FODS COURSE PROJECT REPORT

On

Retrieval-Augmented Generation (RAG)

BY:

2021B3A71738P Akshit Phophaliya
2021B3A70995P Dhruv Ravi Krishnan
2022A7PS1182P Arnav Dham
2021B4A71700P Akshata Khandelwal

SUBMITTED TO:

Prof. Tejasvi Alladi

Department of Computer Science and Information Systems

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

1 Introduction

Retrieval-Augmented Generation (RAG) is an approach that crosses the gap of information retrieval and text generation in order to provide the correct and pertinent answers. In contrast to using merely what the AI model knows, RAG searches outside for information as it answers queries, making it more accurate and less prone to error. This technique is of utmost use whenever you need timely or precise information. RAG contains two parts: a retriever that determines helpful documents and a generator, typically a model like a transformer, which builds a long answer from such documents. RAG is typically employed in applications such as open questions in customer service and business-related searching. Through employment of information in retrieved documents, RAG unites lengthy knowledge with the capacity of the model to generate coherent and well-reasoned text.

In our project, we tried to develop a RAG system that can function on an average home computer to improve responses based on PDFs. We developed embeddings, ranked them for relevance, and employed the best ones to produce responses. Different models were compared to identify the best ones for various tasks. Since this system had to run on a home computer, we tried to keep computer requirements minimal while making sure responses are fast and of high quality.

2 Methodology

NCERT textbooks, namely Class XII Physics, were used to test various models and queries. Our project had 4 primary steps:

- **Text Chunking:** We split the text into chunks around 751 characters each, utilizing a tool called spaCy, to achieve efficient searching.
- **FAISS Indexing:** Used an efficient method to speed up searches and reduce memory usage while finding the most relevant chunks from the textbook.
- **Fine-tuning Embedding:** Trained MiniLM-L6 with Multiple Negatives Ranking (MNR) loss to improve the understanding of the text.
- **Multi-LLM Interface:** Developed an interface with Streamlit and OpenRouter.

We tested three pipelines with different models and methods. Below is a summary of the models we tested:

- **MPNet (all-mpnet-base-v2):** A powerful model that captures deep meanings and relationships in text, making it great for understanding and comparison tasks. It generates rich text embeddings. It is effective for applications such as similar sentence search and content grouping.
- **MiniLM-L6-v2:** A lightweight and fast model that handles similarity search tasks efficiently with minimal computing power. This makes it a good fit for quick searches and content grouping on resource-limited systems.
- **MiniLM Fine-tuned:** A customized version of MiniLM-L6 trained on NCERT physics content to better recognize domain-specific information. We fine-tuned it using textbook PDFs to ensure the project runs smoothly on an average personal computer while staying focused on the physics domain.

- **FAISS:** Created by Meta (Facebook), FAISS is very efficient at rapidly searching and clustering similar text fragments within numerous entries, even in the billions. It's well-suited for large tasks such as recommendation systems and document retrieval and operates well on both CPUs and GPUs.

2.1 Chunking

Chunking splits large documents into smaller, substantial chunks for ease of handling and retrieval. Within our project, we employed sentence-based and fixed-size chunking. We adopted SpaCy for its efficient boundary detection at sentence level, with which we set our target chunk size to about 751 characters. This facilitates each chunk consisting of whole ideas, improving relevance of information obtained for other purposes.

2.2 Retrieval

The retrieval component identifies the most relevant information chunks in response to user queries, converting both the queries and the chunks into embeddings, which are then compared to determine the best matches.

2.3 Generation

Generation is done by applying a large language model (LLM) to generate a coherent, contextually relevant response based on the retrieved chunks and the query of the user. The question and the retrieved context are structured into a prompt and submitted to the LLM, e.g., those accessed through OpenRouter or Hugging Face. By enriching the LLM with domain expertise or recent data, the system is able to give more accurate and reduced hallucinated answers. Prompt engineering and careful context choice are crucial in optimizing the contribution of this stage.

2.4 FAISS

FAISS (Facebook AI Similarity Search) is an open-source library for fast similarity search and clustering of dense vectors. FAISS is compatible with numerous indexing methods, such as flat, IVF, and PQ, allowing for scalable and efficient nearest neighbor searches even for millions of vectors. FAISS is capable of using both CPU and GPU resources, making it particularly applicable to real-time usage. Within our pipeline, FAISS is used to store and query embeddings of text blocks in order to have quick access to semantically near passages.

2.5 Comparison Metrics for Evaluation

Rigorous evaluation of our RAG system required assessment of both embedding quality and generation performance. We implemented standardized metrics and tested against both a generic dataset and a specific dataset to accurately determine the generalization of system effectiveness before and after fine-tuning the model.

Evaluation Metrics: Embedding

- **Average Cosine Similarity**: Measures semantic similarity between vectors in the embedding space:

$$\cos(a, b) = \frac{a \cdot b}{||a|| \cdot ||b||} \tag{1}$$

  This metric helped us compare differences in semantic clarity of the embedding models before and after fine-tuning.

- **Precision@k**: Calculates the proportion of relevant chunks among the top-k retrieved results:

$$\text{Precision@k} = \frac{\text{number of relevant items in top-k results}}{k} \tag{2}$$

  This helped us evaluate the quality of the chunks created and the ability of the retrieval to pull relevant embeddings and chunks from the context.

- **Recall@k**: Measures the proportion of all relevant chunks successfully retrieved:

$$\text{Recall@k} = \frac{\text{number of relevant items in top-k results}}{\text{total number of relevant items}} \tag{3}$$

  We implemented k=5 as the default setting on the Streamlit app, but the slider allows it to change.

- **Mean Reciprocal Rank (MRR)**: A ranking metric focusing on the position of the first relevant result:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} \tag{4}$$

  where rank_i is the position of the first relevant result for query i. This metric helped us understand how accurately the model was able to pull the most relevant chunk first.

Evaluation Metrics: Generation

- **BLEU Score**: Evaluates generated text quality by measuring n-gram precision against reference texts:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{4} w_n \log p_n\right) \tag{5}$$

  where BP is the brevity penalty, w_n are weights, and p_n is the n-gram precision. Brevity penalty refers to the model penalization method that penalizes machine-translated text if it is too short in comparison to the reference translation. N-gram precision checks the overlap between our generated result and the text provided in the PDF, prioritizing higher overlap with the reference text.

- **ROUGE-L F1 Score**: Measures the Longest Common Subsequence (LCS) between generated and reference texts:

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

where:

$$\text{Precision} = \frac{\text{LCS}(a, b)}{\text{number of unigrams in generated text}} \tag{7}$$

$$\text{Recall} = \frac{\text{LCS}(a, b)}{\text{number of unigrams in reference text}} \tag{8}$$

ROUGE-L helped us evaluate the quality of the translation and document summarization from the reference texts that was being done to generate answers to user queries.

2.6 Model Selection

To create embeddings in the RAG pipeline, we used transformer-based models such as MPNet and MiniLM, given their high performance in language tasks. These models, available through libraries like `sentence-transformers`, convert text into embeddings, which are vector representations that capture semantic relationships between words, phrases, and sentences. MPNet is a more complex and computationally heavy model, while MiniLM provides a lightweight alternative with competitive performance and faster inference, making it more suitable for our use case. Fine-tuning MiniLM on domain-specific data can also enhance its ability to distinguish subtle contextual differences, improving retrieval accuracy for more specific tasks.

2.7 Model Fine-tuning: Supervised or Unsupervised?

In this project, the MiniLM model was fine-tuned using the Multiple Negatives Ranking (MNR) loss. This fine-tuning process is considered a **supervised learning** approach, but it is often described as *weakly supervised* or *self-supervised* in the literature because it relies on positive pairs (such as consecutive sentences or question-answer pairs) rather than explicit class labels **Why is it supervised?** The technique may be mistaken as unsupervised learning due to the absence of explicit ground truth labels. However, although the process does not require explicit class labels, it uses labeled "anchor-positive" pairs generated using heuristic methods by parsing the raw PDF file. These pairs indicate a semantic relationship and make the approach supervised, as the model learns from the provided relationships between pairs, rather than inferring structure from completely unlabeled data.

How does it work? The training data consists of anchor-positive pairs, where the anchor could be a question or a sentence, and the positive is a related sentence or its correct answer. For each anchor-positive pair in a batch, all other examples in the batch are treated as negatives. The model is trained to maximize the similarity between the anchor and its positive, while minimizing the similarity between the anchor and the negatives. This is optimized with the MNR loss function.

2.8 Advantages of Using Multiple Negatives Ranking (MNR) Loss

Unlike traditional loss functions that require explicit negative sampling or labeled negative pairs for a conventional supervised learning technique, MNR leverages all other samples in a training batch as negatives for each anchor-positive pair. This approach

increases training efficiency and diversity of negative examples, leading to more robust and discriminative embeddings.

MNR also enables effective use of weakly labeled data, since it does not require manual annotation of hard negatives. By encouraging the model to maximize similarity between true pairs and minimize similarity with all other in-batch samples, optimizing against MNR Loss enhances both retrieval accuracy and generalization.

Method	**Key Limitation**	**MNR Advantage**
Triplet Loss	Requires explicit negative mining	Uses all in-batch examples as negatives, eliminating manual mining.
Softmax Loss	Needs labeled negative pairs	Treats unlabeled in-batch samples as negatives, leveraging unlabeled data.

Table 1: Why MNR Outperforms Traditional Methods

This approach enables the model to generate more meaningful and discriminative embeddings, which are crucial for high-quality retrieval in RAG pipelines.

Real-time retrieval augmented generation pipeline:

1. User query embedding via fine-tuned MiniLM
2. FAISS similarity search (top-5 chunks)
3. Context injection to selected LLM
4. LaTeX-formatted response generation

3 Results

Evaluation on 291 textbook pages showed the following results (available in Figure 1).

Table 2: Comparison of LLM Metrics: Pre-trained RAG, Fine-tuned RAG, and No RAG

Metric	**Pre-trained RAG**	**Fine-tuned RAG**	**No RAG**
Average Cosine Similarity	0.6552	0.6331	–
Mean Reciprocal Rank	1.0000	1.0000	–
Precision@K	0.5766	0.5110	–
Recall@K	0.7333	0.6667	–
Average BLEU Score	0.0634	0.1033	0.03
Average ROUGE-L F1	0.3344	0.3727	0.24
Average Retrieval Time (s)	0.0109	0.0110	–
Average Response Time (s)	4.6117	5.1640	–

```
metrics = compute_metrics(test_data_general, embeddings, embedding_model, pages_and_chunks, k=3)
for key, value in metrics.items():
    print(f"{key}: {value:.4f}")
```

```
Average Cosine Similarity: 0.6552
Mean Reciprocal Rank: 1.0000
Precision@K: 0.5766
Recall@K: 0.7333
Average BLEU Score: 0.0634
Average ROUGE-L F1: 0.3344
Average Retrieval Time (s): 0.0109
Average Response Time (s): 4.6117
```

(a) Pre-trained Embedding model

```
metrics = compute_metrics(test_data_general, embeddings_fine_big, embedding_model_fine_big, pages_and_chunks, k=3)
for key, value in metrics.items():
    print(f"{key}: {value:.4f}")
```

```
Average Cosine Similarity: 0.6331
Mean Reciprocal Rank: 1.0000
Precision@K: 0.5110
Recall@K: 0.6667
Average BLEU Score: 0.1033
Average ROUGE-L F1: 0.3727
Average Retrieval Time (s): 0.0110
Average Response Time (s): 5.1640
```

(b) Fine-tuned Embedding model

```
[319]: metrics = compute_llm_metrics_without_rag(test_data_general)
       print(f"BLEU: {metrics['Average BLEU Score']:.2f}")
       print(f"ROUGE-L: {metrics['Average ROUGE-L F1']:.2f}")
```

```
BLEU: 0.03
ROUGE-L: 0.24
```

(c) Without RAG pipeline

Figure 1: Comparison of LLM outputs: (a) Pre-trained RAG, (b) Fine-tuned RAG, (c) Without RAG

3.1 Embedding Results

Results show the following inferences:

- Without RAG Pipeline: Low ROUGE and BLEU scores indicate poor model performance in generating the response resembling ground truth in the absence of additional context.
- Pre-trained embedding model: We see significantly improved performance with much higher BLEU and ROUGE scores with the use of a RAG pipeline and pre-trained embedding model.
- Fine-tuned embedding model: Lower precision and recall indicate overfitting due to the use of a skewed and small dataset. This was mainly due to computational and resource constraints. Higher BLEU and ROUGE further validate the overfitting since highly specific vocabulary is being used more frequently.

3.2 Streamlit App Implementation

To simplify the user experience and also visually compare the qualitative performance of different LLM models on the same embedding model (fine-tuned miniLM), a locally-hostable app was designed using Streamlit which allowed us to switch between different models and upload PDFs to view the page numbers corresponding to the top-K chunks and a generated answer using the LLM models. To call various models, we used the OpenRouter API, which allowed us to access various free LLM models, including:

- deepseek/deepseek-r1
- google/gemma-3-27b-it
- meta-llama/llama-3.1-8b-instruct

Screenshots showing the UX of the app and generated responses using different LLMs are also available at the end of the report.

4 Conclusion

- The usage of a RAG pipeline significantly improves embedding quality and answer generation for PDF-based context extraction tasks, showing much higher BLEU and ROUGE scores.
- Using FAISS for retrieval of relevant chunks also significantly improves retrieval time, speeding up the answer generation process.
- The MiniLM model is much lighter and faster than the MPNet model, making it more suitable for projects that require local hosting or running on a single computer.
- The importance of robust datasets in fine-tuning existing models is very crucial to avoid overfitting or underfitting.

References

[1] LLM Evaluation Metrics: The Ultimate LLM Evaluation Guide.
https://www.confident-ai.com/blog/llm-evaluation-metrics-everything-you-need-for-

[2] Define your evaluation metrics — Generative AI on Vertex AI.
https://cloud.google.com/vertex-ai/generative-ai/docs/models/determine-eval

[3] LLM Evaluation: Metrics, Methodologies, Best Practices - DataCamp.
https://www.datacamp.com/blog/llm-evaluation

[4] LLM Evaluation: Key Metrics, Best Practices and Frameworks - Aisera.
https://aisera.com/blog/llm-evaluation/

[5] Simplifying RAG evaluation with Ragas - QED42.
https://www.qed42.com/insights/simplifying-rag-evaluation-with-ragas

[6] Evaluation of RAG Metrics for Question Answering in the Telecom Domain.
https://arxiv.org/html/2407.12873v1

[7] A Comprehensive Guide: How to Create Tables in LaTeX - Scholarly.
https://scholarly.so/blog/a-comprehensive-guide-how-to-create-tables-in-latex

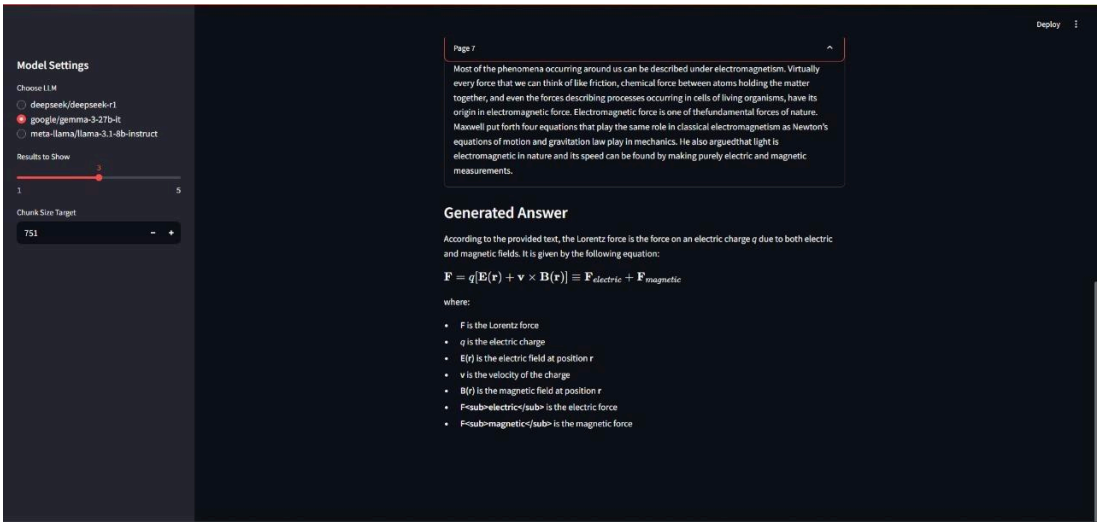
[8] Bibliography - AI and Text Mining for Searching and Screening the Literature.
https://libraryguides.mcgill.ca/text-mining/bibliography

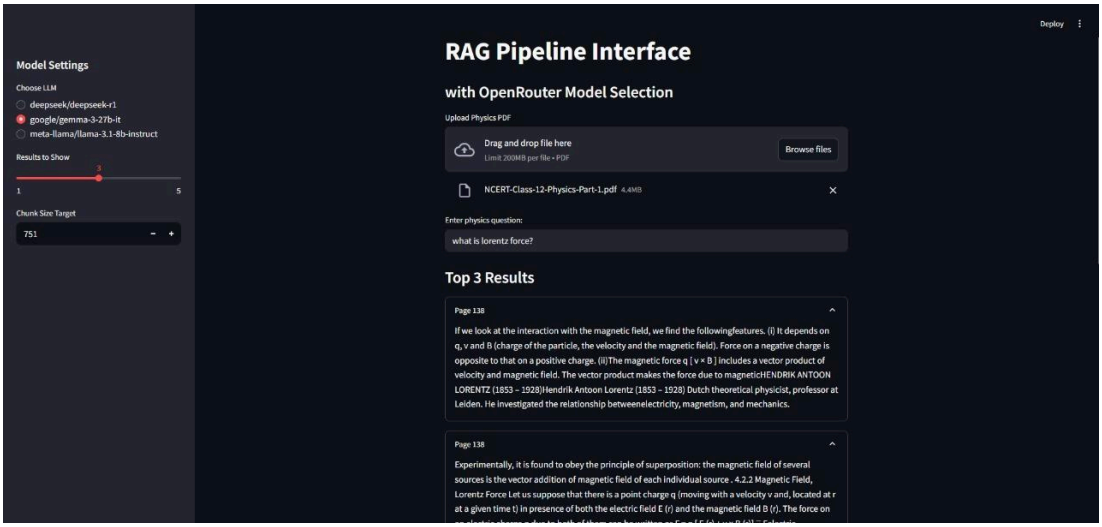
[9] Ragas: Evaluation Framework for Retrieval-Augmented Generation.
https://github.com/explodinggradients/ragas

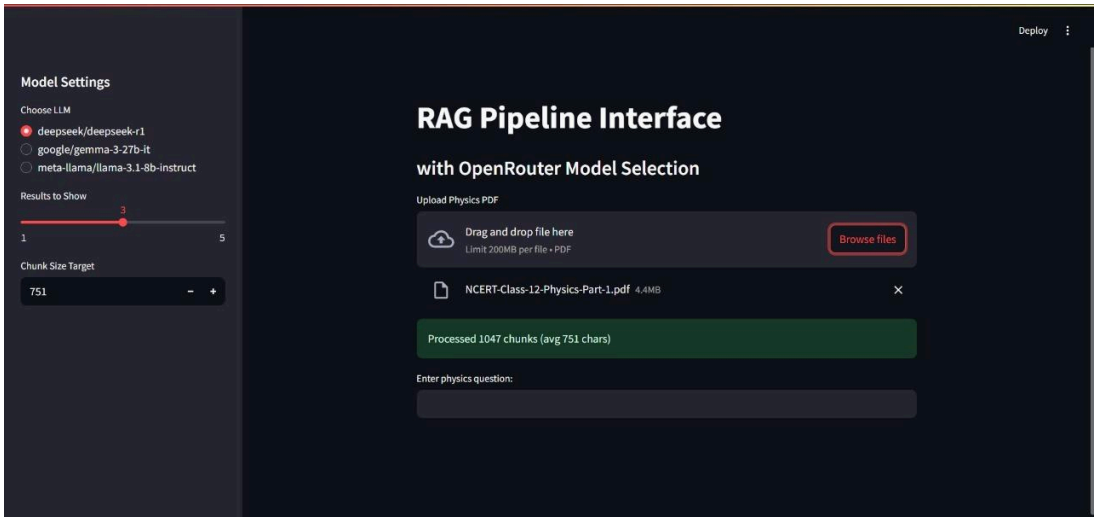
[10] Hugging Face RAG Documentation.
https://huggingface.co/docs/transformers/model_doc/rag

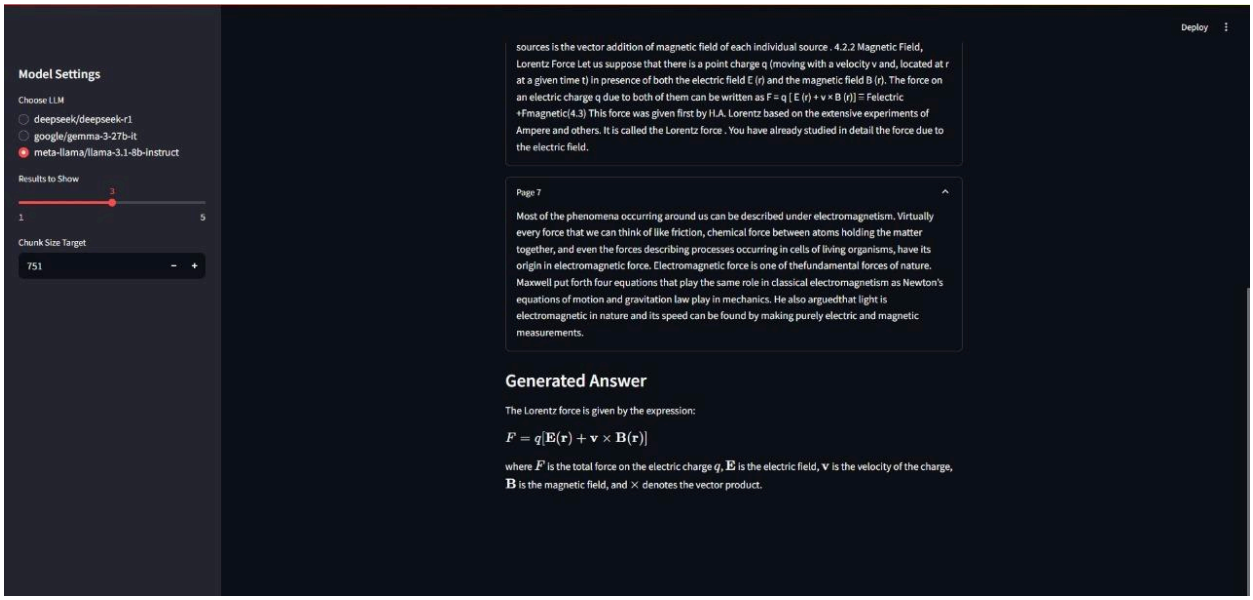
[11] BLEU Score - OpenAI Cookbook.
https://cookbook.openai.com/examples/bleu_score

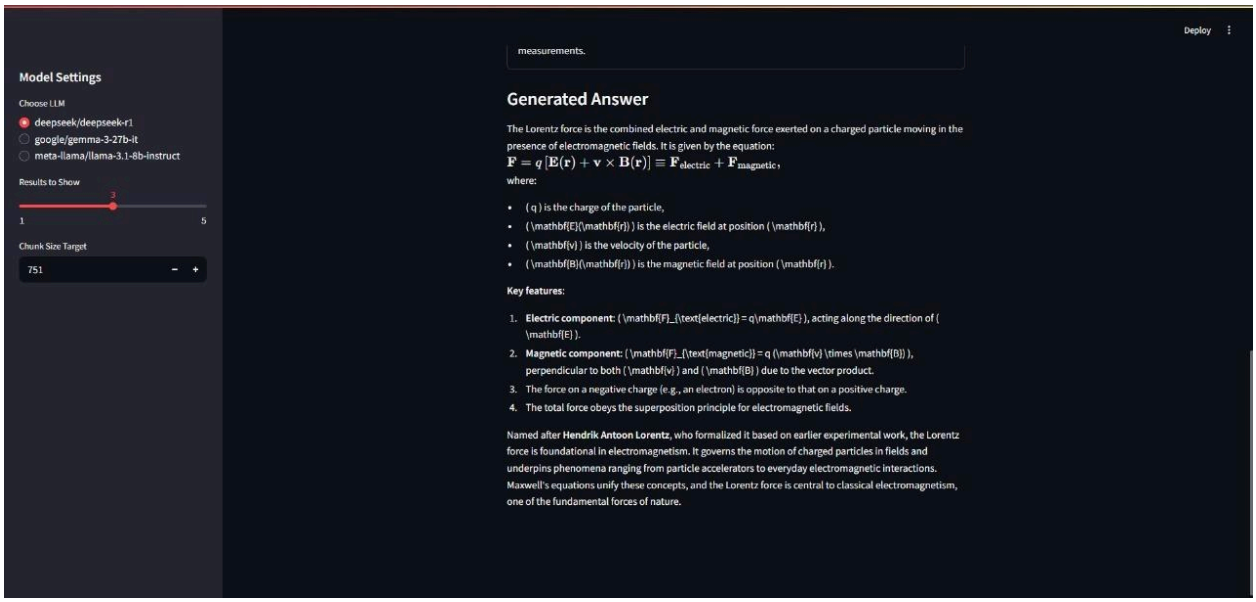
[12] ROUGE: A Package for Automatic Evaluation of Summaries.
https://aclanthology.org/W04-1013/


Deploy
Model Settings
Choose LLM
deepseek/deepseek-r1
google/gemma-3-27b-it
meta-llama/llama-3.1-8b-instruct
Results to Show
3
1
5
Chunk Size Target
751
Page 7
Most of the phenomena occurring around us can be described under electromagnetism. Virtually every force that we can think of like friction, chemical force between atoms holding the matter together, and even the forces describing processes occurring in cells of living organisms, have its origin in electromagnetic force. Electromagnetic force is one of thefundamental forces of nature. Maxwell put forth four equations that play the same role in classical electromagnetism as Newton's equations of motion and gravitation law play in mechanics. He also arguedthat light is electromagnetic in nature and its speed can be found by making purely electric and magnetic measurements.
Generated Answer
According to the provided text, the Lorentz force is the force on an electric charge q due to both electric and magnetic fields. It is given by the following equation:
$\mathbf{F} = q[\mathbf{E}(\mathbf{r}) + \mathbf{v} \times \mathbf{B}(\mathbf{r})] \equiv \mathbf{F}_{electric} + \mathbf{F}_{magnetic}$
where:
F is the Lorentz force
q is the electric charge
E(r) is the electric field at position r
v is the velocity of the charge
B(r) is the magnetic field at position r
F_{electric} is the electric force
F_{magnetic} is the magnetic force


Deploy
Model Settings
Choose LLM
deepseek/deepseek-r1
google/gemma-3-27b-it
meta-llama/llama-3.1-8b-instruct
Results to Show
3
1
5
Chunk Size Target
751
RAG Pipeline Interface
with OpenRouter Model Selection
Upload Physics PDF
Drag and drop file here
Limit 200MB per file • PDF
Browse files
NCERT-Class-12-Physics-Part-1.pdf 4.4MB
Enter physics question:
what is lorentz force?
Top 3 Results
Page 138
If we look at the interaction with the magnetic field, we find the followingfeatures. (i) It depends on q, v and B (charge of the particle, the velocity and the magnetic field). Force on a negative charge is opposite to that on a positive charge. (ii)The magnetic force q [v × B] includes a vector product of velocity and magnetic field. The vector product makes the force due to magneticHENDRIK ANTOON LORENTZ (1853 – 1928)Hendrik Antoon Lorentz (1853 – 1928) Dutch theoretical physicist, professor at Leiden. He investigated the relationship betweenelectricity, magnetism, and mechanics.
Page 138
Experimentally, it is found to obey the principle of superposition: the magnetic field of several sources is the vector addition of magnetic field of each individual source . 4.2.2 Magnetic Field, Lorentz Force Let us suppose that there is a point charge q (moving with a velocity v and, located at r at a given time t) in presence of both the electric field E (r) and the magnetic field B (r). The force on


Deploy
Model Settings
Choose LLM
deepseek/deepseek-r1
google/gemma-3-27b-it
meta-llama/llama-3.1-8b-instruct
Results to Show
3
1
5
Chunk Size Target
751
RAG Pipeline Interface
with OpenRouter Model Selection
Upload Physics PDF
Drag and drop file here
Limit 200MB per file • PDF
Browse files
NCERT-Class-12-Physics-Part-1.pdf 4.4MB
Processed 1047 chunks (avg 751 chars)
Enter physics question:


Deploy
Model Settings
Choose LLM
deepseek/deepseek-r1
google/gemma-3-27b-it
meta-llama/llama-3.1-8b-instruct
Results to Show
3
1
5
Chunk Size Target
751
sources is the vector addition of magnetic field of each individual source . 4.2.2 Magnetic Field, Lorentz Force Let us suppose that there is a point charge q (moving with a velocity v and, located at r at a given time t) in presence of both the electric field E (r) and the magnetic field B (r). The force on an electric charge q due to both of them can be written as F = q [E (r) + v × B (r)] = Felectric +Fmagnetic(4.3) This force was given first by H.A. Lorentz based on the extensive experiments of Ampere and others. It is called the Lorentz force . You have already studied in detail the force due to the electric field.
Page 7
Most of the phenomena occurring around us can be described under electromagnetism. Virtually every force that we can think of like friction, chemical force between atoms holding the matter together, and even the forces describing processes occurring in cells of living organisms, have its origin in electromagnetic force. Electromagnetic force is one of thefundamental forces of nature. Maxwell put forth four equations that play the same role in classical electromagnetism as Newton's equations of motion and gravitation law play in mechanics. He also arguedthat light is electromagnetic in nature and its speed can be found by making purely electric and magnetic measurements.
Generated Answer
The Lorentz force is given by the expression:
$$F = q[\mathbf{E}(\mathbf{r}) + \mathbf{v} \times \mathbf{B}(\mathbf{r})]$$
where F is the total force on the electric charge q, $\mathbf{E}$ is the electric field, $\mathbf{v}$ is the velocity of the charge, $\mathbf{B}$ is the magnetic field, and $\times$ denotes the vector product.


Deploy
Model Settings
Choose LLM
deepseek/deepseek-r1
google/gemma-3-27b-it
meta-llama/llama-3.1-8b-instruct
Results to Show
3
1
5
Chunk Size Target
751
measurements.
Generated Answer
The Lorentz force is the combined electric and magnetic force exerted on a charged particle moving in the presence of electromagnetic fields. It is given by the equation:
$$\mathbf{F} = q\left[\mathbf{E}(\mathbf{r}) + \mathbf{v} \times \mathbf{B}(\mathbf{r})\right] = \mathbf{F}_{\text{electric}} + \mathbf{F}_{\text{magnetic}},$$
where:
(q) is the charge of the particle,
(\mathbf{E}(\mathbf{r})) is the electric field at position (\mathbf{r}),
(\mathbf{v}) is the velocity of the particle,
(\mathbf{B}(\mathbf{r})) is the magnetic field at position (\mathbf{r}).
Key features:
1. Electric component: (\mathbf{F}_{\text{electric}} = q\mathbf{E}), acting along the direction of (\mathbf{E}).
2. Magnetic component: (\mathbf{F}_{\text{magnetic}} = q (\mathbf{v} \times \mathbf{B})), perpendicular to both (\mathbf{v}) and (\mathbf{B}) due to the vector product.
3. The force on a negative charge (e.g., an electron) is opposite to that on a positive charge.
4. The total force obeys the superposition principle for electromagnetic fields.
Named after Hendrik Antoon Lorentz, who formalized it based on earlier experimental work, the Lorentz force is foundational in electromagnetism. It governs the motion of charged particles in fields and underpins phenomena ranging from particle accelerators to everyday electromagnetic interactions. Maxwell's equations unify these concepts, and the Lorentz force is central to classical electromagnetism, one of the fundamental forces of nature.