[Week 1, Part 2] Domain and Task Adaptation Methods

ETMI5: Explain to Me in 5

In this section, we delve into the limitations of general AI models in specialized domains, underscoring the significance of domain-adapted LLMs. We explore the advantages of these models, including depth, precision, improved user experiences, and addressing privacy concerns.

We introduce three types of domain adaptation methods: Domain-Specific Pre-Training, Domain-Specific Fine-Tuning, and Retrieval Augmented Generation (RAG). Each method is outlined, providing details on types, training durations, and quick summaries. We then explain each of these methods in further detail with real-world examples. In the end, we provide an overview of when RAG should be used as opposed to model updating methods.

Using LLMs Effectively

While general AI models such as ChatGPT demonstrate impressive text generation abilities across various subjects, they may lack the depth and nuanced understanding required for specific domains. Additionally, these models are more prone to generating inaccurate or contextually inappropriate content, referred to as hallucinations. For instance, in healthcare, specific terms like “electronic health record interoperability” or “patient-centered medical home” hold significant importance, but a generic language model may struggle to fully comprehend their relevance due to a lack of specific training on healthcare data. This is where task-specific and domain-specific LLMs play a crucial role. These models need to possess specialized knowledge of industry-specific terminology and practices to ensure accurate interpretation of domain-specific concepts. Throughout the remainder of this course, we will refer to these specialized LLMs as **domain-specific LLM**s, a commonly used term for such models.

Here are some benefits of using domain-specific LLMs:

1. **Depth and Precision**: General LLMs, while proficient in generating text across diverse topics, may lack the depth and nuance required for specialized domains. Domain-specific LLMs are tailored to understand and interpret industry-specific terminology, ensuring precision in comprehension.
2. **Overcoming Limitations**: General LLMs have limitations, including potential inaccuracies, lack of context, and susceptibility to hallucinations. In domains like finance or medicine, where specific terminology is crucial, domain-specific LLMs excel in providing accurate and contextually relevant information.
3. **Enhanced User Experiences**: Domain-specific LLMs contribute to enhanced user experiences by offering tailored and personalized responses. In applications such as customer service chatbots or dynamic AI agents, these models leverage specialized knowledge to provide more accurate and insightful information.

4. **Improved Efficiency and Productivity**: Businesses can benefit from the improved efficiency of domain-specific LLMs. By automating tasks, generating content aligned with industry-specific terminology, and streamlining operations, these models free up human resources for higher-level tasks, ultimately boosting productivity.
5. **Addressing Privacy Concerns**: In industries dealing with sensitive data, such as healthcare, using general LLMs may pose privacy challenges. Domain-specific LLMs can provide a closed framework, ensuring the protection of confidential data and adherence to privacy agreements.

If you recall from the previous section, we had multiple ways to use LLMs in specific use cases, namely

1. **Zero-shot learning**
2. **Few-shot learning**
3. **Domain Adaptation**

Zero-shot learning and few-shot learning involve instructing the general model either through examples or by prompting it with specific questions of interest. Another concept introduced is domain adaptation, which will be the primary focus in this section. More details about the first two methods will be explored when we delve into the topic of prompting.

Types of Domain Adaptation Methods

There are several methods to incorporate domain-specific knowledge into LLMs, each with its own advantages and limitations. Here are three classes of approaches:

1. **Domain-Specific Pre-Training:**

   - ***Training Duration****:* Days to weeks to months
   - ***Summary****:* Requires a large amount of domain training data; can customize model architecture, size, tokenizer, etc.

   In this method, LLMs are pre-trained on extensive datasets representing various natural language use cases. For instance, models like PaLM 540B, GPT-3, and LLaMA 2 have been pre-trained on datasets with sizes ranging from 499 billion to 2 trillion tokens. Examples of domain-specific pre-training include models like ESMFold, ProGen2 for protein sequences, Galactica for science, BloombergGPT for finance, and StarCoder for code. These models outperform generalist models within their domains but still face limitations in terms of accuracy and potential hallucinations.

2. **Domain-Specific Fine-Tuning:**

   - ***Training Duration****:* Minutes to hours
   - ***Summary****:* Adds domain-specific data; tunes for specific tasks; updates LLM model

   Fine-tuning involves training a pre-trained LLM on a specific task or domain, adapting its knowledge to a narrower context. Examples include Alpaca (fine-tuned

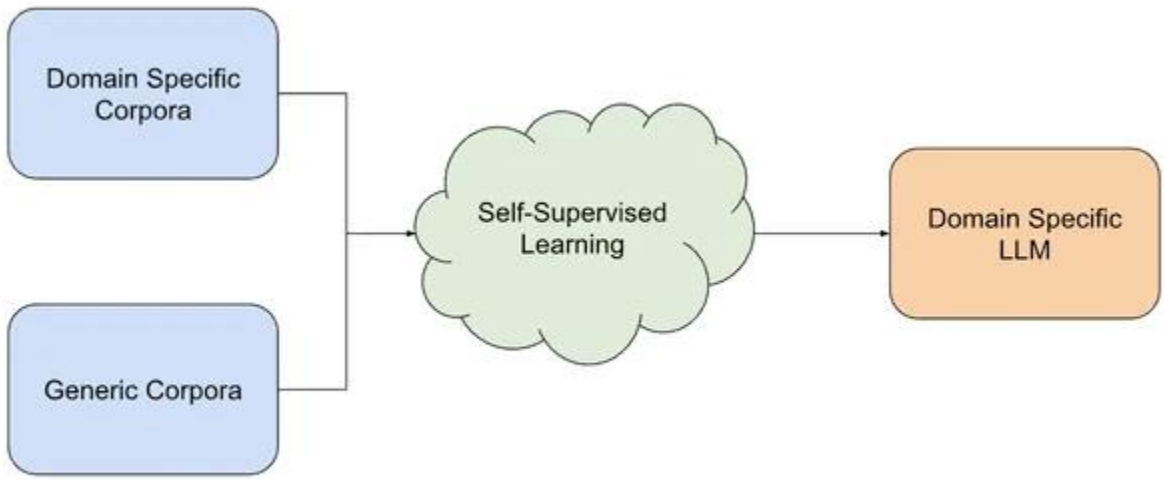
LLaMA-7B model for general tasks), xFinance (fine-tuned LLaMA-13B model for financial-specific tasks), and ChatDoctor (fine-tuned LLaMA-7B model for medical chat). The costs for fine-tuning are significantly smaller compared to pre-training.

3. **Retrieval Augmented Generation (RAG):**

   - ***Training Duration****:* Not required
   - ***Summary****:* No model weights; external information retrieval system can be tuned

   RAG involves grounding the LLM's parametric knowledge with external or non-parametric knowledge from an information retrieval system. This external knowledge is provided as additional context in the prompt to the LLM. The advantages of RAG include no training costs, low expertise requirement, and the ability to cite sources for human verification. This approach addresses limitations such as hallucinations and allows for precise manipulation of knowledge. The knowledge base is easily updatable without changing the LLM. Strategies to combine non-parametric knowledge with an LLM's parametric knowledge are actively researched.

Domain-Specific Pre-Training



domain_specific

`Image Source [https://www.analyticsvidhya.com/blog/2023/08/domain-specific-llms/](https://www.analyticsvidhya.com/blog/2023/08/domain-specific-llms/)`

Domain-specific pre-training involves training large language models on extensive datasets that specifically represent the language and characteristics of a particular domain or field. This process aims to enhance the model's understanding and performance within a defined

subject area. Let's understand domain specific pretraining through the example of BloombergGPT, a large language model for finance.

BloombergGPT is a 50 billion parameter language model designed to excel in various tasks within the financial industry. While general models are versatile and perform well across diverse tasks, they may not outperform domain-specific models in specialized areas. At Bloomberg, where a significant majority of applications are within the financial domain, there is a need for a model that excels in financial tasks while maintaining competitive performance on general benchmarks. BloombergGPT can perform the following tasks:

1. **Financial Sentiment Analysis:** Analyzing and determining sentiment in financial texts, such as news articles, social media posts, or financial reports. This helps in understanding market sentiment and making informed investment decisions.
2. **Named Entity Recognition:** Identifying and classifying entities (such as companies, individuals, and financial instruments) mentioned in financial documents. This is crucial for extracting relevant information from large datasets.
3. **News Classification:** Categorizing financial news articles into different topics or classes. This can aid in organizing and prioritizing news updates based on their relevance to specific financial areas.
4. **Question Answering in Finance:** Answering questions related to financial topics. Users can pose queries about market trends, financial instruments, or economic indicators, and BloombergGPT can provide relevant answers.
5. **Conversational Systems for Finance:** Engaging in natural language conversations related to finance. Users can interact with BloombergGPT to seek information, clarify doubts, or discuss financial concepts.

To achieve this, BloombergGPT undergoes domain-specific pre-training using a large dataset that combines domain-specific financial language documents from Bloomberg's extensive archives with public datasets. This dataset, named FinPile, consists of diverse English financial documents, including news, filings, press releases, web-scraped financial documents, and social media content. The training corpus is roughly divided into half domain-specific text and half general-purpose text. The aim is to leverage the advantages of both domain-specific and general data sources.

The model architecture is based on guidelines from previous research efforts, containing 70 layers of transformer decoder blocks (read more in the paper)

Domain-Specific Fine-Tuning

Domain-specific fine-tuning is the process of refining a pre-existing language model for a particular task or within a specific domain to enhance its performance and tailor it to the unique context of that domain. This method involves taking an LLM that has undergone pre-training on a diverse dataset encompassing various language use cases and subsequently fine-tuning it on a narrower dataset specifically related to a particular domain or task.

💡 Note that the previous method, i.e., domain-specific pre-training involves training a language model exclusively on data from a specific domain, creating a specialized model for that domain. On the other hand, domain-specific fine-tuning takes a pre-trained general model and further trains it on domain-specific data, adapting it for tasks within that domain without starting from scratch. Pre-training is domain-exclusive from the beginning, while fine-tuning adapts a more versatile model to a specific domain.

The key steps in domain-specific fine-tuning include:

1. **Pre-training:** Initially, a large language model is pre-trained on an extensive dataset, allowing it to grasp general language patterns, grammar, and contextual understanding (A general LLM).
2. **Fine-tuning Dataset:** A more focused dataset, tailored to the desired domain or task, is collected or prepared. This dataset contains relevant examples and instances related to the target domain, potentially including labeled examples for supervised learning.
3. **Fine-tuning Process:** The pre-trained language model undergoes further training on this domain-specific dataset. During fine-tuning, the model's parameters are adjusted based on the new dataset, while retaining the general language understanding acquired during pre-training.
4. **Task Optimization:** The fine-tuned model is optimized for specific tasks within the chosen domain. This optimization may involve adjusting parameters related to the task, such as the model architecture, size, or tokenizer, to achieve optimal performance.

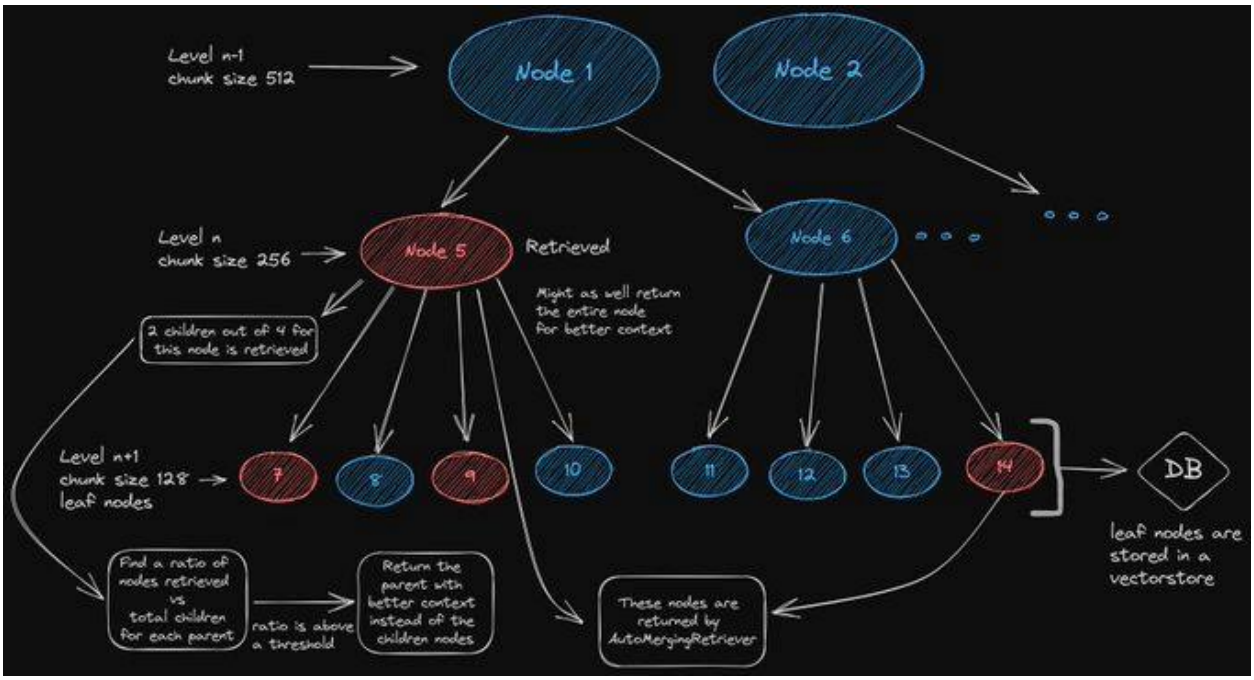
Domain-specific fine-tuning offers several advantages:

- It enables the model to specialize in a particular domain, enhancing its effectiveness for tasks within that domain.
- It saves time and computational resources compared to training a model from scratch, leveraging the knowledge gained during pre-training.
- The model can adapt to the specific requirements and nuances of the target domain, leading to improved performance on domain-specific tasks.

A popular example for domain-specific fine-tuning is the ChatDoctor LLM which is a specialized language model fine-tuned on Meta-AI's large language model meta-AI (LLaMA) using a dataset of 100,000 patient-doctor dialogues from an online medical consultation platform. The model undergoes fine-tuning on real-world patient interactions, significantly improving its understanding of patient needs and providing more accurate medical advice. ChatDoctor uses real-time information from online sources like Wikipedia and curated offline medical databases, enhancing the accuracy of its responses to medical queries. The model's contributions include a methodology for fine-tuning LLMs in the medical field, a publicly shared dataset, and an autonomous ChatDoctor model capable of retrieving updated medical knowledge. Read more about ChatDoctor in the paper here.

Retrieval Augmented Generation (RAG)

Retrieval Augmented Generation (RAG) is an AI framework that enhances the quality of responses generated by LLMs by incorporating up-to-date and contextually relevant information from external sources during the generation process. It addresses the inconsistency and lack of domain-specific knowledge in LLMs, reducing the chances of hallucinations or incorrect responses. RAG involves two phases: retrieval, where relevant information is searched and retrieved, and content generation, where the LLM synthesizes an answer based on the retrieved information and its internal training data. This approach improves accuracy, allows source verification, and reduces the need for continuous model retraining.



RAG_w1.png

Image Source: https://www.deeplearning.ai/short-courses/langchain-for-llm-application-development/

The diagram above outlines the fundamental RAG pipeline, consisting of three key components:

1. **Ingestion:**
   - Documents undergo segmentation into chunks, and embeddings are generated from these chunks, subsequently stored in an index.
   - Chunks are essential for pinpointing the relevant information in response to a given query, resembling a standard retrieval approach.
2. **Retrieval:**

- Leveraging the index of embeddings, the system retrieves the top-k documents when a query is received, based on the similarity of embeddings.

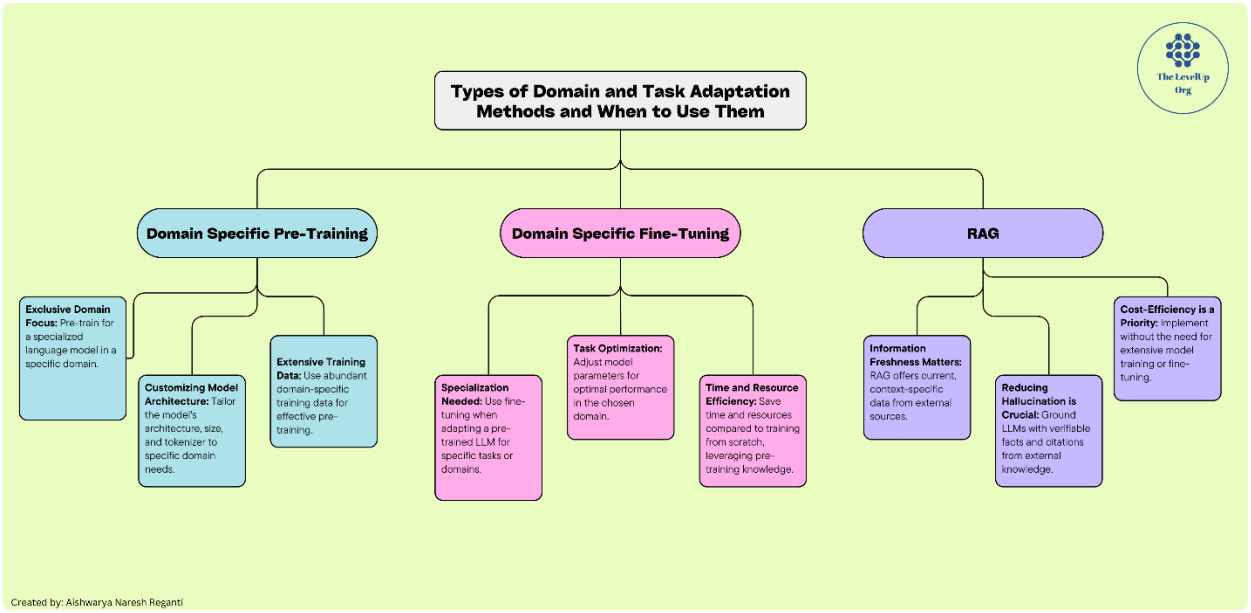
3. **Synthesis:**
   - Examining the chunks as contextual information, the LLM utilizes this knowledge to formulate accurate responses.

💡 Unlike previous methods for domain adaptation, it's important to highlight that RAG doesn't necessitate any model training whatsoever. It can be readily applied without the need for training when specific domain data is provided.

In contrast to earlier approaches for model updates (pre-training and fine-tuning), RAG comes with specific advantages and disadvantages. The decision to employ or refrain from using RAG depends on an evaluation of these factors.

Advantages of RAG	Disadvantages of RAG
Information Freshness: RAG addresses the static nature of LLMs by providing up-to-date or context-specific data from an external database.	Complex Implementation (Multiple moving parts): Implementing RAG may involve creating a vector database, embedding models, search index etc. The performance of RAG depends on the individual performance of all these components
Domain-Specific Knowledge: RAG supplements LLMs with domain-specific knowledge by fetching relevant results from a vector database	Increased Latency: The retrieval step in RAG involves searching through databases, which may introduce latency in generating responses compared to models that don't rely on external sources.
Reduced Hallucination and Citations: RAG reduces the likelihood of hallucinations by grounding LLMs with external, verifiable facts and can also cite sources	
Cost-Efficiency: RAG is a cost-effective solution, avoiding the need for extensive model training or fine-tuning	

Choosing Between RAG, Domain-Specific Fine-Tuning, and Domain-Specific Pre-Training



types_domain_task.png

Use Domain-Specific Pre-Training When:

- **Exclusive Domain Focus:** Pre-training is suitable when you require a model exclusively trained on data from a specific domain, creating a specialized language model for that domain.
- **Customizing Model Architecture:** It allows you to customize various aspects of the model architecture, size, tokenizer, etc., based on the specific requirements of the domain.
- **Extensive Training Data Available:** Effective pre-training often requires a large amount of domain-specific training data to ensure the model captures the intricacies of the chosen domain.

Use Domain-Specific Fine-Tuning When:

- **Specialization Needed:** Fine-tuning is suitable when you already have a pre-trained LLM, and you want to adapt it for specific tasks or within a particular domain.
- **Task Optimization:** It allows you to adjust the model's parameters related to the task, such as architecture, size, or tokenizer, for optimal performance in the chosen domain.
- **Time and Resource Efficiency:** Fine-tuning saves time and computational resources compared to training a model from scratch since it leverages the knowledge gained during the pre-training phase.

Use RAG When:

- **Information Freshness Matters:** RAG provides up-to-date, context-specific data from external sources.
- **Reducing Hallucination is Crucial:** Ground LLMs with verifiable facts and citations from an external knowledge base.
- **Cost-Efficiency is a Priority:** Avoid extensive model training or fine-tuning; implement without the need for training.

Read/Watch These Resources (Optional)

1. https://www.deeplearning.ai/short-courses/langchain-for-llm-application-development/
2. https://www.superannotate.com/blog/llm-fine-tuning#what-is-llm-fine-tuning
3. https://aws.amazon.com/what-is/retrieval-augmented-generation/#:~:text=Retrieval-Augmented Generation (RAG),sources before generating a response.
4. https://www.youtube.com/watch?v=cXPYtkosXG4
5. https://gradientflow.substack.com/p/best-practices-in-retrieval-augmented

Read These Papers (Optional)

1. https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf
2. https://arxiv.org/abs/2202.01110
3. https://arxiv.org/abs/1801.06146