Extracting Key Insights from Financial Text: A Summarization Approach Project Update II

Parisha Desai
Computer Science Department
Purdue University Fort Wayne
2101E Coliseum Blvd,
Fort Wayne, IN 46805
desap01@pfw.edu

Manvitha Chowdari Gottipati
Computer Science Department
Purdue University Fort Wayne
2101E Coliseum Blvd,
Fort Wayne, IN 46805
gottm01@pfw.edu

1 Introduction

In today's financial world, extracting insights from reports, news, and disclosures is critical. Investors and agencies rely on this data to assess risks and maximize returns. However, manual processing of large datasets is time-consuming and error-prone. NLP and text summarization techniques offer concise summaries that highlight key points efficiently.

This project focuses on summarizing financial reports while retaining essential details. This report outlines the progress made since Project 1, detailing implementation updates and the next steps leading to the final submission.

2 Project Overview

This project focuses on financial narrative summarization using a dataset sourced from Hugging Face, containing training, validation, and test files. Each file comprises annual reports and corresponding gold summaries, with records featuring average word lengths ranging from 39,980 to 41,065 for reports and 1,016 to 1,124 for summaries.

To preserve the dataset's integrity, text preprocessing we initially avoided aggressive normalization techniques like removing numbers, stop words, or short words. Instead, white spaces and unnecessary trailing spaces were removed to retain essential financial details.

We employed two summarization methodologies: Extractive Summarization, which directly selects key sentences, using models like TextRank, and Abstractive Summarization, which generates summaries using NLP techniques, leveraging transformer models like T5 and BART.

The models were implemented as follows:

- **TextRank:** An unsupervised graph-based ranking algorithm generating summaries by ranking top sentences. It achieved initial results using validation and test data, with performance measured via Rouge and BLEU scores.
- **T5 and BART:** Pretrained transformer models fine-tuned using cross-entropy loss and Adam optimizer. Both models were evaluated on validation and test datasets, with hyperparameter adjustments (e.g., token lengths, beam search, and length penalties) to improve summarization quality and computational efficiency.

Execution time and hyperparameter analysis highlighted key trade-offs. TextRank was the fastest at 2 hours due to its simplicity, while T5 required the longest (3 hours) due to extensive tuning. BART balanced efficiency and performance with a 1.5-hour runtime.

These insights guide the project's next steps, focusing on refining preprocessing, optimizing hyperparameters, and exploring additional summarization techniques to enhance performance further.

3 Our Approach and Implementation Methodology

In Project Update 1, we proposed several enhancements to improve our summarization models. Building on that foundation, Update 2 focuses on the following strategies:

1. **Enhanced Preprocessing and Text Normalization**:

- **Planned:** We aimed to refine our preprocessing pipeline by removing special characters (e.g., @, !, ?) that add little value, reducing noise, and enhancing focus on meaningful content. Additionally, we intended to incorporate lemmatization and stop word removal, which had been excluded previously to align with gold summaries. These steps normalize words to their root forms and remove redundant ones, potentially improving model accuracy.

- **Update:** We introduced preprocessing techniques to remove special characters, implemented lemmatization and stop word removal, and evaluated all three models with these enhancements.

2. **TextRank Model Optimization**:

- **Planned:** To enhance the TextRank model, we proposed increasing the number of top sentences used for summarization. Expanding from 3 to a larger number was expected to improve the balance between capturing key details and maintaining conciseness, particularly for longer texts.

- **Update:** We experimented with increasing the top sentence count from 3 to 5 and captured performance results.

3. **T5 and BART Model Refinement** :

- **Planned:** We aimed to fine-tune hyperparameters, such as increasing token length during encoding to accommodate the average word length of gold summaries being 1000 and reducing penalties for generating longer summaries. These changes were intended to improve the models' performance and ability to generate more accurate and insightful summaries.

- **Update:** We reduced the penalties for generating longer summaries (from 2.0 to 1.5) and increased the min-length parameter for encoding from 30 to 200 in both BART and T5 models

4. **Handling Missing Data** :

- **Planned:** While missing data was minimal (1–2 records) in the initial dataset, we planned to remove all rows with missing values and recompute model performance to ensure consistency and reliability in evaluation metrics.

- **Update:** A function to handle missing values was introduced removing all the rows where either the annual report or gold summaries was missing and applied consistently across all three datasets (train, test, and validation) in all model implementations.

5. **Automated Hyperparameter Search** :

- **Planned:** We proposed implementing automated hyperparameter tuning techniques, such as grid search or random search, to optimize configurations, particularly for advanced transformers like T5 and BART.

- **Update:** This task was not completed during this phase and will be prioritized in a future update.

6. **Evaluation of Alternative Summarization Techniques** :

- **Planned:** We intended to explore alternative approaches, including hybrid methods that combine extractive and abstractive techniques. These models would then be compared with our improved implementations of TextRank, T5, and BART to identify better strategies for summarizing complex financial data.

- **Update:** We implemented the Pegasus model, known for its effectiveness in abstractive test summarization, and captured initial performance results.

Implementation of Pegasus Model The Pegasus model, a transformer-based architecture designed for abstractive summarization, was used to generate summaries for the dataset. The pre-trained google/pegasus-large model was loaded, and its tokenizer was employed to prepare the input data. Summarization involved tokenizing the input text into token IDs, processing them through the model, and decoding the generated tokens back into human-readable summaries.

Training was conducted over 3 epochs using the training dataset. The AdamW optimizer was employed to update model weights, and a loss function optimized for text generation was used. To enhance learning efficiency, the dataset was divided into batches during training. Summaries generated by the model were evaluated on the validation and test datasets using the ROUGE and BLEU metrics. These metrics compared the model-generated summaries against gold-standard summaries, providing insights into performance improvements.

The Github link of our code for Project Update 2 is Github:Text-Summarization-Financial-Text:Update-2

4 Current Results

Table 1 captures the key metrics we used for evaluating the performance of our models. These scores help assess the effectiveness of each model in capturing key information and generating coherent summaries. .

5 Current Analysis

Analyzing the performance of the summarization models using the provided metrics reveals several key insights about the effectiveness of the changes implemented:

1. **TextRank**:

- There was a noticeable impact of Top Sentence Adjustments that was undertaken to optimize the model wherein Top-n = 3 along with preprocessing changes and handling for missing data resulted in a decline in ROUGE and BLEU scores compared to the baseline, suggesting that including pre-processing impaired the model's ability to capture key information.

- With Top-n = 5 there were further small declines in validation but slight improvements in the test set compared to Top-n=3. This indicates limited improvements in generalization with a higher Top-n but lower consistency overall suggesting that a smaller number of sentences might be optimal for this model.

- The decrease in performance with increasing top-n sentences could be due to the model's inability to effectively rank sentences beyond a certain threshold.

2. **T5 - Transformer**:

- There were overall improvements observed in Rogue and Bleu scores because of including enhanced preprocessing and missing data handling for validation and test datasets wherein ROUGE-1 improved by over 40 percent and BLEU by nearly 150 percent compared to the baseline performance of Project Update 1, demonstrating its adaptability to changes.

- Hyperparameter tuning for increasing the min-length=200 further boosted performance further boosted performance by 2-5 percent. This indicates that the additional preprocessing and normalization steps, along with the increased minimum length, contribute to better summary quality.

3. **BART**:

- Initial changes of including preprocessing techniques and handling for missing values caused a sharp performance drop of over 30-50 percent across metrics, underscoring the sensitivity of this model to parameter adjustments. Also suggesting that the changes might not be beneficial for this model.

- Increasing minimum length helped recover some performance, but the scores remained 20-25 percent below baseline levels obtained during project Update-1, showing limited resilience to disruption.

4. **Pegasus**:

- Pegasus was evaluated after incorporating preprocessing changes, handling for missing values and min-length=200 the performance metrics for test dataset improved slightly, with ROUGE-1 (0.1835) and BLEU (0.0511). While not surpassing T5, Pegasus showed better baseline performance than the disrupted BART revisions and comparable coherence to TextRank.

5. **Comparative Study**:

- T5 stands out as the most improved model, with consistent and significant gains wherein it has consistently outperformed other models across all metrics.

- However, BART requires careful hyperparameter tuning to recover and surpass baseline performance. While during Project update 1 we were expecting the changes to show positive impact in terms of improved model performance; BART didn't really reflect that kind of performance.

- Pegasus offers stable, baseline-level results and could complement other models in hybrid approaches, making it a strong contender, amongst other model.

6 Next Steps: Upcoming Results and Analysis

- **Integrating Pegasus with T5:** Pegasus shows potential for integration with T5, especially for datasets like ours that requires versatility to generate more comprehensive and informative summaries. We anticipate this hybrid model implementation will exceed all past model performance.

- **Experiment with different hyperparameter settings:** Finetuning other key hyperparameters like like learning rate, batch size, and number of epochs increasing it to 5 epochs across all 4 (T5, BART, Pegasus and Hybrid model) models to check if it further improve model performance.

- **Automated hyperparameter tuning techniques:** Implementing automated hyperparameter tuning techniques, such as grid search or random search, to optimize configurations, particularly for advanced transformers like T5, BART, Pegasus and hybrid model (Pegasus + T5).

- **TextRank Experimentation:** While it showed reduced performance for top-n increase we want to view the impact of decreasing the top-n from 3 to 2 and verify if the decrease in top-n leads to improved model performance.

- **Graphical Analysis:** Incorporating visualizations, such as graphs, to analyze validation and test losses as well as performance metrics for all models.

- **Error Analysis:** Analyzing the error for kind of summaries generated through manual summary evaluation to verify if the model is hallucinating generating summary not part of annual report or capturing incorrect information or if the generated summary is being overly concise as compared to gold summary.

- **Exploring Alternative Evaluation Metrics:** We plan to explore an additional evaluation metrics to ensure a comprehensive assessment of our model's performance. These include:
  - **METEOR:** Combines precision, recall, and lexical matching for a well-rounded evaluation.
  - **BERTScore:** Utilizes BERT to evaluate semantic similarity between generated and reference summaries.
  - **ROUGE Scores:** Presented with precision, recall, and F1 score.

- **Comparison with Best Results for Task Submission:** The top-performing teams in the FNS 2020 task achieved ROUGE scores ranging from 0.20 to 0.40 on this dataset. Our objective is to ensure that at least one of our final submission models attains a ROUGE score within this range for the test data.

Model Name	**Dataset**	**Rogue-1 Score**	**Rogue-2 Score**	**Rogue-L Score**	**BLEU Score**
TextRank Project-1 update results	Validation	0.1932	0.0846	0.0998	0.0451
	Test	0.198	0.0908	0.1051	0.0484
TextRank with changes top-n=3	Validation	0.1048	0.0551	0.0613	0.0304
	Test	0.1045	0.0594	0.0673	0.0334
TextRank with changes top-n=5	Validation	0.0943	0.0631	0.0647	0.0342
	Test	0.0963	0.0674	0.0702	0.0376
T5 Project-1 update results	Validation	0.1645	0.0849	0.1188	0.029
	Test	0.1583	0.0842	0.1151	0.0291
T5 Revised with changes	Validation	0.2283	0.1316	0.1583	0.0829
	Test	0.2065	0.1091	0.1387	0.0612
T5 Revised with changes min-len=200	Validation	0.2336	0.1345	0.1613	0.0892
	Test	0.2171	0.1176	0.1452	0.0699
BART Project-1 update results	Validation	0.1811	0.0913	0.1244	0.0214
	Test	0.1715	0.0865	0.1182	0.0198
BART Revised with changes	Validation	0.1154	0.0089	0.0526	0.0038
	Test	0.1172	0.0089	0.0540	0.0037
BART Revised with changes min-len=200	Validation	0.1447	0.0576	0.0857	0.0350
	Test	0.1403	0.0528	0.0815	0.0279
Pegasus min-len=200	Validation	0.1786	0.0953	0.1310	0.0468
	Test	0.1835	0.1005	0.1327	0.0511

Table 1: Evaluation Results
*Changes here imply following elements- Enhanced Preprocessing and Text Normalization, Handling Missing Data