ROAD MAP FOR STUDYING LARGE LANGUAGE MODELS

Mathematics For Machine Learning

Before mastering machine learning, It is important to understand the fundamental mathematical concepts that power these algorithms.

1. Linear Algebra:

- **Vectors:** Vectors represent both data and model parameters. Understanding vector operations, dot products, and vector spaces is essential.
- **Matrices:** Matrices are fundamental for many machine learning operations. Learn about matrix multiplication, inverses, and transformations.
- **Determinants:** Understand the determinant of a matrix and its importance in solving systems of linear equations and computing inverses.
- **Eigenvalues and Eigenvectors:** These are crucial for understanding transformations and diagonalization of matrices, which are common in machine learning.
- **Vector Spaces:** Gain a deep understanding of vector spaces and subspaces, which are fundamental to linear algebra in machine learning.

2. Calculus:

- **Differential Calculus:** Learn about derivatives and how they are used in optimization algorithms, such as gradient descent.
- **Integral Calculus:** Understand integrals and their role in computing areas, volumes, and probabilities.
- **Limits and Continuity:** Study limits to grasp the idea of approaching a value as it gets infinitely close and continuity of functions.
- **Multivariable Calculus:** Extend your calculus knowledge to functions of multiple variables, necessary for optimizing complex functions.
- **Gradients:** Explore the concept of gradients and their use in optimization techniques like gradient descent.

3. Probability and Statistics:

- **Probability Theory:** Understand basic probability concepts, including events, random variables, and probability distributions.
- **Random Variables:** Learn about discrete and continuous random variables, probability mass functions, and probability density functions.
- **Probability Distributions:** Explore common probability distributions like the Gaussian (normal) distribution and the binomial distribution.
- **Expectations:** Study expected values and moments of random variables, which are used in model building and analysis.
- **Variance, Covariance, and Correlation:** Understand measures of variability and the relationships between variables.
- **Hypothesis Testing:** Learn about statistical hypothesis testing, p-values, and significance levels.
- **Confidence Intervals:** Understand how to construct confidence intervals for estimating population parameters.
- **Maximum Likelihood Estimation:** Learn about the method for estimating model parameters that maximize the likelihood function.
- **Bayesian Inference:** Explore the Bayesian approach to statistical inference and probability.

4. Optimization:

- **Gradient Descent:** Master the concept of gradient descent and its variants, which are the primary optimization techniques for training machine learning models.

- **Convex Optimization:** Understand the principles of convex optimization, which play a crucial role in many machine learning algorithms.
- **Stochastic Gradient Descent (SGD):** Learn about the stochastic variant of gradient descent, which is widely used for large datasets.
- **Hyperparameter Tuning:** Study how to optimize hyperparameters to fine-tune model performance.
- **Optimization Libraries:** Familiarize yourself with optimization libraries like SciPy and TensorFlow's optimization modules.

Resources :-

- ✔ 3Blue1Brown - The Essence of Linear Algebra: Series of videos that give a geometric intuition to these concepts.
- ✔ StatQuest with Josh Starmer - Statistics Fundamentals: Offers simple and clear explanations for many statistical concepts.
- ✔ AP Statistics Intuition by Ms Aerin: List of Medium articles that provide the intuition behind every probability distribution.
- ✔ Immersive Linear Algebra: Another visual interpretation of linear algebra.
- ✔ Khan Academy - Linear Algebra: Great for beginners as it explains the concepts in a very intuitive way.
- ✔ Khan Academy - Calculus: An interactive course that covers all the basics of calculus.
- ✔ Khan Academy - Probability and Statistics: Delivers the material in an easy-to-understand format.

PyTorch For Machine Learning

PyTorch is an open-source machine learning library developed by Facebook's AI Research lab (FAIR). It is primarily used for deep learning and artificial intelligence research, as well as for building and training machine learning models. PyTorch provides a flexible and dynamic computational framework that allows developers to define and manipulate computational graphs in a more intuitive way compared to some other deep learning frameworks.

1. Basics of PyTorch:

- **Tensors:** Learn how to create and manipulate tensors, the fundamental data structure in PyTorch.
- **Operations:** Explore tensor operations, including arithmetic operations, reshaping, and element-wise operations.
- **Autograd:** Understand PyTorch's automatic differentiation capabilities, which are crucial for gradient-based optimization.

2. Building and Training Models:

- **Neural Networks:** Dive into the world of neural networks, understanding layers, activation functions, and building custom neural network architectures.
- **Loss Functions:** Learn about common loss functions used for different types of tasks (e.g., mean squared error for regression, cross-entropy for classification).
- **Optimizers:** Explore various optimization algorithms available in PyTorch, such as SGD, Adam, and RMSprop.
- **Model Training:** Understand the process of training a model, including forward and backward passes, weight updates, and mini-batch processing.

3. **Dataset Handling:**

- **Data Loading:** Explore PyTorch's data loading utilities, including DataLoader and custom data loading pipelines.
- **Data Augmentation:** Learn techniques for data augmentation to increase the diversity of training data.
- **Data Preprocessing:** Understand data preprocessing steps, such as normalization and data splitting.

4. **Deep Learning Techniques:**

- **Convolutional Neural Networks (CNNs):** Study CNNs for computer vision tasks, including image classification and object detection.
- **Recurrent Neural Networks (RNNs):** Learn about RNNs for sequence modeling, such as natural language processing and time series analysis.
- **Transfer Learning:** Explore how to leverage pre-trained models and fine-tune them for specific tasks.
- **Generative Adversarial Networks (GANs):** Delve into GANs for tasks like image generation and style transfer.
- **Reinforcement Learning:** Understand reinforcement learning principles for applications like game playing and robotics.

5. **Model Evaluation and Validation:**

- **Metrics:** Learn about evaluation metrics, such as accuracy, precision, recall, F1-score, and mean absolute error.
- **Cross-Validation:** Implement techniques like k-fold cross-validation to assess model performance.
- **Overfitting and Regularization:** Explore strategies to prevent overfitting, such as dropout and weight decay.

6. **Deployment and Production:**

- **Model Deployment:** Understand how to deploy PyTorch models in production environments, using tools like TorchScript.
- **Serving Models:** Learn about serving models through web services or other deployment methods.
- **Model Optimization:** Optimize models for production, including quantization and reducing model size.

7. **Advanced Topics:**

- **PyTorch Lightning:** Explore the PyTorch Lightning framework for cleaner and more organized code.
- **Distributed Training:** Learn how to train models on distributed systems using PyTorch.
- **Custom Layers and Loss Functions:** Develop custom layers and loss functions tailored to specific tasks.
- **Interoperability:** Understand how to integrate PyTorch with other libraries and frameworks.

Resources :-

- ✔ PyTorch Beginner Series From PyTorch
- ✔ PyTorch Fundamentals on Youtube by freeCodeCamp
- ✔ 24 HOURS PyTorch for DeepLearning by Daniel Bourke
- ✔ PyTorch Python DeepLearning Neural Network API
- ✔ PyTorch Tutorials by Aladdin Persson
- ✔ PyTorch Tutorials – Complete Beginner Course by Patrick Loeber
- ✔ PyTorch for DeepLearning by Sentdex

Neural Networks

Week 1: Introduction to Machine Learning

- Understand the basics of machine learning, supervised and unsupervised learning, and the role of neural networks in machine learning.
- Explore essential concepts like data, features, and labels.

Week 2: Fundamentals of Neural Networks

- Learn about the basic building blocks of neural networks: neurons and layers.
- Explore activation functions, weights, and biases.
- Understand the feedforward process in neural networks.

Week 3: Training Neural Networks

- Dive into the concept of loss functions and how they measure model performance.
- Understand backpropagation and gradient descent, the fundamental techniques used to train neural networks.
- Implement a simple neural network in a programming language like Python.

Week 4: Deep Learning and Neural Network Architectures

- Explore the difference between shallow and deep neural networks.
- Learn about different types of neural network architectures, including feedforward neural networks and convolutional neural networks (CNNs).
- Discuss applications of deep learning in image and text data.

Intermediate Level: Weeks 5-8

Week 5: Convolutional Neural Networks (CNNs)

- Deep dive into CNNs and their applications in computer vision tasks.
- Understand the importance of convolutional and pooling layers.
- Work on image classification projects using CNNs.

Week 6: Recurrent Neural Networks (RNNs)

- Explore RNNs for sequence data, such as time series and natural language processing.
- Learn about the challenges of vanishing gradients and long short-term memory (LSTM) and gated recurrent unit (GRU) architectures.

Week 7: Autoencoders and Generative Adversarial Networks (GANs)

- Understand autoencoders for dimensionality reduction and unsupervised learning.
- Learn about GANs for generating new data samples.
- Experiment with GANs for image generation.

Week 8: Advanced Topics in Deep Learning

- Discover advanced concepts like transfer learning, reinforcement learning, and attention mechanisms.
- Explore state-of-the-art architectures like Transformers for NLP tasks.
- Discuss ethical considerations and bias in deep learning models.

Advanced Level: Weeks 9-12

Week 9: Custom Neural Network Architectures

- Explore custom architectures, such as Siamese networks, capsule networks, and graph neural networks.
- Discuss their use cases and implement custom architectures.

Week 10: Hyperparameter Tuning and Model Optimization

- Dive into hyperparameter tuning, grid search, and random search techniques.
- Learn about regularization methods and optimization algorithms.
- Optimize neural network models for performance.

Week 11: Deployment and Real-World Applications

- Understand the process of deploying neural network models in production.
- Explore real-world applications of neural networks in industries like healthcare, finance, and autonomous vehicles.
- Learn about model interpretation and explainability.

Week 12: Advanced Projects and Research

- Work on advanced projects that apply neural networks to complex problems.
- Dive into current research papers and participate in discussions about the latest developments in neural networks.

Ongoing Learning: Beyond Week 12

- Stay updated with the latest research and developments in the field of neural networks by reading research papers, blogs, and following experts on platforms like arXiv, Medium, and LinkedIn.
- Continue building projects to deepen your practical experience and expertise.
- Explore specialized areas like natural language processing (NLP), computer vision, and reinforcement learning, depending on your interests and career goals.

Natural Language Processing (NLP)

NLP is a fascinating branch of artificial intelligence that bridges the gap between human language and machine understanding. From simple text processing to understanding linguistic nuances, NLP plays a crucial role in many applications like translation, sentiment analysis, chatbots, and much more.

Week 1-2: Introduction to Natural Language Processing (NLP)

- Overview of NLP and its applications.
- Text data preprocessing.
- Tokenization, stemming, and lemmatization.
- Text normalization techniques.

Week 3-4: Word Vectors and Word Embeddings

- Word representation in NLP.
- Introduction to Word2Vec, GloVe, and FastText.
- Training word embeddings.
- Pre-trained word embeddings and their usage.

Week 5-6: Text Processing Techniques

- Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) representations.
- Text classification with BoW and TF-IDF.
- N-grams and feature engineering.

Week 7-8: Neural Classifiers

- Introduction to neural networks for NLP.
- Building simple feedforward neural networks for text classification.
- Activation functions, loss functions, and backpropagation.
- Regularization techniques.

Week 9-10: Recurrent Neural Networks (RNNs)

- Understanding sequential data.
- Introduction to RNNs.
- Vanishing gradient problem.
- Building and training RNNs for text data.

Week 11-12: Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU)

- The need for LSTMs and GRUs.
- Architecture and functioning of LSTMs and GRUs.
- Sequence modeling with LSTMs and GRUs.
- Text generation using LSTMs.

Week 13-14: Machine Translation

- Introduction to machine translation.
- Statistical Machine Translation vs. Neural Machine Translation.
- Sequence-to-sequence models for machine translation.
- Attention mechanisms in machine translation.

Week 15-16: Text Summarization

- Extractive vs. abstractive summarization.
- Abstractive summarization with LSTMs and Transformers.
- Evaluation metrics for text summarization.
- Building your own summarization model.

Week 17-18: Language Modeling

- Introduction to language modeling.
- N-gram language models.
- Recurrent Neural Network-based language models.
- Evaluation metrics for language models.

Week 19-20: Question Answering

- Overview of question answering tasks.
- Building a simple rule-based QA system.
- Deep learning-based QA models.
- Datasets and evaluation metrics for QA.

Week 21-22: Sequence to Sequence Models

- Understanding sequence-to-sequence tasks.
- Encoder-Decoder architectures.
- Building sequence-to-sequence models for tasks like chatbots.
- Evaluation metrics for sequence-to-sequence models.

Week 23-24: Self-Attention and Transformers

- Introduction to self-attention mechanisms.
- The Transformer architecture.
- Building a Transformer model from scratch.
- Transfer learning with Transformers.

Week 25-26: Pre-training Transformers

- Pre-training vs. fine-tuning in NLP.
- Pre-trained language models (e.g., BERT, GPT).
- Fine-tuning pre-trained models for specific NLP tasks.

- Using Hugging Face Transformers library.

Resources :-

- ✔ CS224N :- NLP With Deep Learning By Christopher Manning
- ✔ Sequence Models by Coursera
- ✔ NLP With Deep Learning PyTorch Tutorial From Stanford
- ✔ NLP With Deep Learning Hugging Face Tutorial From Stanford
- ✔ Natural Language Processing By Jovian With PyTorch
- ✔ 12 hours DeepLearning For Natural Language Processing Complete Course

The Transformer Architecture

The Transformer model, introduced in the "Attention is All You Need" paper, is the neural network architecture at the core of large language models. The original paper is difficult to read and even contains some mistakes, which is why alternative resources are recommended.

Week 1: Introduction to Transformers

- Overview of NLP and the need for Transformers.
- The limitations of RNNs and LSTMs in sequence modeling.
- Key concepts: self-attention, multi-head attention, position encoding.

Week 2: Self-Attention Mechanism

- In-depth understanding of self-attention mechanism.
- The concept of attention scores.
- Calculating attention scores and weighted values.
- Multi-head attention and its advantages.

Week 3: Positional Encoding

- Why positional encoding is necessary.
- Different positional encoding techniques (e.g., sine and cosine functions).
- Adding positional encodings to input sequences.

Week 4: Transformer Architecture

- Building blocks of the Transformer: encoder and decoder.
- Stacking encoder and decoder layers.
- Residual connections and layer normalization.

Week 5: Training Transformers

- Training objectives: masked language modeling (MLM) and next sentence prediction (NSP).
- Pre-training vs. fine-tuning.
- BERT (Bidirectional Encoder Representations from Transformers) and its architecture.

Week 6: Fine-tuning Transformers

- Fine-tuning for various NLP tasks (e.g., text classification, named entity recognition).

- Datasets and data preprocessing for fine-tuning.
- Hyperparameter tuning and regularization.

Week 7: Transformers for Text Classification

- Using pre-trained Transformers for text classification.
- Implementing text classification models using Transformers (e.g., BERT, RoBERTa).
- Fine-tuning for custom classification tasks.

Week 8: Transformers for Named Entity Recognition

- Introduction to Named Entity Recognition (NER).
- Building NER models using Transformers.
- Training on NER datasets like CoNLL-2003.

Week 9: Transformers for Question Answering

- Understanding the question answering task.
- Implementing question answering models using Transformers.
- Fine-tuning on QA datasets like SQuAD.

Week 10: Transformers for Language Generation

- Introduction to text generation tasks.
- GPT (Generative Pre-trained Transformer) architecture.
- Fine-tuning GPT for text generation tasks.

Week 11: Transformers for Machine Translation

- Seq2Seq with Transformers for machine translation.
- Using the Transformer architecture for both encoder and decoder.
- Training on translation datasets like WMT.

Week 12: Attention Mechanisms Beyond Transformers

- Other attention mechanisms: local attention, sparse attention, and more.
- Research advancements and hybrid models.
- Ethical considerations in AI and NLP.

Week 13-14: Advanced Topics

- Advanced Transformer-based models and architectures (e.g., T5, XLNet, GPT-3).
- Understanding the latest research papers in the field.
- Project work: Implement a research paper or develop a custom NLP application using Transformers.

Resources :-

- ➔ Stanford CS25 – Transformers United With Andrej Karpathy (2023)
- ➔ Stanford CS25 – Transformers United 2022
- ➔ The Transformer Architecture by Andrew Ng
- ➔ Stanford CS224N – Transformers
- ➔ Introduction to the Transformer by Rachel Thomas from University of SAN FRANCISCO
- ➔ The Transformer Architecture by Sebastian Raschaka
- ➔ MIT Recurrent Neural Networks , Transformers, and Attention

Pre-Trained Language Models

Week 1-2: Introduction to Pre-trained Language Models

- Overview of the significance of pre-trained language models in NLP.
- Introduction to traditional models like BERT, GPT-2, T5, RoBERTa, and GPT-3.
- Brief overview of the latest models: GPT-4, PaLM, Llama, ELECTRA, DeBERTa, and UniLM.
- Understand the evolution of language models and their applications.

Week 3-4: GPT-4 (Generative Pre-trained Transformer 4)

- In-depth exploration of GPT-4, its architecture, and capabilities.
- Multimodal understanding: Text and image input.
- Comparison with GPT-3 and its improvements.
- Discuss potential applications of GPT-4 in content creation, translation, and research.

Week 5-6: PaLM (Pathways Language Model)

- Dive into the architecture and training process of PaLM.
- Explore PaLM's capabilities in question answering, math problem-solving, and coding.
- Discuss practical uses of PaLM in building chatbots, providing answers, and more.

Week 7-8: Llama (Large Language Model Meta AI)

- Introduction to Llama and its open-source accessibility.
- Discover the adaptability of Llama and its various sizes.
- Understand the accidental release of Llama and its influence on related models.
- Explore potential applications of Llama in practical use and experimentation.

Week 9-10: ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately)

- Study ELECTRA's innovative pre-training approach.
- Learn about replaced token detection vs. masked language modeling.
- Discuss the computational efficiency of ELECTRA.

Week 11-12: DeBERTa (Decoding-enhanced BERT with disentangled attention)

- Explore DeBERTa's architecture and improvements over BERT.
- Understand the importance of disentangled attention and an enhanced mask decoder.
- Investigate how DeBERTa surpasses human baselines in NLP benchmarks.

Week 13-14: UniLM (Unified Language Model)

- In-depth study of the Unified Language Model developed by Microsoft Research.
- Explore its bidirectional transformer architecture and its ability to handle multiple language tasks.
- Understand how UniLM simplifies NLP applications and enhances efficiency.

Week 15: Advanced Topics and Project Work

- Discuss advanced pre-trained models like XLNet, ALBERT, and StructBERT.
- work on individual or group projects involving the latest pre-trained models.
- Project presentations and peer reviews.

Resources :-

- ✔ Stanford CS224N : NLP with Deep Learning BERT and Other pre-trained models
- ✔ Recent Advances in Vision and Language Pre-Training by CVPR
- ✔ Stanford CS224N Transformers and Pre-Training
- ✔ Generative Pre-Traind Transformer GPT-v1 by Sebastian Raschka
- ✔ Leveraging Pre-trained Language models for Natural Language understanding from Toronto
- ✔ Social Application of pre-trained language models Anjalie Field From Stanford

✔ Open Pretrained Transformers Stanford ML seminar

Advanced Language Modeling

To fine-tune your skills, learn how to create embeddings with sentence transformers, store them in a vector database, and use parameter-efficient supervised learning or RLHF to fine-tune LLMs.

✗ **Sentence Transformers**: Sentence Transformers are models that can derive semantically meaningful embeddings for sentences, paragraphs, or texts. Learn how to store and retrieve these embeddings using an appropriate vector database for rapid similarity search.

✗ **Fine-Tuning Language Models:** After understanding and using pre-trained models, the next step is fine-tuning them on a domain-specific dataset. It allows the model to be more accurate for certain tasks or domains, such as medical text analysis or sentiment analysis for movie reviews.

✗ **Parameter-Efficient Learning Techniques**: Explore more efficient ways to train or fine-tune your models without requiring massive amounts of data or computational resources, such as LoRA.

Resources :-

•SBERT.net: Python library to implement sentence transformers, with a lot of examples.

•Pinecone - Sentence Transformers: Mini-book about NLP for semantic search in general.

•Hugging Face - RLHF: Blog post introducing the concept of RLHF.

•Hugging Face - PEFT: Another library from Hugging Face implementing different techniques, such as LoRA.

•Efficient LLM training by Phil Schmid: Implementation of LoRA to fine-tune a Flan-T5 model.

Large Language Model Operations (LLMOps)

Finally, dive into Large Language Model Operations (LLMOps), learn how to handle prompt engineering, build frameworks with LangChain and Llamaindex, and optimize inference with weight quantization, pruning, distillation, and more.

✔ Fine-tuning LLaMA: Instruction fine-tuning has become extremely popular since the (accidental) release of LLaMA. The size of these models and the peculiarities of training them on instructions and answers introduce more complexity and often require parameter-efficient learning techniques such as QLoRA.

✔ Build LLM Frameworks: LLMs are a new building block in system design, where the rest of the architecture is handled by libraries such as LangChain and LlamaIndex, allowing you to query vector databases, improving the model's memory or providing various tools.

✔ Optimization Techniques for Inference: As the size of LLMs grows, it becomes increasingly important to apply optimization techniques to ensure that the models can be efficiently used for inference. Techniques include weight quantization (4-bit, 3-bit), pruning, knowledge distillation, etc.

✔ LLM deployment: These models can be deployed locally like llama.cpp or in the cloud like Hugging Face's text generation inference or vLLM.

Resources :-

- ➔ MLExpert - Fine-tuning Alpaca: Guide to fine-tune LLaMA on a custom dataset.
- ➔ Hugging Face - LLM.int8(): Introduction to 8-bit matrix multiplication with LLM.int8().
- ➔ Hugging Face - QLoRA: Blog post introducing QLoRA with notebooks to test it.
- ➔ Kanaries - AutoGPTQ: Simple guide to use AutoGPTQ.
- ➔ Emerging Architectures for LLM Applications: overview of the LLM app stack.
- ➔ Pinecone - LangChain AI Handbook: Excellent free book on how to master the LangChain library.
- ➔ A Primer to using LlamaIndex: Official guides to learn more about LlamaIndex.