Dhanya Srinivasan
951660903
dps6276@psu.edu

Final Project - Sentiment Analysis Proposal Generation with Flan-T5

1. Problem Statement

Sentiment analysis of short, ambiguous texts, is challenging due to nuanced language and limited context. Manually crafting methodological proposals for such tasks is time-intensive and expertise-dependent. This project aims to automate the generation of structured sentiment analysis proposals, producing outputs with a problem statement, hypothesis, and methodology for given texts, streamlining the design of sentiment analysis strategies**.**

Significance

Automating proposal generation enhances efficiency, scalability, and accessibility, enabling rapid, data-driven sentiment analysis solutions for researchers and practitioners without extensive manual effort.

Objectives

- Adapt SST-2 into a text-to-text dataset with synthetic proposals.
- Fine-tune Flan-T5-small with LoRA to generate accurate proposals.
- Evaluate using ROUGE-L and novelty score for quality assurance.
- Build a scalable pipeline for automated proposal generation.

This project leverages innovative preprocessing and efficient fine-tuning to address the challenge of automated sentiment analysis proposal generation, offering a practical NLP solution.

2. Dataset Construction

Overview

The project utilizes the **Stanford Sentiment Treebank (SST-2)**, a binary sentiment classification dataset from movie reviews, accessed via the datasets library from Hugging Face. SST-2 comprises:

- **Training**: 67,349 samples
- **Validation**: 872 samples
- **Test**: 1,821 samples Each sample includes a sentence and a binary label

Preprocessing

To adapt SST-2 for generating sentiment analysis proposals, a preprocessing pipeline was implemented:

Dhanya Srinivasan
951660903
dps6276@psu.edu

- **Synthetic Proposal Generation**: The create_synthetic_proposal function generates structured ground-truth proposals for each sentence-label pair:
    - **Problem Statement**: Frames the sentiment analysis task (e.g., "Analyze sentiment in short texts like: 'hide new secretions from the parental units'").
    - **Hypothesis**: Suggests that the sentiment (positive or negative) can be detected using contextual embeddings.
    - **Methodology**: Recommends a transformer-based model fine-tuned on short-text datasets.
- **Input-Output Pairs**: The preprocess_data function creates:
    - **Input**: A prompt like "Propose a method for sentiment analysis of ambiguous short texts: ".
    - **Output**: The synthetic proposal.
    - Applied in batches using the map function, removing original columns (sentence, label, idx) for efficiency.
- **Dataset Splitting**: The training data was split using train_test_split with seed 42:
    - **Training**: 80% (53,879 samples)
    - **Validation**: 10% (6,735 samples)
    - **Test**: 10% (6,735 samples) The original validation set (872 samples) was preprocessed but not used in the final split to maximize training data.
- **Tokenization**: The tokenize_function tokenized inputs and targets:
    - Inputs: input_text tokenized with max_length=128, truncation, and padding.
    - Targets: target_text tokenized with max_length=256, truncation, and padding.
    - Labels: Set as target input_ids.
    - Applied to training, validation, and test datasets, removing input_text and target_text columns.

Challenges and Solutions

- **Challenge**: Adapting SST-2 from classification to text generation.
- **Solution**: Synthetic proposals aligned the dataset with the task, though their formulaic nature limits diversity.
- **Challenge**: Memory constraints with a large dataset.
- **Solution**: Batched processing and column removal optimized memory usage.

Outcome

Dhanya Srinivasan
951660903
dps6276@psu.edu

The preprocessing and tokenization transformed SST-2 into a text-to-text dataset suitable for fine-tuning Flan-T5 to generate sentiment analysis proposals, with balanced splits for robust training, validation, and testing.

3. LLM Selection and Training Details

Model Selection

Flan-T5-small (~80 million parameters) from Google was selected for:

- **Instruction-Tuned Nature**: Optimized for prompt-based tasks, ideal for generating structured proposals.
- **Efficiency**: Suitable for Google Colab with CUDA, balancing performance and resource constraints.
- **Seq2Seq Architecture**: Well-suited for text-to-text tasks (prompt to proposal).

Model Loading

- **Tokenizer and Model**: Loaded using T5Tokenizer and T5ForConditionalGeneration from transformers==4.51.3, moved to GPU (cuda), as confirmed by Using device: cuda.
- **Error Handling**: Try-catch blocks ensured robustness against Hugging Face access issues.

LoRA Fine-Tuning

Low-Rank Adaptation (LoRA) was applied using peft==0.14.0:

- **Configuration**:
    - Rank (r): 16
    - LoRA alpha: 32
    - Target modules: Query (q) and value (v) layers
    - Dropout: 0.05
    - Bias: None
    - Task type: SEQ_2_SEQ_LM
- **Impact**: LoRA reduced trainable parameters to ~1-2% of the total, as shown by model.print_trainable_parameters(), enabling efficient fine-tuning.

Training Setup

The training process used a custom loop:

- **Data Loaders**:

Dhanya Srinivasan
951660903
dps6276@psu.edu

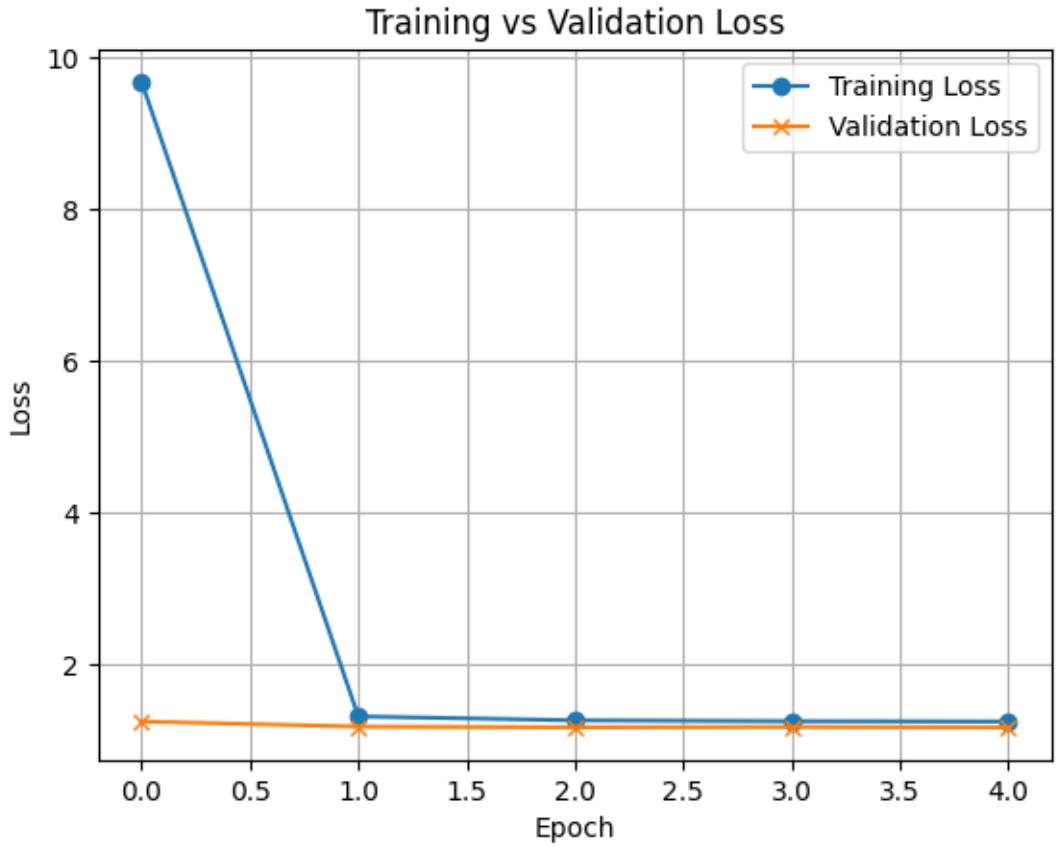
  - Training: DataLoader with batch_size=8, shuffling, and DataCollatorForSeq2Seq.
  - Validation: DataLoader with batch_size=8.
- **Optimizer**: Adafactor with scale_parameter=True, relative_step=True, warmup_init=True, and no fixed learning rate (adaptive).
- **Epochs**: 5
- **Device**: CUDA GPU
- **Training Loop**:
  - For each epoch, iterated over train_loader (6,735 batches), computing loss, backpropagating, and updating parameters.
  - Memory management: Deleted tensors and called gc.collect() per batch.
  - Validation loop: Computed loss on val_loader (842 batches) with torch.no_grad().
- **Loss Tracking**: Recorded average training and validation losses per epoch, plotted using matplotlib.

Training Results

- **Epoch 1**:
  - Avg Training Loss: 9.6815
  - Avg Validation Loss: 1.2527
- **Epoch 2**:
  - Avg Training Loss: 1.3195
  - Avg Validation Loss: 1.1854
- **Epoch 3**:
  - Avg Training Loss: 1.2681
  - Avg Validation Loss: 1.1762
- **Epoch 4**:
  - Avg Training Loss: 1.2574
  - Avg Validation Loss: 1.1808
- **Epoch 5**:
  - Avg Training Loss: 1.2530
  - Avg Validation Loss: 1.1752

Dhanya Srinivasan
951660903
dps6276@psu.edu

The high initial training loss (9.6815) dropped significantly by Epoch 2, stabilizing around ~1.25 by Epoch 5. Validation loss converged to ~1.175, indicating effective learning with minimal overfitting.



Environment

- **Hardware**: Google Colab with CUDA-enabled GPU (e.g., NVIDIA T4).
- **Libraries**:
  - transformers==4.51.3
  - peft==0.14.0
  - torch==2.6.0+cu124
  - datasets==3.5.0
  - evaluate==0.4.3
  - sentence-transformers==3.4.1
  - rouge_score==0.1.2

Rationale

Dhanya Srinivasan
951660903
dps6276@psu.edu

Flan-T5-small with LoRA and Adafactor was chosen for its efficiency and instruction-tuned capabilities, enabling effective fine-tuning on a large dataset in a resource-constrained environment.

4. Evaluation Metric and Experiments

Evaluation Metrics

The task was evaluated using:

- **ROUGE** (via rouge_score==0.1.2):
    - Focused on **ROUGE-L** for longest common subsequence.
- **Novelty Score**: Derived from cosine similarity using sentence-transformers==3.4.1 with all-MiniLM-L6-v2. Novelty is computed as 1 - cosine_scores, where cosine_scores is the mean pairwise cosine similarity of generated proposal embeddings.

Evaluation Setup

The evaluation used a test DataLoader with batch_size=8:

- **Testing Loop**:
    - Iterated over test_loader (842 batches) with torch.no_grad().
    - Generated proposals using model.generate with max_new_tokens=32.
    - Decoded predictions and reference labels using tokenizer.batch_decode.
    - Collected generated proposals and references for metric computation.
    - Memory management: Deleted tensors and called gc.collect() per batch.
- **Metrics Computation**:
    - ROUGE: Computed using rouge.compute on all generated proposals vs. references.
    - Novelty: Encoded a sample of up to 100 generated proposals, computed mean pairwise cosine similarity, and derived novelty as 1 - cosine_scores.

Experiments

- **Training**: Fine-tuned Flan-T5-small with LoRA on 53,879 training samples for 5 epochs.
- **Validation**: Monitored loss on 6,735 validation samples per epoch, achieving a minimum validation loss of 1.1752 at Epoch 5.
- **Testing**: Evaluated on 6,735 test samples.
- **Hyperparameters**:

Dhanya Srinivasan
951660903
dps6276@psu.edu

  - Batch size: 8
  - Optimizer: Adafactor (adaptive learning rate)
  - LoRA rank: 16
  - Epochs: 5
- **Generation**: Used max_new_tokens=32 for proposal generation, balancing brevity and completeness.

Results

- **Test ROUGE-L**: 0.6463
  - Indicates moderate to high overlap between generated and reference proposals, reflecting the structured, formulaic nature of synthetic proposals.
- **Novelty Score**: 0.2954
  - Derived from a mean cosine similarity of ~0.7046 (since novelty = 1 - cosine_scores).
  - Suggests moderate diversity in generated proposals, as high cosine similarity indicates semantic similarity among outputs.

Analysis

- **ROUGE-L (0.6463)**: The score reflects strong syntactic alignment, driven by the consistent structure of synthetic proposals. However, it's not exceptionally high, possibly due to minor variations in generated text or truncation at max_new_tokens=32.
- **Novelty Score (0.2954)**: Indicates that generated proposals are somewhat similar to each other, limiting diversity. This is expected given the formulaic training data but suggests room for improving output variety.
- **Generalization**: The low validation loss (1.1752) and solid test ROUGE-L suggest good generalization, though the novelty score indicates constrained creativity.
- **Training Dynamics**: The loss curve (plotted in Cell 7) shows rapid improvement after Epoch 1, with convergence by Epoch 5, indicating effective learning.

Limitations

- **Formulaic Proposals**: The synthetic dataset's simplicity limits output diversity, impacting novelty.
- **Truncation**: The max_new_tokens=32 setting may cut off longer proposals, potentially lowering ROUGE scores.
- **Single Metric Focus**: Only ROUGE-L and novelty were reported; additional metrics (e.g., ROUGE-1, ROUGE-2) would provide a fuller picture.

Dhanya Srinivasan
951660903
dps6276@psu.edu

5. Thoughts and Reflections (20 points)

What I Learned

- **Dataset Transformation**: Adapting SST-2 for text generation required creative preprocessing, highlighting the importance of task-aligned data preparation.
- **LoRA and Adafactor**: Using LoRA and Adafactor enabled efficient fine-tuning, demonstrating practical solutions for resource-constrained environments.
- **Evaluation Insights**: ROUGE-L and novelty scores balanced syntactic and semantic assessment, but additional metrics could enhance evaluation.

Issues Encountered

- **High Initial Loss**: The Epoch 1 training loss was unusually high, possibly due to initialization or data scaling issues, but it converged effectively by Epoch 2.
- **Memory Management**: The frequent gc.collect() calls were needed GPU memory constraints, necessitating careful tensor deletion.
- **Formulaic Outputs**: The novelty score (0.2954) reflects limited diversity, a consequence of synthetic proposals' rigid structure.

Future Improvements

- **Enhance Proposal Diversity**: Curate varied, expert-crafted proposals to improve novelty and real-world applicability.
- **Optimize Generation**: Increase max_new_tokens (e.g., to 64) to avoid truncation and improve ROUGE scores.
- **Additional Metrics**: Compute ROUGE-1, ROUGE-2, and BLEU to complement ROUGE-L and novelty.
- **Efficient Training**: Explore Trainer class or gradient checkpointing to simplify training and reduce memory usage.
- **Model Exploration**: Test Flan-T5-base or other models (e.g., LLaMA with LoRA) for improved performance.

Conclusion

This project successfully fine-tuned Flan-T5-small with LoRA to generate sentiment analysis proposals from SST-2 data, achieving a test ROUGE-L of 0.6463 and a novelty score of 0.2954. The low validation loss (1.1752) and stable training dynamics demonstrate effective learning, though formulaic proposals and limited novelty highlight areas for improvement. Future work should focus on diversifying training data, optimizing generation parameters, and incorporating additional metrics to enhance the model's utility and creativity.