473`

[3]Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention Is All You Need. *arXiv preprint arXiv:1706.03762.* `https://arxiv.org/abs/1706.03762`

words in a sequence simultaneously. This design allowed full parallelization, significantly faster training, and better scalability. Additional mechanisms such as multi-head attention and positional encoding further enhanced the architecture's power and flexibility.

2.4 Transfer Learning

With scalable architectures in place, efficiency became a key challenge. **Transfer learning** addressed this by pre-training large Transformer models on massive unlabeled text corpora using self-supervised objectives such as language modeling. The pre-trained models acquire a deep, general-purpose understanding of language, grammar, and world knowledge. They can then be **fine-tuned** on smaller, task-specific datasets, achieving high performance with relatively low computational cost. Notable models using this paradigm include BERT[4] and GPT[5].

2.5 Scaling to Large Language Models

Finally, researchers applied **scaling laws**, increasing model parameters, dataset size, and computational resources. This approach led to not only incremental improvements but also **emergent abilities** such as in-context learning, multi-step reasoning, and instruction following[6]. These scaled Transformer models became the first true **Large Language Models (LLMs)** such as GPT-3, demonstrating the feasibility of general-purpose, task-agnostic engines for natural language understanding and generation.

3 Working Principles of Transformers and LLM

The Transformer model, introduced by Vaswani et al. in *"Attention Is All You Need"* (2017)[7], represents a significant advancement in natural language processing. Unlike Recurrent Neural Networks (RNNs) or Long Short-Term Memory

[4]Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*. `https://arxiv.org/abs/1810.04805`

[5]Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving Language Understanding by Generative Pre-Training. `https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf`

[6]Kaplan, J., McCandlish, S., Henighan, T., Brown, T., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361*. `https://arxiv.org/abs/2001.08361`

[7]Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention Is All You Need. *arXiv preprint arXiv:1706.03762*. `https://arxiv.org/abs/1706.03762`

networks (LSTMs), the Transformer does not rely on sequential processing of input data. Instead, it employs a **self-attention mechanism** that allows the model to consider all elements of the input sequence simultaneously. This architecture resolves long-term dependency issues and information bottlenecks, forming the backbone of modern Large Language Models (LLMs) such as GPT, BERT, and LLaMA.

3.1 Core Components of the Transformer

- **Input Embeddings:** Converts tokens into dense numerical vectors.
- **Positional Encoding:** Adds sequential information to token embeddings, compensating for the order-agnostic nature of self-attention.
- **Encoder:** Processes input sequences through multiple layers of self-attention and feed-forward networks, producing contextualized representations.
- **Decoder:** Generates output sequences step by step using encoder outputs and previously generated tokens.
- **Attention Mechanism:** Calculates relationships among all tokens, focusing on the most relevant information.
- **Feed-Forward Networks:** Apply non-linear transformations to enhance token representations.
- **Residual Connections and Layer Normalization:** Stabilize training and maintain information across layers.

3.2 Self-Attention Mechanism

Self-attention enables the model to weigh the importance of each token relative to others in the sequence. For each token, three vectors are computed:

- **Query (Q)**
- **Key (K)**
- **Value (V)**

Attention scores are obtained by calculating the similarity between the Query of one token and the Key of another. The output representation is the weighted sum of Value vectors. This mechanism allows the model to capture meaningful relationships between words, enabling long-range dependencies to be learned efficiently.

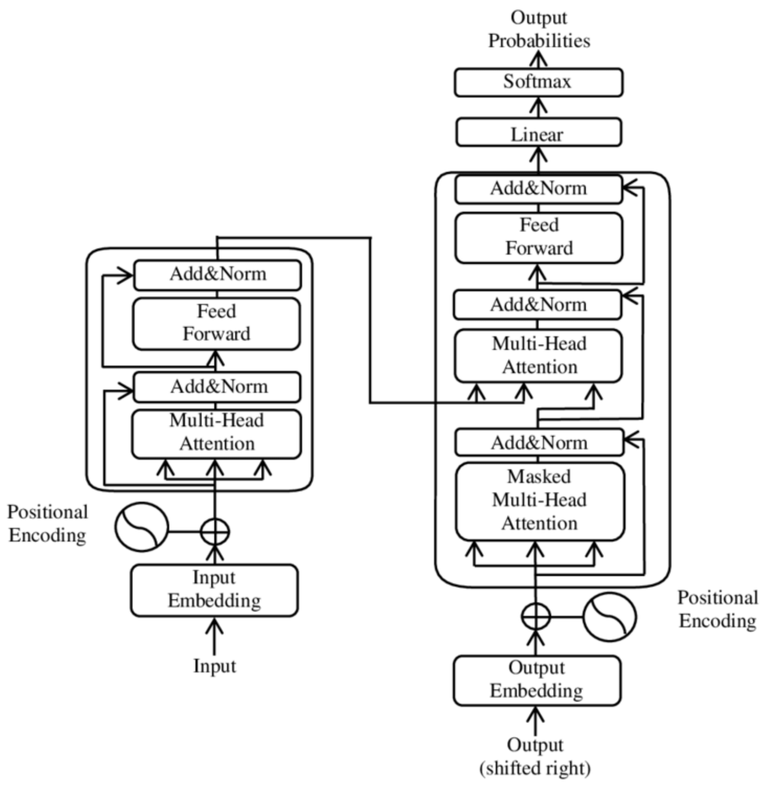

Figure 1: The Transformer Attention mechanism (Source: Wikipedia)

3.3 Multi-Head Attention

The Transformer uses multiple attention “heads” in parallel. Each head learns different aspects of token relationships, such as syntactic patterns or semantic meaning. Outputs from all heads are concatenated and passed through feed-forward networks, producing rich contextual representations.

3.4 Encoder-Decoder Workflow

Encoder:

- Receives input tokens with embeddings and positional encodings.
- Processes sequences via multi-head self-attention and feed-forward networks.
- Produces contextualized vectors representing the input.

Decoder:

- Receives encoder output and previous tokens.
- Uses masked self-attention to prevent peeking at future tokens.

- Integrates encoder-decoder attention to align outputs with relevant input positions.
- Generates output tokens sequentially until an End-of-Sequence token is produced.

3.5 Output Generation

The decoder produces a probability distribution over the vocabulary for each step. Tokens are generated autoregressively: the most probable token is selected and fed back into the decoder to predict the next token. This continues until the sequence is complete. For example, given the input *"I am Borhan"*, the decoder could generate the output sequence:

$$\text{“I”} \rightarrow \text{“am”} \rightarrow \text{“Borhan”}.$$

3.6 Large Language Models (LLMs)

LLMs are built upon the Transformer architecture and scale it to billions of parameters. They are trained on massive text corpora and follow a two-stage process: **pre-training** and **fine-tuning**[8].

Pre-Training: The model learns general language representations from large unlabeled corpora. Strategies include:

- **Masked Language Modeling (MLM):** Some tokens are masked and the model predicts them (BERT).
- **Causal Language Modeling (CLM):** The model predicts the next token in a sequence autoregressively (GPT).

Fine-Tuning: The model is adapted to task-specific datasets for applications such as sentiment analysis, text classification, translation, and summarization. Fine-tuning leverages knowledge gained during pre-training to achieve high performance on smaller datasets.

Inference: During inference, LLMs generate outputs for unseen input text by processing it through Transformer layers and sequentially producing tokens in an autoregressive manner.

[8]Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*. `https://arxiv.org/abs/1810.04805`; Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving Language Understanding by Generative Pre-Training. `https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf`

3.7 Scaling LLMs

Scaling involves increasing model parameters, dataset size, and computational resources. OpenAI and other labs discovered that larger models exhibit **emergent abilities**, such as in-context learning, multi-step reasoning, and instruction following[9]. This scaling led to the creation of the first true LLMs, such as GPT-3, capable of general-purpose language understanding and generation.

In summary, modern LLMs are built on Transformer architectures using self-attention, multi-head attention, and feed-forward networks. Through pre-training on vast corpora and fine-tuning on specific tasks, these models efficiently capture complex patterns, long-range dependencies, and semantic relationships in text, enabling a wide variety of natural language processing applications.

4 Tokenization and Embedding Methods

Before processing text, Large Language Models (LLMs) convert raw text into a numerical format that the model can understand. This process begins with **tokenization**, where text is split into smaller units such as words, subwords, or characters. Modern LLMs primarily rely on **subword tokenization** methods, including:

- **Byte-Pair Encoding (BPE):** Iteratively merges the most frequent pairs of characters or subwords to build a vocabulary.
- **WordPiece:** Constructs a subword vocabulary based on likelihood maximization, as employed in BERT.
- **SentencePiece:** Learns a language-independent subword vocabulary and operates directly on raw text without pre-tokenization.

Once tokenized, each token is mapped to a **dense vector embedding**, capturing semantic information. Because Transformers do not process sequences sequentially, **positional encoding** is added to preserve token order. This ensures that semantically distinct sequences, such as *"dog bites man"* and *"man bites dog"*, are differentiated.

[9]Kaplan, J., McCandlish, S., Henighan, T., Brown, T., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361.* `https://arxiv.org/abs/2001.08361`

5 Types of Large Language Models

LLMs can be categorized according to architecture and training objectives, with each type suited to specific tasks:

1. **Autoregressive Models**: Predict the next token in a sequence given all previous tokens. Ideal for text generation, dialogue, and completion tasks. *Example, GPT.*
2. **Masked Language Models** : Randomly mask tokens and predict them based on surrounding context. Primarily used for understanding tasks such as classification, question answering, and sentence embeddings. *Example, BERT.*
3. **Encoder-Decoder Models**): Employ an encoder to process input sequences and a decoder to generate outputs. Suitable for translation, summarization, and other sequence-to-sequence tasks.

In general, **autoregressive models excel in generation**, **masked models excel in understanding**, and **encoder-decoder models are optimal for input-to-output transformations**.

6 Limitations and Challenges

Despite their remarkable capabilities, LLMs face several limitations:

- **Biases in Training Data:** Models can inherit and amplify biases present in the corpus, leading to unfair or harmful outputs.
- **Hallucinations:** LLMs may produce plausible but factually incorrect content.
- **Computational and Environmental Costs:** Training and deploying large models require extensive computational resources and energy.
- **Difficulty Handling Very Long Sequences:** Extremely long sequences can exceed model context windows, affecting coherence and performance.

7 Ethical Considerations

While LLMs provide transformative tools for language understanding and generation, their deployment introduces ethical responsibilities:

- **Bias and Fairness:** Models can reflect societal biases, risking discriminatory outputs.
- **Misinformation and Hallucinations:** Incorrect or misleading outputs can propagate false narratives.
- **Privacy Concerns:** Training on sensitive data may inadvertently expose confidential information.
- **Environmental Impact:** Large-scale model training consumes substantial energy.
- **Misuse Potential:** Models could be exploited for disinformation, phishing, or spam.

8 Future Opportunities

Research is actively addressing LLM limitations while expanding their capabilities:

- **Enhanced Multimodal Capabilities:** Combining text with images, audio, and video for richer AI understanding.
- **Long-Context and Memory-Augmented Models:** Processing very long sequences for improved document analysis and dialogue systems.
- **Personalized and Adaptive LLMs:** Adapting to individual user preferences while maintaining privacy.
- **Energy-Efficient Models:** Optimizing models for lower computational and environmental costs.
- **Robustness and Bias Mitigation:** Improving factual accuracy and reducing harmful outputs.
- **Cross-Lingual and Low-Resource Models:** Expanding support for underserved languages.
- **Integration with Knowledge Bases:** Combining LLMs with structured data to enhance reasoning and decision-making.

9 Conclusion

Large Language Models represent a paradigm shift in natural language processing. By leveraging the **Transformer architecture**, **transfer learning**, and massive-scale data, they achieve unprecedented performance across diverse tasks, from creative writing and translation to code generation and conversational AI.

However, the deployment of LLMs carries ethical responsibilities, including addressing biases, misinformation, environmental impact, and potential misuse. Ongoing research focuses on improving safety, efficiency, adaptability, and inclusivity, ensuring that LLMs can be applied responsibly and effectively.

In essence, LLMs are not merely tools for text generation—they are evolving, general-purpose engines for human-computer interaction, capable of transforming knowledge discovery, creative expression, and communication when developed and deployed thoughtfully.