Comparative Study of Text Summarization Models

This report presents a comprehensive analysis of various text summarization models built using two different paradigms:

1. **RNN-based Models** (Tasks B1 and B2)
2. **Transformer-based Models** (Tasks C1 and C2)

For RNN models, we experimented with:

- A basic Seq2seq RNN model
- RNN enhanced with GloVe embeddings
- RNN with a Hierarchical Encoder
- RNN with a 2-layer Decoder
- RNN with all improvements combined (GloVe + Hierarchical Encoder + 2-layer Decoder + Beam Search)

For Transformer models, we compared:

- A fine-tuned pre-trained T5 model
- Zero-shot prompting with FLAN-T5 models (base and large) using different prompt variants

Now, we detail the dataset preparation, model configurations, and experimental results (including execution times and ROUGE score comparisons), followed by visualisations and insights.

1. Task A:Dataset and Preprocessing

- **Cleaning**: Lowercase conversion, removal of non-ASCII characters and punctuation using regular expressions.
- **Tokenization & Filtering**: The text is tokenized, stopwords are removed using an NLTK list, and tokens are stemmed using the PorterStemmer.
- **Validation Extraction**: 500 articles are randomly selected from the training set for validation.

Execution Time:

Stage	Train Data(s)	Validation Data(s)	Test Data(s)
Time(s)	360.06	13.60	2.85

Final data sizes: Train set: 13379,

Validation set: 500,

Test set: 100

2. Task B1: Basic RNN Seq2seq Model

Observed Results (Basic RNN Model):

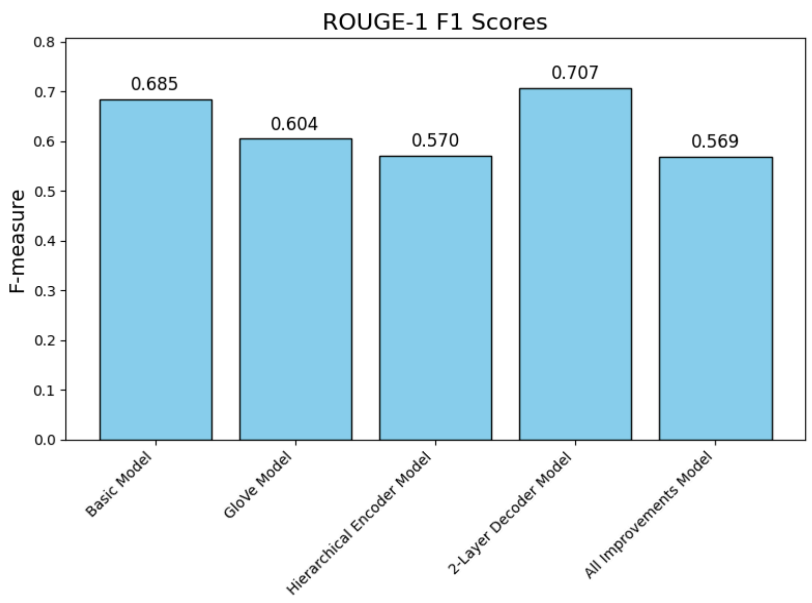
Metric	Trian Loss	Validation Loss	ROUGE-1	ROUGE-2	ROUGE-L
Value	1.5017	2.176	0.7037	0.3704	0.7037

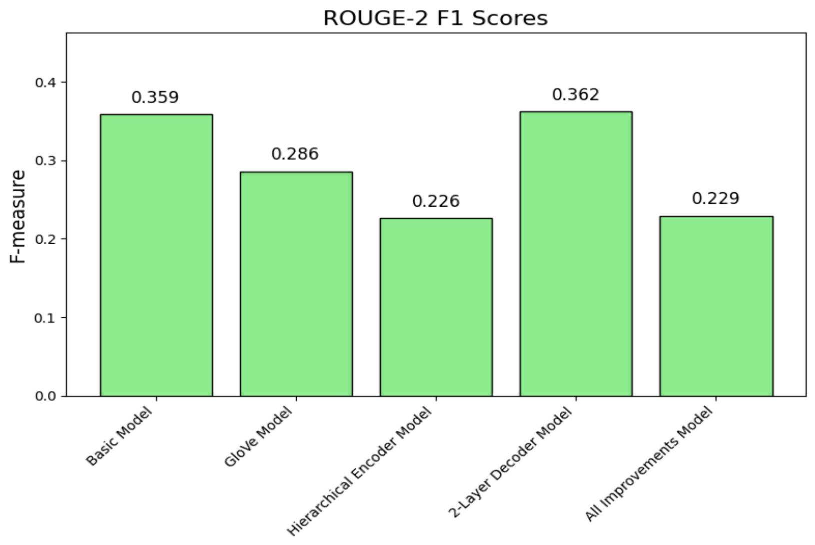
3. Task B2: Improvements to RNN Model

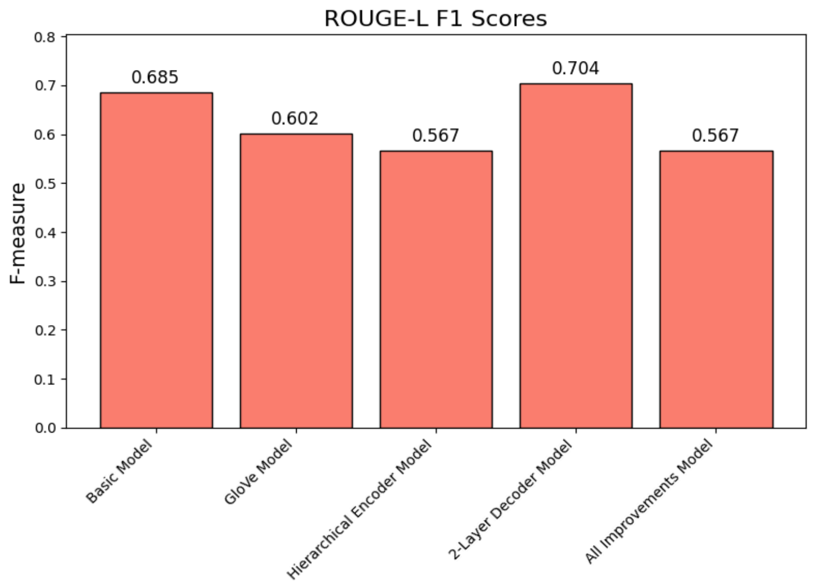
1. **GloVe Embeddings**:Load pretrained GloVe embeddings into the EncoderRNN using a setter function (load_glove_embeddings).
2. **Hierarchical Encoder**:Implement a HierEncoderRNN class and include a loadable embedding function.
3. **2-Layer Decoder**:Create a Decoder2RNN class for deeper decoding.
4. **Beam Search**:Modify Seq2seqRNN to support beam search decoding via a parameter.

Model Variants and Their Results(with visualizations):

Model Variant	Train Time (s)	Eval Time (s)	ROUGE-1	ROUGE-2	ROUGE-L
Basic RNN Model	1962.35	1.24	0.7037	0.3704	0.7037
RNN + GloVe Embeddings	1928.48	1.25	0.6429	0.3091	0.6429
RNN + Hierarchical Encoder	3539.23	1.57	0.5697	0.2260	0.5672
RNN + 2-Layer Decoder	1992.54	1.23	0.7066	0.3622	0.7037
RNN + All Improvements Combined	3553.52	2.11	0.7690	0.5291	0.7665





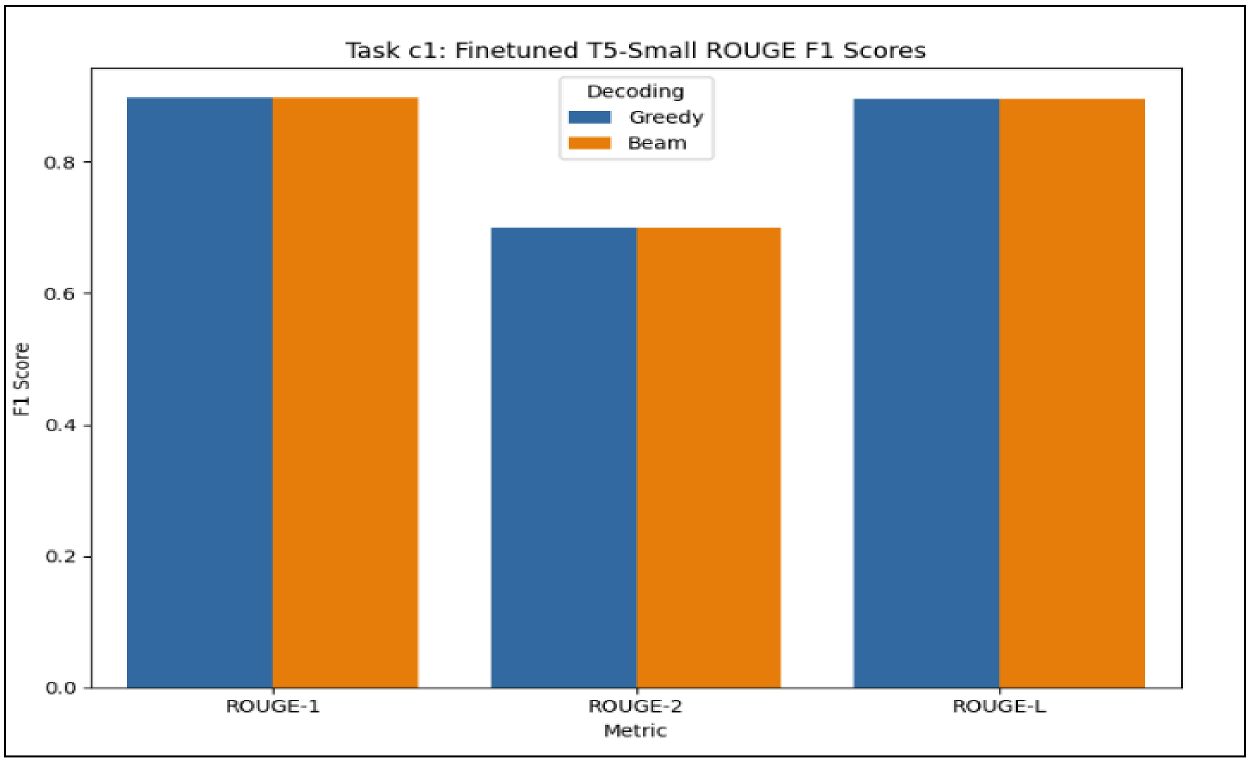


4. Task C1: Transformer Model Fine-Tuning

- **Model**: Load the pre-trained google/t5-small model using AutoModelForSeq2SeqLM.
- **Training**: Fine-tune the model on raw text (with punctuation, stopwords, etc.) using the Seq2SeqTrainer with various hyperparameters.
- **Evaluation**: Generate predictions on the test set with both greedy and beam search decoding, and calculate ROUGE metrics.
- Time taken for training : **4777.69 seconds**

Observed Results (T5 Fine-Tuned)

Metric	**Greedy**	**Beam Search**
ROUGE-1	0.8971	0.8971
ROUGE-2	0.6991	0.6991
ROUGE-L	0.8959	0.8959



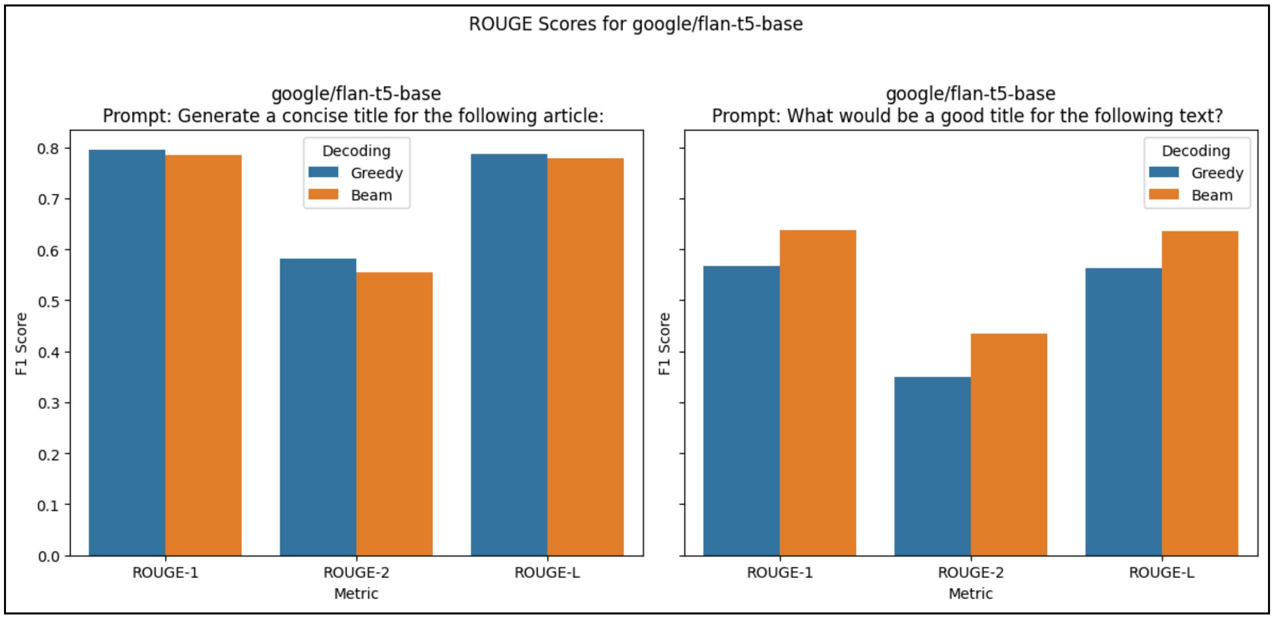
5. Task C2: Zero-shot Prompting with FLAN-T5 Model

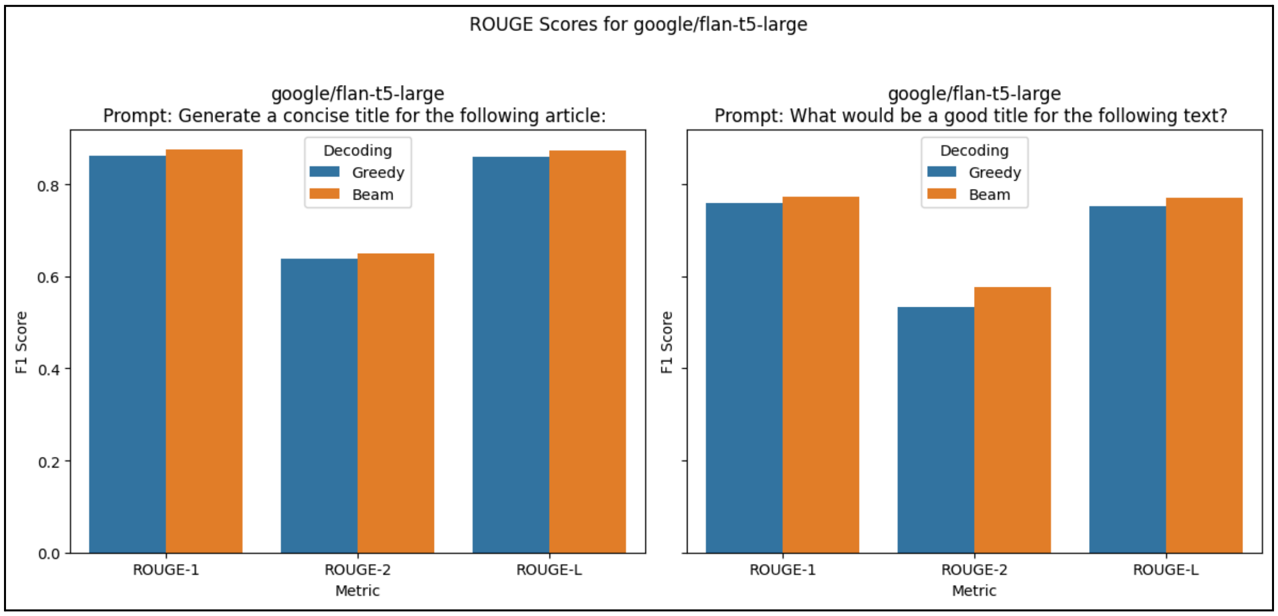
- **Models**: Load pre-trained google/flan-t5-base and google/flan-t5-large using AutoModelForSeq2SeqLM.
- **Prompt Engineering**: Using 2 prompt variants for each model to generate article titles.
- **Evaluation**: Evaluate generated outputs using ROUGE metrics.

Observed Results (FLAN-T5 Zero-shot Prompting)

Variant & Prompt	Strategy	ROUGE-1	ROUGE-2	ROUGE-L
FLAN-T5-Base - Prompt 1	Greedy	0.794507	0.580889	0.786905
FLAN-T5-Base - Prompt 2	Beam	0.784528	0.554000	0.778774
FLAN-T5-Base - Prompt 1	Greedy	0.567114	0.348699	0.563604
FLAN-T5-Base - Prompt 2	Beam	0.637457	0.434251	0.635230
FLAN-T5-Large - Prompt 1	Greedy	0.861026	0.638000	0.858835
FLAN-T5-Large - Prompt 2	Beam	0.875131	0.651000	0.874024
FLAN-T5-Large - Prompt 1	Greedy	0.760484	0.534255	0.751751
FLAN-T5-Large - Prompt 2	Beam	0.772368	0.577688	0.769866

Visualizations:





7. Insights:

Performance Observations

- **Basic RNN Model**: Serves as the baseline. Its performance is limited by its inability to capture long-term dependencies.ROUGE-1 at 0.7037, serving as the standard.
- **RNN Models with Improvements**:
    - **GloVe Embeddings**: Enhanced lexical knowledge slightly improves performance.Slight drop in ROUGE-1 (0.6429) suggests that embedding alignment is critical.
    - **Hierarchical Encoder**: While intended to better capture long sequences, its performance might be lower if chunking does not align with sentence boundaries.Noticeable reduction to ROUGE-1 of 0.5697 indicates potential issues with sentence boundary misalignment.
    - **2-Layer Decoder**: Deepens decoding capacity and improves quality.ROUGE-1 increases marginally to 0.7066, demonstrating that extra depth can yield minor gains while maintaining similar evaluation time (~1.23 s).
    - **All Improvements Combined**: Demonstrates a synergy of all enhancements and achieves the highest ROUGE scores among RNN-based approaches.

- **Transformer Models**:
    - **T5 Fine-tuned**: Achieves high ROUGE scores due to robust pre-trained attention mechanisms.High performance with ROUGE-1 at 0.8971, proving the strength of pre-trained attention mechanisms despite a longer training time (~4777 s).
    - **FLAN-T5 Zero-shot Prompting**: Offers competitive performance without task-specific training and demonstrates the power of prompt engineering.Performance varies by prompt; for instance, FLAN-T5-Large can achieve ROUGE-1 up to 0.8751, highlighting the effectiveness of prompt engineering even without task-specific fine-tuning.

Time Efficiency

- **RNN Models**: Training time increases with added improvements; however, the improvements can justify the additional computation.
- **Transformer Models**: Although Transformers are computationally intensive, their pre-trained nature and efficient fine-tuning often result in faster convergence and superior performance.

Design Decisions and Novelty

- **Preprocessing**: The choice to remove noise (punctuations, non-ASCII, stopwords, stemming) for RNN models helped streamline training.Aggressive cleaning aids RNNs (lower noise, better focus), while raw text benefits Transformers by preserving contextual cues.
- **Model Architecture**: Experimenting with hierarchical encoding and deeper decoders shows the incremental benefits of model complexity.
- **Embedding Integration**: Using external embeddings (GloVe) enabled the model to leverage prior linguistic knowledge.
- **Transformer Comparison**: Including both fine-tuning and zero-shot prompting on Transformer models provides a comprehensive understanding of their performance trade-offs.

Conclusion

- The best performance overall was achieved by the Transformer fine-tuning, which benefits from state-of-the-art pre-trained attention mechanisms.
- Among RNN-based models, the combination of all improvements (including beam search for decoding) produced the highest ROUGE scores.
- Improved architectures (like the 2-layer decoder) incur minimal additional evaluation cost but with modest performance gains. Conversely, hierarchical methods may require refined implementation to properly capture context.
- The results emphasize that while Transformer models are more computationally demanding, they are superior in generating high-quality summaries. However, with careful engineering (e.g., GloVe embeddings, hierarchical encoding), RNN-based models can remain competitive and may be preferred in resource-constrained environments.