Project overview

The internship goal is to: (1) implement and finetune a suite of open(-weight) summarization models on the NewsSumm multi-document Indian English dataset; (2) evaluate them with ROUGE-1, ROUGE-2, ROUGE-L and BERTScore; and (3) design and benchmark a novel multi-document abstractive model that outperforms baselines on NewsSumm.

Dataset and Research paper:

- **NewsSumm paper:** https://www.mdpi.com/2073-431X/14/12/508
- **NewsSumm dataset link (Zenodo):** https://zenodo.org/records/17670865

Phase 1 – Dataset understanding and setup

- Read the NewsSumm paper carefully (MDPI Computers 2025) to understand dataset splits, cluster structure (multiple articles per news story), and summary annotation protocol.
- Download the NewsSumm dataset from the Zenodo link given in the paper (train/validation/test files and metadata).
- Implement a consistent loader:
    - Each sample should expose: a cluster ID, list of source articles (texts, titles, publication metadata if available), and the human reference summary.
    - Implement tokenization utilities and a data statistics script (average tokens per cluster, average number of documents per cluster, etc.).

Deliverables:

- `data_pipeline.md` explaining file structure, preprocessing steps, cleaning rules (HTML, boilerplate, etc.), and token length distribution plots.
- A Python module (e.g., `newssumm_dataset.py`) that yields ready-to-feed batches for any model.

Phase 2 – Baseline model implementation (10 models)

For each model below, interns may use PyTorch, Hugging Face Transformers, or any framework (Colab, Kaggle, local GPU, university cluster). The key is consistent preprocessing and evaluation.

2.1 Long-context encoder–decoders

Models:

- PRIMERA (BigBird-based long-context summarizer).
- LED (LongformerEncoderDecoder).
- LongT5 / UL2 summarization variants.

Tasks:

- Load pretrained checkpoints and tokenizers from Hugging Face.
- Implement a multi-document input strategy:
  - Concatenate all documents in a cluster with clear delimiters (e.g., `[DOC]` tokens or special markers).
  - Respect maximum context length; if needed, truncate by date/importance or apply simple sentence-level selection.
- Fine-tune on NewsSumm train set, validate on dev set.

Hyperparameters (to be tuned once and reused with minor variations):

- Batch size, learning rate, max_grad_norm, number of epochs, warmup steps.
- Max input length (e.g., 4096–8192 tokens, model-dependent) and max target length (e.g., 128–256 tokens).

Deliverables:

- Training script `train_primeraled_longt5.py` with clear flags for model name and max length.
- Validation metrics after each epoch.

2.2 Instruction/LLM-style open models

Models:

- Flan-T5 (large/XL/XXL).
- LLaMA-2 (or equivalent open-weight variant with license compliance).
- Mistral 7B / Mixtral.
- Qwen2/Qwen2.5 (7B–14B).

- Gemma/Gemma 2.
- GPT-OSS-style open-weight large model (if compute allows).

Tasks:

- For encoder–decoder LLMs (Flan-T5, some Qwen variants), treat them like Phase 2.1: finetune with standard seq2seq loss.
- For decoder-only LLMs (LLaMA-style, Mistral, Gemma), implement:
    - Prompt template for multi-document summarization, e.g.:
        - "You are a news editor. Write a concise, factual summary of the following Indian news articles…" followed by concatenated documents.
    - Fine-tuning via LoRA/PEFT to reduce compute; supervised fine-tuning on NewsSumm.
- Respect NewsSumm style: factual, multi-article synthesis, no hallucinated content.

Deliverables:

- `train_llm_lora.py` with choices for base model, rank, and alpha.
- Documentation of GPU requirements and approximate training times for each model.

Phase 3 – Evaluation framework (ROUGE & BERTScore)

Metrics:

- ROUGE-1, ROUGE-2, ROUGE-L (F1 scores).
- BERTScore (F1, using multilingual/English BERTScore model).

Tasks:

- Implement a unified evaluation script `evaluate_newssumm.py`:
    - Input: path to model outputs (JSONL/CSV with columns: cluster_id, generated_summary, reference_summary).
    - Compute ROUGE using `rouge-score` or `py-rouge`, with consistent tokenization and stemming.
    - Compute BERTScore with a fixed model (e.g., `microsoft/deberta-xlarge-mnli` or standard BERTScore model), ensuring same IDF and rescale settings for all runs.
- For each model:

- Evaluate on the official test split only once final hyperparameters are decided.
- Record mean scores and 95% confidence intervals using bootstrap or cluster-level variance.

Deliverables:

- A single CSV/Excel file `newssumm_baselines_scores.csv` with columns:
  - `model_name, params(M), context_len, training_type (FT/LoRA/zero-shot), ROUGE1, ROUGE2, ROUGEL, BERTScore`
- A plotting script to generate bar charts or tables for the paper.

Phase 4 – Novel model design and implementation

Goal: Interns must propose and implement one novel model tailored to NewsSumm and compare it against all baselines.

Suggested directions (they can choose one or combine):

- Hierarchical dual-encoder for NewsSumm:
  - Sentence/document-level encoders for each article.
  - Cross-article attention or graph layer linking entities/events across documents.
  - Global decoder conditioned on cluster-level summary plan.
- Salience-aware / energy-aligned model:
  - Latent salience variable capturing which sentences or entities must appear in the final summary.
  - Coverage constraints or energy-based reweighting to reduce redundancy and hallucination.
- Indian-news–aware adapter:
  - Domain adapter for Indian news: special tokens for state, topic, politics, finance, etc.
  - Auxiliary loss for entity coverage and geographic diversity.

Implementation tasks:

- Decide base backbone (e.g., LongT5, PRIMERA, or Mistral 7B) to keep compute reasonable.
- Add one or two additional modules only:
  - Example: graph aggregator over article-level sentence embeddings, or a planning head predicting key bullet points before decoding.

- Train from baseline checkpoint on NewsSumm with extra loss terms:
    - Main cross-entropy loss for summary.
    - Optional auxiliary loss for salience prediction (sentence-level labels from reference summary alignments) or coverage penalty.

Deliverables:

- Model specification document (`novel_model_spec.md`) with:
    - Architecture diagram.
    - Mathematical formulation (encoder, planner, decoder, losses).
    - Justification: why this should improve synthesis and coverage for multi-document Indian news.
- Code (`models/novel_newssumm_model.py`) and training script.

Phase 5 – Benchmark construction and reporting

Tasks:

- Run the full pipeline on the test set for:
    - All 10 baselines.
    - The proposed novel model.
- Build a clear benchmark table for the paper/report:
    - Columns: Model, Type (enc-dec/LLM/graph/hierarchical), Context length, Training regime (zero-shot, few-shot, FT, LoRA), ROUGE-1, ROUGE-2, ROUGE-L, BERTScore.
    - Highlight best and second-best scores for each metric.
- Perform error analysis:
    - Randomly sample 100 clusters and compare outputs of strongest baseline vs novel model.
    - Categorize errors: missing key event, wrong entity, hallucinated facts, redundancy, poor coherence.
- Optional: human evaluation with a small pool of annotators (relevance, coherence, factuality on Likert scales).

Deliverables:

- `newssumm_benchmark.md` containing:
    - Final tables and plots.
    - Short narrative analysis of where each model family excels or fails.
- Ready-to-use section drafts for a paper/thesis:
    - “Experimental Setup”

- “Baselines”
- “Proposed Model”
- “Results and Discussion”

Phase 6 – Engineering, reproducibility, and documentation

Tasks:

- Create a structured Git repository:
  - `data/`, `models/`, `scripts/`, `configs/`, `results/`, `docs/`.
- Provide:
  - `README.md` with step-by-step instructions:
    - How to download NewsSumm.
    - How to preprocess.
    - How to train/evaluate each model.
  - `requirements.txt` or `environment.yml`.
- Ensure every experiment is tracked:
  - Configuration JSON/YAML per run.
  - Random seed, GPU type, runtime.

Deliverables:

- Reproducible scripts such that any future intern can re-run at least one baseline and the novel model and recover similar scores.

Expectations from interns

By the end of the internship, interns should be able to:

- Implement and fine-tune at least 10 open(-weight) summarization models on a large multi-document dataset (NewsSumm).
- Build a robust evaluation pipeline with ROUGE and BERTScore.
- Propose, implement, and benchmark a novel multi-document abstractive model that sets a new baseline for NewsSumm.

Output of Work:

1. International Journal Submission

2. International Conference - USA, UK submission

Benchmark table for NewsSumm Dataset baselines								
Model name	HF Checkpoint	Params (M)	Context len (tokens)	Training type	**ROUGE-1(Score after 50 Epochs)**	**ROUGE-2(Score after 50 Epochs)**	**ROUGE-L (Score after 50 Epochs)**	**BERTScor(Score after 50 Epochs)**
PRIMERA	allenai/PRIMERA	~150	4096–8192	From-scratch				
LongT5-base	google/long-t5-tglobal-base	~248	4096	From-scratch				
LED-base	allenai/led-base-16384	~162	16384	From-scratch				
Flan-T5-XL	google/flan-t5-xl	~3000	512–1024	From-scratch				
Flan-T5-XXL	google/flan-t5-xxl	~11000	512–1024	From-scratch				
Mistral-7B-Instruct	mistralai/Mistral-7B-Instruct-v0.3	~7000	32k (prompted)	From-scratch				
Llama-3-8B-Instruct	meta-llama/Meta-Llama-3-8B-Instruct	~8000	128k (prompted)	From-scratch				
Qwen2-7B-Instruct	Qwen/Qwen2-7B-Instruct	~7000	128k (prompted)	From-scratch				
Gemma-2-9B-Instruct	google/gemma-2-9b-it	~9000	8k (prompted)	From-scratch				
Mixtral-8x7B-Instruct	mistralai/Mixtral-8x7B-Instruct-v0.1	~47000	32k (prompted)	From-scratch				