Advancing Domain-Specific Abstractive Summarization in the Financial Sector: A Comprehensive Comparative Study of Classical Extractive and Modern Transformer-Based Approaches

Project Proposal

Submitted by:
Eman Sarfraz (L1F22BSAI0034)
Ahmad Yasin (L1F22BSAI0052)

Core NLP Technique: Sequence-to-Sequence Abstractive Summarization
Contribution Type: Fine-tune to improve accuracy and ,In-Depth Comparative Analysis & Evaluation

Department of Applied Computing and Technology
Bachelor of Science in Artificial Intelligence
December 6, 2025

Table of Contents

16. References

---

1. Team Composition and Contribution Statement

Both team members are fully and equally engaged in every phase of the project — from literature review and methodology design to implementation, experimentation, analysis, and report writing. Our collaboration ensures comprehensive coverage of both theoretical depth and practical implementation.

Role	Name	Student ID	Primary Responsibilities
Project Coordinator & Report Author	Eman Sarfraz	L1F22BSAI0034	Literature review, experimental design, documentation, evaluation framework design
Technical Lead & Implementation Specialist	Ahmad Yasin	L1F22BSAI0052	Model implementation, training pipeline, evaluation metrics implementation, reproducibility

Collaboration Model: All major decisions, including methodology selection, experimental parameters, and result interpretation, are made jointly through regular coordination meetings and shared documentation.

---

2. Executive Summary

The financial industry faces unprecedented information overload, with analysts, investors, and regulatory bodies processing vast volumes of complex textual data daily including earnings call transcripts, SEC filings (10-K, 10-Q), analyst reports, and real-time market commentary. Manual synthesis of this information is increasingly impractical, creating urgent demand for automated, reliable summarization systems.

This research project conducts a **rigorous, domain-focused comparative study** of abstractive text summarization techniques specifically tailored to financial documents. We systematically benchmark state-of-the-art Transformer-based sequence-to-sequence models (BART and T5) against robust classical extractive baselines (TextRank enhanced with TF-IDF and sentence embeddings) using multiple financial datasets totaling over 15,000 document-summary pairs.

Our evaluation extends beyond conventional lexical overlap metrics (ROUGE) to incorporate **advanced semantic similarity measures** (BERTScore) and, critically, **factual consistency evaluation** using natural language inference (NLI) a requirement in finance where hallucinations can have severe legal and financial consequences. We employ parameter-efficient fine-tuning (LoRA/PEFT) to adapt pre-trained models to financial domain terminology and discourse patterns while maintaining computational feasibility.

The study delivers evidence-based insights into performance trade-offs across multiple dimensions: summary quality, computational efficiency, factual fidelity, domain-specific terminology handling, and interpretability.

Our findings provide actionable guidance for deploying reliable summarization systems in professional financial environments, bridging the gap between academic NLP research and real-world financial applications.

---

3. Introduction and Motivation

3.1 The Financial Information Overload Problem

The global financial sector generates approximately **2.5 quintillion bytes** of textual data daily (source: industry estimates). Financial analysts spend an estimated **40-60% of their time** on information gathering and synthesis rather than analysis and decision-making. Key challenges include:

- **Volume**: A single S&P 500 company may release hundreds of pages of quarterly reports
- **Complexity**: Financial texts contain dense technical terminology, numerical data, and legal language
- **Timeliness**: Market-moving information requires rapid processing and dissemination
- **Accuracy Requirements**: Errors in financial summarization can lead to misguided investment decisions, regulatory violations, or reputational damage

3.2 The Promise and Peril of Abstractive Summarization

Extractive summarization selecting and concatenating salient sentences guarantees factual consistency but produces stilted, often redundant summaries that lack coherence and may miss higher-level insights.

Abstractive summarization generating novel text that captures core meaning offers more natural, concise summaries and can synthesize information across multiple sentences. However, it introduces **hallucination risk**: the generation of plausible-sounding but factually incorrect content.

In financial contexts, this trade-off is particularly acute:

- **High-stakes domain**: Incorrect summaries can influence billion-dollar decisions
- **Regulatory scrutiny**: Financial communications are legally binding and auditable
- **Complex terminology**: Models must accurately preserve technical financial concepts
- **Numerical precision**: Revenue figures, percentages, and dates must be exact

3.3 Research Gap

While abstractive summarization has achieved impressive results on general-domain benchmarks (e.g., CNN/DailyMail), its application to **specialized, high-risk domains like finance** remains under-explored. Specifically:

1. Limited research on **domain-specific fine-tuning** for financial text

2. Insufficient emphasis on **factual consistency** versus fluency in evaluation
3. Lack of **systematic comparisons** between extractive and abstractive approaches in finance
4. Minimal investigation of **computational efficiency** trade-offs for production deployment

This project addresses these gaps through comprehensive comparative analysis.

---

4. Problem Statement and Research Questions

4.1 Primary Problem Statement

How can we reliably automate the summarization of complex financial documents while maintaining factual accuracy, domain terminology fidelity, and computational feasibility for production deployment?

4.2 Research Questions

RQ1: How do fine-tuned Transformer-based abstractive models (BART, T5) compare to classical extractive methods (TextRank) on financial text across multiple evaluation dimensions?

RQ2: What is the extent of factual consistency degradation (hallucination) in abstractive summaries of financial documents, and how does it vary by model architecture and training configuration?

RQ3: Can domain-specific fine-tuning with parameter-efficient methods (LoRA) enable pre-trained models to accurately handle financial terminology and numerical data?

RQ4: What are the computational cost trade-offs (training time, inference speed, memory requirements) between extractive and abstractive approaches for practical deployment?

RQ5: Under what conditions (document type, length, complexity) does each approach perform optimally, and what deployment recommendations emerge?

---

5. Literature Review

5.1 Classical Extractive Summarization

Extractive summarization has a rich history in NLP, offering interpretability and factual safety:

Graph-Based Methods:

- **TextRank** (Mihalcea & Tarau, 2004): Applies PageRank to sentence similarity graphs, treating sentences as nodes and similarity scores as edges
- **LexRank** (Erkan & Radev, 2004): Uses eigenvector centrality on sentence similarity matrices

- These methods are **deterministic, fast, and factually safe** by design

Feature-Based Methods:

- **TF-IDF** weighting combined with positional and length biases
- **Sentence position**, **query relevance**, and **named entity density** as ranking features
- Machine learning approaches using logistic regression or SVMs on hand-crafted features

Limitations:

- Lack of discourse coherence in selected sentences
- Redundancy when multiple similar sentences are selected
- Inability to perform information fusion across sentences
- Poor handling of implicit information

5.2 Neural Abstractive Summarization

The neural revolution transformed summarization through end-to-end learning:

Sequence-to-Sequence Era (2015-2017):

- **RNN-based encoder-decoders** with attention mechanisms (Bahdanau et al., 2015)
- **Pointer-Generator Networks** (See et al., 2017): Combined copying from source with generation, reducing hallucination
- **Coverage mechanisms** to prevent repetition

Transformer Era (2017-Present):

- **BERT** (Devlin et al., 2019): Bidirectional pre-training revolutionized NLP
- **BART** (Lewis et al., 2020): Denoising autoencoder combining bidirectional encoder with autoregressive decoder, trained on diverse corruption schemes
- **T5** (Raffel et al., 2020): Unified text-to-text framework treating all NLP tasks uniformly
- **PEGASUS** (Zhang et al., 2020): Pre-trained specifically for summarization using gap-sentence generation

Domain Adaptation:

- **FinBERT** (Araci, 2019): BERT fine-tuned on financial texts for sentiment analysis

- Limited work on domain-specific summarization fine-tuning

5.3 Evaluation Beyond ROUGE

Traditional ROUGE metrics (Lin, 2004) measure n-gram overlap but fail to capture semantic similarity or factual accuracy:

Semantic Metrics:

- **BERTScore** (Zhang et al., 2020): Computes similarity between BERT embeddings of generated and reference text
- **BLEURT** (Sellam et al., 2020): Learned metric trained on human judgments

Factual Consistency:

- **FactCC** (Kryscinski et al., 2020): NLI-based fact-checking for summarization
- **QuestEval** (Scialom et al., 2021): Question-answering based consistency evaluation
- **DAE** (Goyal & Durrett, 2021): Dependency arc entailment for factuality

5.4 Financial NLP Applications

Financial text processing presents unique challenges:

- **Named Entity Recognition** for companies, financial instruments, and regulatory references
- **Sentiment Analysis** on earnings calls and market commentary
- **Event Extraction** from news and filings
- **Limited research** specifically on financial document summarization

---

6. Theoretical Framework

6.1 Extractive Summarization as Graph-Based Ranking

TextRank models document summarization as a graph problem:

Graph Construction:

- **Nodes:** Sentences ($S = \{s_1, s_2, ..., s_n\}$)
- **Edges:** Weighted by sentence similarity ($sim(s_i, s_j)$)

Similarity Computation: [sim(s_i, s_j) = \frac{|\{w_k | w_k \in s_i \land w_k \in s_j\}|}{log(|s_i|) + log(|s_j|)}]

Ranking: [Score(s_i) = (1-d) + d \times \sum_{s_j \in In(s_i)} \frac{w_{ji} \times Score(s_j)}{\sum_{s_k \in Out(s_j)} w_{jk}}]

where (d) is damping factor (typically 0.85).

6.2 Abstractive Summarization as Conditional Generation

Given source document (X = (x_1, ..., x_m)), generate summary (Y = (y_1, ..., y_n)) by maximizing:

[
P(Y|X) = \prod_{t=1}^{n} P(y_t | y_{<t}, X)
]

BART Architecture:

- **Encoder:** Bidirectional Transformer processing corrupted input
- **Decoder:** Autoregressive Transformer generating summary
- **Pre-training:** Denoising objectives (token masking, deletion, infilling, sentence permutation)

6.3 Factual Consistency via Natural Language Inference

Frame consistency checking as entailment:

- **Premise:** Source document (X)
- **Hypothesis:** Generated summary sentence (y_i)
- **Task:** Classify as Entailment, Neutral, or Contradiction

[
FactScore(Y|X) = \frac{1}{|Y|} \sum_{y_i \in Y} \mathbb{1}[NLI(X, y_i) = \text{Entailment}]
]

---

7. Methodology

7.1 Reference Paper for State-of-the-Art Model

Lewis, M., et al. (2020). "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension." ACL 2020.

Justification:

- BART's denoising pre-training paradigm is particularly effective for generation tasks requiring strong comprehension and fluent synthesis
- Demonstrated state-of-the-art performance on CNN/DailyMail and XSum benchmarks
- Flexibility in handling various corruption schemes makes it robust to noisy financial text
- Availability of pre-trained checkpoints and Hugging Face implementation ensures reproducibility

Alternative Consideration: T5-base will be evaluated in parallel for comparative insights.

7.2 Models to be Implemented

Category	Model	Implementation Details
Extractive Baseline	TextRank + TF-IDF	Custom Python implementation using NetworkX and scikit-learn
Extractive Enhanced	TextRank + Sentence-BERT	Using sentence-transformers for semantic similarity
Abstractive (Primary)	BART-Large (406M params)	Hugging Face Transformers, LoRA-based fine-tuning
Abstractive (Secondary)	T5-Base (220M params)	Hugging Face Transformers, comparison benchmark

7.3 Datasets

We curate a comprehensive financial text corpus from multiple sources:

Dataset 1: Financial News Articles

- **Source:** Kaggle Financial News Dataset (15,000+ articles)
- **URL:** https://www.kaggle.com/datasets/yogeshchary/financial-news-dataset
- **Coverage:** Market news, company announcements, economic indicators
- **Summary Availability:** Human-written headlines and first sentences used as reference summaries

Dataset 2: Earnings Call Transcripts

- **Source:** Kaggle Earnings Call Transcripts Dataset (2,000+ transcripts)
- **URL:** https://www.kaggle.com/datasets/ramssvimala/earning-call-transcripts

- **Coverage:** Quarterly earnings calls from public companies
- **Summary Availability:** Management discussion sections serve as pseudo-summaries

Dataset 3: High-Quality Financial News (NLP-Ready)

- **Source:** Kaggle High-Quality Financial News Dataset
- **URL:** https://www.kaggle.com/datasets/sayelabualigah/high-quality-financial-news-dataset-for-nlp-tasks
- **Coverage:** Curated financial news with enhanced metadata
- **Preprocessing:** Pre-cleaned and tokenized

General-Domain Baseline:

- **CNN/DailyMail:** 287,000 article-summary pairs for initial model validation

Total Training Data: ~15,000 financial document-summary pairs
Validation Split: 10% (stratified by document type)
Test Split: 10% (held-out, never seen during training)

7.4 Financial-Specific Preprocessing Pipeline

Financial text requires specialized preprocessing:

1. **Currency Normalization:**
   - Convert all currency representations to standardized format: $XX.XX billion
   - Handle multiple currencies (USD, EUR, GBP, etc.)
2. **Numerical Standardization:**
   - Preserve precision for financial figures
   - Normalize percentage representations
   - Handle scientific notation
3. **Temporal Expression Handling:**
   - Normalize fiscal year references (FY2024, Q3 2025)
   - Standardize date formats
4. **Boilerplate Removal:**
   - Strip legal disclaimers and forward-looking statements
   - Remove repeated section headers
5. **Entity Recognition:**

- Preserve company names, financial instruments, and regulatory references
- Maintain proper nouns without aggressive lowercasing

6. **Truncation Strategy:**
    - Input: 1024 tokens (BART limit)
    - Output: 256 tokens (summary length target)
    - Sliding window approach for documents exceeding limits

7.5 Training Configuration

Extractive Models:

- **TextRank:**
    - Damping factor: 0.85
    - Similarity threshold: 0.2
    - Top-k sentences: 5-7 (dynamically based on document length)
- **Sentence-BERT Enhanced:**
    - Model: `all-mpnet-base-v2`
    - Cosine similarity for edge weights

Abstractive Models:

BART-Large Fine-Tuning:

Model: facebook/bart-large-cnn (fine-tuned starting point)
Parameter-Efficient Training: LoRA (r=8, alpha=16)
Learning Rate: 5e-5 (linear warmup: 500 steps)
Batch Size: 4 (gradient accumulation: 8 steps)
Effective Batch Size: 32
Epochs: 5
Optimizer: AdamW (β1=0.9, β2=0.999, ε=1e-8)
Weight Decay: 0.01
Max Input Length: 1024 tokens
Max Output Length: 256 tokens
Beam Search: 4 beams
Length Penalty: 2.0
Hardware: Single NVIDIA A100 GPU (40GB)
Mixed Precision: FP16
Training Time: ~18 hours

T5-Base Fine-Tuning:

Model: t5-base
LoRA Configuration: r=8, alpha=16
Learning Rate: 3e-4
Other parameters: Similar to BART
Training Time: ~12 hours

Reproducibility:

- Fixed random seeds: `torch.manual_seed(42)`, `numpy.random.seed(42)`
- Deterministic CUDA operations enabled
- Version pinning: PyTorch 2.0+, Transformers 4.35+

---

8. Experimental Design

8.1 Baseline Establishment

1. **General-Domain Performance:**
    - Evaluate pre-trained BART-CNN on CNN/DailyMail test set
    - Establish upper-bound performance expectations
2. **Zero-Shot Financial Performance:**

- Apply pre-trained models directly to financial test set without fine-tuning
- Quantify domain gap

8.2 Fine-Tuning Experiments

Experiment 1: Impact of Domain-Specific Fine-Tuning

- Compare zero-shot vs. fine-tuned BART on financial datasets
- Measure improvement in ROUGE, BERTScore, and factual consistency

Experiment 2: Parameter-Efficient Training

- Full fine-tuning vs. LoRA vs. adapter layers
- Evaluate performance-efficiency trade-offs

Experiment 3: Dataset Size Sensitivity

- Training curves with 1k, 5k, 10k, 15k examples
- Determine minimum data requirements for effective adaptation

8.3 Comparative Evaluation

Head-to-Head Comparison:

- TextRank (baseline)
- TextRank + Sentence-BERT
- BART-Large fine-tuned
- T5-Base fine-tuned

Evaluation Dimensions:

1. Lexical overlap (ROUGE-1/2/L)
2. Semantic similarity (BERTScore F1)
3. Factual consistency (FactCC score)
4. Computational efficiency (tokens/sec, memory usage)
5. Human evaluation (subset of 100 examples rated by domain experts)

---

9. Evaluation Framework

9.1 Automated Metrics

Metric Category	Specific Metric	Purpose	Implementation
Lexical Overlap	ROUGE-1	Unigram overlap	`rouge-score` library
	ROUGE-2	Bigram overlap (fluency proxy)	
	ROUGE-L	Longest common subsequence	
Semantic Similarity	BERTScore (F1)	Contextual embedding similarity	`bert-score` library
Factual Consistency	FactCC Score	NLI-based entailment	Fine-tuned RoBERTa-large
	Hallucination Rate	% sentences not entailed by source	Custom NLI pipeline
Information Coverage	Entity Recall	% key entities in summary	spaCy NER + custom financial entities
Computational Efficiency	Inference Speed	Tokens/second on GPU	Custom timing
	Memory Usage	Peak GPU memory (GB)	`torch.cuda.max_memory_allocated()`

9.2 Qualitative Analysis

Error Taxonomy:

1. **Hallucination:** Generated content not supported by source
2. **Omission:** Critical information missing from summary
3. **Over-compression:** Loss of important details
4. **Terminology Errors:** Incorrect financial terms or concepts
5. **Numerical Errors:** Wrong figures or percentages

Case Study Analysis:

- Select 50 diverse documents (earnings calls, news, filings)
- Generate summaries from all models

- Annotate errors by type
- Identify systematic failure patterns

9.3 Human Evaluation (Limited Scope)

Due to resource constraints, human evaluation will be conducted on a stratified sample:

- **Sample Size:** 100 document-summary pairs
- **Evaluators:** 2 team members + 1 external financial domain expert (if available)
- **Criteria:**
  - Factual Accuracy (1-5 scale)
  - Coherence (1-5 scale)
  - Relevance (1-5 scale)
  - Overall Quality (1-5 scale)
- **Inter-Annotator Agreement:** Cohen's Kappa

---

10. Anticipated Challenges and Mitigation Strategies

Challenge	Impact	Mitigation Strategy
High computational cost of training	May exceed available GPU resources	Use LoRA for parameter-efficient fine-tuning; leverage Colab Pro or university GPU clusters
Limited financial summary data	Insufficient training examples	Augment with pseudo-summaries from leads/abstracts; apply data augmentation techniques
Hallucination in abstractive outputs	Undermines practical deployment	Implement comprehensive factual consistency evaluation; explore constrained decoding
Domain terminology complexity	Poor handling of financial jargon	Create custom tokenizer vocabulary extensions; use financial entity lexicons
Evaluation subjectivity	Difficult to define "good" summary objectively	Combine multiple automated metrics; include limited human evaluation
Reproducibility concerns	Others may struggle to replicate	Extensive documentation; fixed seeds; version control; public code release

11. Timeline and Milestones

Project Duration: December 6, 2025 – January 10, 2026 (5 weeks)

Phase 1: Foundation (Week 1: Dec 6-12)

- ✓ Complete literature review
- ✓ Finalize dataset selection and acquisition
- ✓ Set up development environment (Colab notebooks, version control)
- ✓ Implement data preprocessing pipeline
- **Deliverable:** Cleaned, preprocessed datasets ready for modeling

Phase 2: Baseline Implementation (Week 2: Dec 13-19)

- Implement TextRank extractive baseline
- Implement Sentence-BERT enhanced extractive model
- Conduct initial extractive model evaluation
- **Deliverable:** Working extractive baselines with preliminary results

Phase 3: Abstractive Model Development (Week 3: Dec 20-26)

- Load and configure pre-trained BART and T5 models
- Implement LoRA fine-tuning pipeline
- Begin model training on financial datasets
- **Deliverable:** Fine-tuned abstractive models

Phase 4: Evaluation and Analysis (Week 4: Dec 27 - Jan 2)

- Compute all automated metrics (ROUGE, BERTScore, FactCC)
- Conduct qualitative error analysis
- Perform computational efficiency benchmarking
- Generate visualizations and comparative tables
- **Deliverable:** Complete experimental results

Phase 5: Documentation and Reporting (Week 5: Jan 3-10)

- Write comprehensive final report
- Create presentation materials
- Prepare code for public release
- Finalize Google Colab notebook with full reproducibility
- **Deliverable:** Final report, presentation, and reproducible code

Critical Milestones:

- **Dec 19:** Extractive baselines complete
- **Dec 26:** Abstractive models trained
- **Jan 2:** All evaluations complete
- **Jan 10:** Final submission

---

12. Expected Contributions

This project delivers the following contributions:

12.1 Empirical Contributions

1. **First systematic comparison** of extractive vs. abstractive summarization specifically on financial texts with emphasis on factual consistency
2. **Comprehensive multi-dimensional evaluation** beyond standard ROUGE metrics
3. **Quantified performance-efficiency trade-offs** for production deployment guidance

12.2 Methodological Contributions

1. **Financial-aware preprocessing pipeline** handling currency, numbers, and domain terminology
2. **Reproducible fine-tuning protocol** using parameter-efficient methods (LoRA)
3. **Factual consistency evaluation framework** adapted for financial domain

12.3 Practical Contributions

1. **Deployment recommendations** specifying when extractive vs. abstractive is appropriate

2. **Open-source implementation** with fully documented Colab notebook
3. **Error taxonomy** for financial summarization failures

12.4 Academic Positioning

Target Venue: This work is suitable for submission to:

- NLP workshops at major conferences (EMNLP, ACL, NAACL)
- Domain-specific venues (Financial NLP Workshop, Finance & Economics on the Semantic Web)
- Undergraduate research symposia

Novelty Statement: While we do not claim to introduce a novel architecture or algorithm, we provide valuable empirical insights into the application of state-of-the-art methods to a critical, under-explored domain with unique requirements.

---

13. Future Extensions

This project establishes a foundation for several promising research directions:

13.1 Hybrid Architectures

- **Extractive-Abstractive Fusion:** Use TextRank outputs as soft prompts or constraints for abstractive generation
- **Retrieval-Augmented Generation:** Incorporate external financial knowledge bases

13.2 Controllable Summarization

- **Aspect-Conditioned Summaries:** Generate summaries focused on specific financial aspects (revenue, risk, guidance, market reaction)
- **Length and Style Control:** User-specified summary length and formality level

13.3 Multilingual and Cross-Domain

- **Multilingual Financial Summarization:** Extend to non-English financial documents using mBART
- **Cross-Domain Transfer:** Evaluate transfer learning from financial to legal or medical domains

13.4 Real-Time Applications

- **Streaming Summarization:** Process real-time earnings call transcripts with incremental updates

- **Multi-Document Synthesis:** Aggregate information from multiple sources (e.g., news + filings + analyst reports)

13.5 Enhanced Factuality

- **Constrained Decoding:** Enforce entity and numerical accuracy during generation
- **Post-Hoc Fact Verification:** Integrate automated fact-checking modules

---

14. Ethical Considerations

14.1 Potential Risks

1. **Misuse for Market Manipulation:** Automated summarization could be used to rapidly disseminate misleading information
2. **Over-Reliance on AI:** Users may trust summaries without verifying source documents
3. **Bias Amplification:** Models may perpetuate biases present in training data (e.g., overrepresentation of large-cap companies)
4. **Job Displacement:** Automation may affect financial analyst roles

14.2 Mitigation Measures

- Include disclaimers emphasizing that summaries are AI-generated and should be verified
- Provide source document links alongside summaries
- Transparent reporting of model limitations and failure modes
- Advocate for AI as augmentation tool rather than replacement for human judgment

14.3 Regulatory Compliance

- Acknowledge that financial communications are subject to securities regulations
- Recommend human review before public dissemination
- Maintain audit trails of generated summaries

---

15. Conclusion

This project undertakes a rigorous, domain-focused investigation of text summarization in the financially critical domain. By systematically comparing classical extractive and modern Transformer-based abstractive approaches across multiple evaluation dimensions — with particular emphasis on factual consistency — we aim to provide clear, evidence-based guidance for deploying reliable summarization systems in professional financial settings.

Our research bridges the gap between academic NLP advances and real-world financial applications, addressing the unique challenges of a high-stakes domain where accuracy is non-negotiable. Through comprehensive experimentation, transparent evaluation, and careful analysis of failure modes, we contribute meaningfully to both the NLP research community and the practical needs of financial professionals navigating information overload.

The deliverables — a fully reproducible Google Colab implementation, detailed comparative analysis, and deployment recommendations — will serve as a valuable resource for researchers and practitioners seeking to understand the capabilities and limitations of summarization technologies in specialized domains.

---

16. References

Core Methodology

1. **Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., ... & Zettlemoyer, L. (2020).** BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *ACL 2020*.
2. **Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... & Liu, P. J. (2020).** Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140), 1-67.

Extractive Baselines

3. **Mihalcea, R., & Tarau, P. (2004).** TextRank: Bringing order into texts. *EMNLP 2004*.
4. **Erkan, G., & Radev, D. R. (2004).** LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22, 457-479.

Evaluation Metrics

5. **Lin, C. Y. (2004).** ROUGE: A package for automatic evaluation of summaries. *Text Summarization Branches Out, ACL Workshop*.

6. **Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2020).** BERTScore: Evaluating text generation with BERT. *ICLR 2020*.
7. **Kryscinski, W., McCann, B., Xiong, C., & Socher, R. (2020).** Evaluating the factual consistency of abstractive text summarization. *EMNLP 2020*.
8. **Scialom, T., Dray, P. A., Lamprier, S., Piwowarski, B., & Staiano, J. (2021).** QuestEval: Summarization asks for fact-based evaluation. *EMNLP 2021*.

Parameter-Efficient Training

9. **Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... & Chen, W. (2021).** LoRA: Low-rank adaptation of large language models. *ICLR 2022*.
10. **Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., ... & Gelly, S. (2019).** Parameter-efficient transfer learning for NLP. *ICML 2019*.

Neural Abstractive Summarization

11. **See, A., Liu, P. J., & Manning, C. D. (2017).** Get to the point: Summarization with pointer-generator networks. *ACL 2017*.
12. **Zhang, J., Zhao, Y., Saleh, M., & Liu, P. (2020).** PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. *ICML 2020*.

Financial NLP

13. **Araci, D. (2019).** FinBERT: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*.
14. **Chen, C. C., Huang, H. H., & Chen, H. H. (2020).** From opinion mining to financial argument mining. *Proceedings of the First Workshop on Financial Technology and Natural Language Processing*, 56-66.

Foundational Deep Learning

15. **Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019).** BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL 2019*.
16. **Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017).** Attention is all you need. *NeurIPS 2017*.

Evaluation and Consistency

17. **Goyal, T., & Durrett, G. (2021).** Annotating and modeling fine-grained factuality in summarization.

NAACL 2021.

18. **Sellam, T., Das, D., & Parikh, A. P. (2020).** BLEURT: Learning robust metrics for text generation. *ACL 2020*.

Additional Financial Resources

19. **Malo, P., Sinha, A., Korhonen, P., Wallenius, J., & Takala, P. (2014).** Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4), 782-796.
20. **Xie, Q., Dai, Z., Hovy, E., Luong, M. T., & Le, Q. V. (2020).** Unsupervised data augmentation for consistency training. *NeurIPS 2020*.

Appendices

Appendix A: Detailed Dataset Statistics

Dataset	# Documents	Avg. Length (words)	Domain Coverage	Summary Type
Financial News	15,000	450	Market news, company announcements	Headlines + lead sentences
Earnings Transcripts	2,000	3,500	Quarterly earnings calls	Management discussion sections
High-Quality News	8,000	380	Curated financial news	Professional summaries
CNN/DailyMail (baseline)	287,000	760	General news	Human-written highlights

Appendix B: Computational Resources

Required Infrastructure:

- GPU: NVIDIA A100 (40GB) or V100 (32GB)
- RAM: 64GB system memory
- Storage: 100GB for datasets and model checkpoints
- Platform: Google Colab Pro+ or university HPC cluster

Estimated Costs:

- Colab Pro+ subscription: $50/month
- Total compute time: ~50 GPU hours
- Estimated cost: $100-150 (if using cloud resources)

Appendix C: Code Repository Structure

```
financial-summarization/
│
├── data/
│   ├── raw/                # Original datasets
│   ├── processed/          # Preprocessed data
│   └── splits/             # Train/val/test splits
│
├── src/
│   ├── preprocessing.py      # Financial text preprocessing
│   ├── extractive_models.py  # TextRank implementations
│   ├── abstractive_models.py # BART/T5 fine-tuning
│   ├── evaluation.py         # Metrics computation
│   └── utils.py              # Helper functions
│
├── notebooks/
│   ├── 01_data_exploration.ipynb
│   ├── 02_preprocessing.ipynb
│   ├── 03_extractive_baseline.ipynb
│   ├── 04_abstractive_training.ipynb
│   ├── 05_evaluation.ipynb
│   └── 06_analysis_visualization.ipynb
│
├── configs/
│   ├── bart_config.yaml
│   └── t5_config.yaml
│
├── results/
│   ├── metrics/
│   ├── summaries/
│   └── visualizations/
│
├── requirements.txt
```

```
├── README.md
└── LICENSE
```

Appendix D: Sample Financial Preprocessing Code

```python
import re
from typing import List, Dict

def normalize_financial_text(text: str) -> str:
    """
    Normalize financial-specific elements in text.
    """
    # Normalize currency
    text = re.sub(r'\$(\d+\.?\d*)\s*(billion|B)', r'$\1 billion', text)
    text = re.sub(r'\$(\d+\.?\d*)\s*(million|M)', r'$\1 million', text)

    # Normalize percentages
    text = re.sub(r'(\d+\.?\d*)\s*%', r'\1%', text)

    # Normalize fiscal periods
    text = re.sub(r'(FY|fiscal year)\s*(\d{4})', r'FY\2', text)
    text = re.sub(r'(Q[1-4])\s*(\d{4})', r'\1 \2', text)

    return text

def remove_boilerplate(text: str) -> str:
    """
    Remove common financial document boilerplate.
    """
    # Remove forward-looking statement disclaimers
    boilerplate_patterns = [
        r'This presentation contains.*?forward-looking statements.*?(?=\n\n)',
        r'Safe Harbor Statement.*?(?=\n\n)',
        r'Non-GAAP Financial Measures.*?(?=\n\n)'
    ]

    for pattern in boilerplate_patterns:
        text = re.sub(pattern, '', text, flags=re.DOTALL | re.IGNORECASE)

    return text
```

Academic Positioning Statement

This project presents a structured, methodologically rigorous comparative evaluation of extractive and abstractive summarization techniques in the financial domain. Rather than claiming algorithmic novelty, we emphasize empirical contributions: comprehensive multi-dimensional evaluation with particular focus on factual consistency, domain-specific adaptation methodologies, and practical deployment considerations. Our work bridges academic NLP research and real-world financial applications, providing evidence-based guidance for system design and deployment in high-stakes professional environments.

Acknowledgments

We thank the Department of Computer Science and the AI program faculty for their guidance and support. We acknowledge the use of publicly available datasets from Kaggle and Hugging Face, and the open-source contributions of the PyTorch, Transformers, and NLP research communities.

Project Repository: To be made public upon completion
Contact: eman.sarfraz@nexariza.com, ahmad.yasin@nexariza.com

Last Updated: December 6, 2025