Hybrid Abstractive-Extractive Summarization of Romanian Parliamentary Proceedings

Natural Language Processing Techniques - Practical Project

Alexandru Lorintz

January 16, 2026

Contents

1 Problem Statement

The primary objective of this project is to develop an automatic text summarization (ATS) system specifically tailored for Romanian parliamentary proceedings. The source data originates from the Chamber of Deputies (Camera Deputaților), where sessions often span 3 to 5 hours, resulting in transcripts exceeding 300,000 characters.

This task presents two significant NLP challenges:

1. **High Noise Ratio:** Raw transcripts contain administrative roll calls, procedural interruptions, applause, and repetitive polite formulations (e.g., "Stimați colegi", "Doamna președinte") which dilute the core political arguments.

2. **Context Window Limitations:** Standard Transformer-based models (like BERT or T5) typically have a limit of 512 tokens. Processing a 300k-character document requires sophisticated handling beyond simple truncation.

The goal is to produce a concise, coherent summary that captures the legislative outcomes (laws debated, votes taken) while filtering out procedural noise.

2 Proposed Solution

To address the limitations of purely extractive or purely abstractive approaches on such long documents, I propose a **Hybrid Pipeline**. This architecture uses an Extractive model as a "content selector" to filter the document, followed by an Abstractive model acting as a "rewriter" to synthesize the selection.

2.1 Theoretical Background

Automatic Text Summarization (ATS) is the process of shortening a set of data computationally, to create a subset (a summary) that represents the most important or relevant information within the original content. There are two main paradigms in ATS:

2.1.1 Extractive vs. Abstractive Summarization

- **Extractive Summarization** works by selecting specific sentences or phrases directly from the source text and assembling them to form a summary. It is analogous to highlighting important sections in a book with a marker.
  - *Pros:* Factual hallucination is impossible since every word comes from the source. It preserves the original tone.
  - *Cons:* Can result in disjointed flow; cannot synthesize information distributed across multiple sentences; strictly bound by source vocabulary.
- **Abstractive Summarization** involves generating new sentences that capture the essence of the source text, similar to how a human would write a summary. It uses Natural Language Generation (NLG).
  - *Pros:* Can rephrase, paraphrase, and condense information more naturally. Can merge ideas from different parts of the text.
  - *Cons:* Computationally expensive; prone to "hallucinations" (generating non-factual information); requires massive training data.

2.1.2 The Extractive Model: MatchSum (Siamese BERT)

For the extraction phase, I employ **MatchSum** (Matching Summarization), which represents a shift from sentence-level scoring to document-level matching.

- **Base Architecture (BERT):** I use `bert-base-romanian-cased-v1` ($\approx$ **110M parameters**), a Transformer encoder pre-trained on a large corpus of Romanian text. BERT uses a mechanism called "Self-Attention" to understand the context of each word relative to every other word.
- **Siamese Network Structure:** MatchSum uses a Siamese framework, meaning two identical BERT models share weights. One encodes the whole document D, and the other encodes a candidate summary C.
- **Semantic Matching:** The model projects D and C into a shared vector space and calculates their cosine similarity. The goal is to find the set of sentences C that is semantically closest to the entire document D. This captures global context better than scoring sentences in isolation.

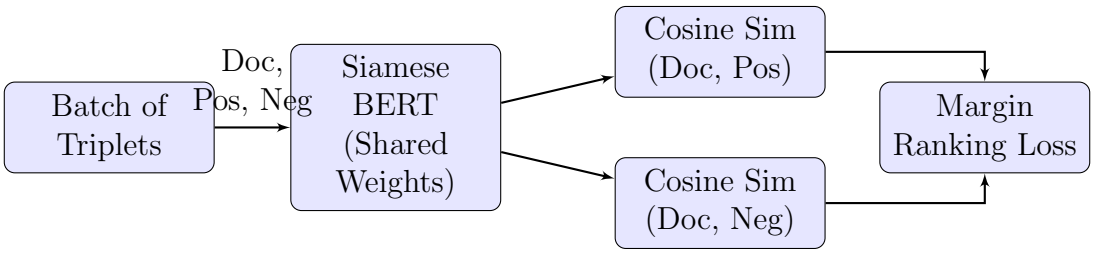

Figure 1: Training Pipeline for the Extractive (MatchSum) Model

2.1.3 The Abstractive Model: mT5

For generation, I use **mT5-small (Multilingual Text-to-Text Transfer Transformer)** which has roughly **300M parameters**.

- **Encoder-Decoder Architecture:** Unlike BERT (which is Encoder-only), T5 is an Encoder-Decoder model. The Encoder processes the input text into a high-dimensional representation, and the Decoder generates the summary token-by-token.
- **Span Corruption Pre-training:** mT5 was pre-trained on the mC4 dataset (101 languages) using a "fill-in-the-blank" objective. It learns to reconstruct missing spans of text, giving it strong text generation capabilities.
- **Fine-tuning:** I fine-tuned the `google/mt5-small` checkpoint specifically on my Parliamentary dataset to adapt its generation style to the formal, legislative language of the Romanian Chamber of Deputies.

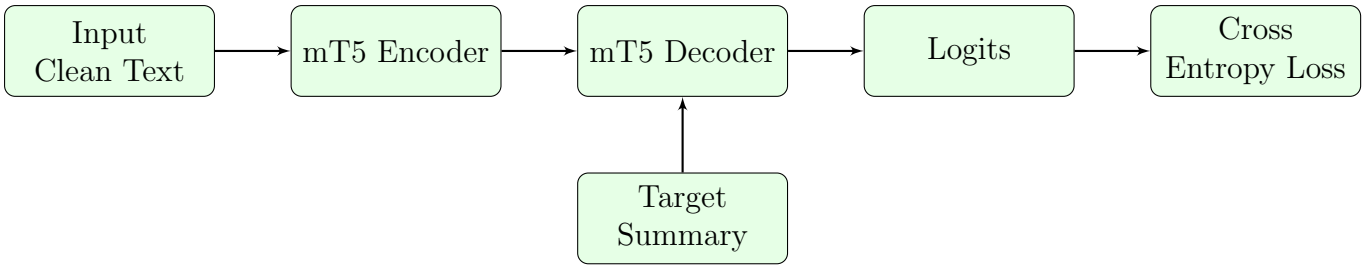

Figure 2: Training Pipeline for the Abstractive (mT5) Model

2.2 Data Set Used in Application

I constructed a custom dataset by scraping the official *Camera Deputaților* website.

- **Source:** `cdep.ro` (Debates section).
- **Extraction:** I extracted the full transcript (Source) and the "SUMAR" section (Target/Ground Truth).
- **Size:** The dataset comprises approximately 1.000 meetings spanning the 2017-2022 legislative periods.
- **Preprocessing:** Data was cleaned to remove HTML tags and encoding errors. The average input length is 150,000 characters.
- **Splits:**
  - **Training Data:** I utilized sessions with IDs **8000 to 9000**, corresponding roughly to the **2019-2022** legislative period.
  - **Evaluation Data:** I evaluated on a held-out set of IDs **7800 to 7950**, primarily from the **2017-2018** sessions.

2.3 Application Diagram

The application flow is illustrated below:

1. Input: Raw Transcript (300k chars)
↓
2. Preprocessing: Spacy Sentence Splitting
↓
3. Extractive Filter (MatchSum): Selects top 15% semantic sentences
↓
4. Regex Cleaning: Removes titles/names
↓
5. Chunking Strategy: Sliding window (450 tokens)
↓
6. Abstractive Generator (mT5): Generates summary segments
↓
7. Output: Final Concatenated Summary

Figure 3: The Hybrid Summarization Pipeline

3 Implementation Details

The critical innovation of this project lies in the robust engineering of the data pipeline and the seamless integration of two disparate models. The implementation is modular, organized into `processing`, `inference`, and `evaluation` components.

3.1 Data Acquisition & Processing Pipeline

Handling 300k+ character documents requires rigorous preprocessing to prevent "garbage in, garbage out".

A. Scraping strategy: I implemented a robust scraper (`CDEPScraper`) that handles legacy SSL protocols (SECLEVEL=1) required to access the government server. It iterates through session IDs, extracting the raw HTML content.

B. Cleaning & Segmentation: Raw transcripts are extremely noisy. The `DataProcessor` pipeline performs the following:

1. **Noise Removal:** Regex patterns strip procedural text (e.g., "(applause)", "Point 3 on the agenda").
2. **Spacy Segmentation:** I use `ro_core_news_sm` to split the text into sentences. This is non-trivial because of abbreviations (e.g., "Art. 5", "O.U.G. nr. 10"). Spacy's dependency parser helps correctly identify sentence boundaries where simple rule-based splitting fails.
3. **Dialogue Grouping:** Turns are grouped by speaker to maintain context coherence.

3.2 The "Vacuum" Filter (MatchSum Logic)

The Extractive model acts as a "vacuum", sucking out the most important sentences to create a condensed representation of the document.

1. **Global Candidate Scoring:** Unlike the abstractive model which requires chunking, the Extractive model can iterate through **all sentences** in the document.
   - I tokenize every sentence individually (S_i).
   - I compare each S_i against the dense vector representation of the document's context (D).
   - This allows the model to "hunt" for key information from the beginning, middle, and end of the transcript without windowing boundaries.
2. **Selection Heuristic Parameter Tuning:** I empirically determined a selection ratio of $r = 0.15$ with strict bounds:

$$k = \max(20, \min(80, N \times 0.15))$$

   The rationale behind these values is threefold:

- **Why 15%?** There is a natural trade-off between ROUGE scores and summary conciseness. My experiments demonstrated that increasing the ratio from 5% to 10% to 15% yielded consistent gains in ROUGE (higher recall of key facts). However, beyond this point, the gain diminishes while the noise drastically increases.
- **Why Max=80?** Computing resources and the Abstractive model's context window are finite. Feeding more than 80 sentences ($\approx$ 2000 tokens) tends to overwhelm the mT5 decoder, leading to "forgetting" or hallucinations. This cap acts as a safety valve for marathon sessions.
- **Why Min=20?** For extremely short debates, a percentage-based cut might yield only 1-2 sentences, lacking the context required for a coherent narrative. I enforce a minimum floor to ensure the generator always has enough material to work with.

3.3 "Divide and Conquer" Chunking Strategy

Even after filtering, the text might exceed the 512-token limit of mT5-small. To handle this, I employ a multi-step strategy:

1. **Regex Post-Processing:** Before generation, I strip specific parliamentary boilerplate that might confuse the model. This is handled in `src/processing/post_processing.py`:

```
    # Aggressively remove titles and polite fluff
    patterns = [
        r"Deputat.*?(?=\.)",
        r"Circumscriptia.*?(?=\.)",
        r"Declaratia.*?intitulata",
        r"Stimati colegi",
        r"Doamna presedinte",
        r"Domnule presedinte",
    ]
    for pat in patterns:
        text = re.sub(pat, "", text, flags=re.IGNORECASE)

    return re.sub(r"\s+", " ", text).strip()

```

2. **Sliding Windowing:** The clean text is then split into chunks of 450 tokens.
3. **Processing:** Each chunk is prepended with the prompt "summarize:" and fed to mT5.
4. **Generation Override:** I explicitly override the default generation configuration to prevent premature stopping:

```
    gen = pipeline(
        "summarize: " + chunk_text,
        max_length=300,         # Allow long summaries
        min_length=60,          # Force detailed output
        num_beams=4,            # Beam Search for higher quality
        no_repeat_ngram_size=3 # Strict penalty for repetition
    )

```

5. **Concatenation:** The partial summaries are joined to form the final result.

4 Experiments and Results

I evaluated my models in a test set of **64 meetings**. I utilized the **ROUGE** metric (Recall-Oriented Understudy for Gisting Evaluation) to measure overlap with the official human-written summaries.

4.1 Experimental Setup

I compared three approaches:

1. **Baseline (Blind Slicing):** Feed the first, middle, and last 5,000 characters directly to mT5 without intelligent filtering.
2. **Extractive Only (MatchSum):** The raw output of the MatchSum model.
3. **Hybrid:** the complete pipeline (MatchSum $\rightarrow$ mT5).

4.2 Evaluation Metric: ROUGE

To quantitatively assess the quality of my summaries, I utilize the **ROUGE** (Recall-Oriented Understudy for Gisting Evaluation) metric, which is the standard de-facto metric for NLP summarization tasks. ROUGE measures the overlap between the generated summary (S) and the reference human summary (R).

I focus on two specific variants:

- **ROUGE-1:** Measures the overlap of unigrams (single words). It is effective for assessing content coverage (informativeness).

$$\text{ROUGE-1} = \frac{\sum_{r \in R} \sum_{w \in r} \text{count}_{\text{match}}(w)}{\sum_{r \in R} \sum_{w \in r} \text{count}(w)}$$

- **ROUGE-L:** Based on the Longest Common Subsequence (LCS). It takes into account sentence-level structure similarity and identifies the longest co-occurring in-sequence n-grams automatically. It captures sentence fluency better than n-gram overlap.

4.3 Results

The final quantitative results are presented in Table 1 and visualized in Figure 4.

Method	ROUGE-1 (Content)	ROUGE-L (Structure)
1. Baseline (Blind Slicing)	0.3051	0.1837
2. Extractive (MatchSum)	0.3380	**0.2523**
3. Hybrid	**0.3439**	0.2367

Table 1: Comparative performance on N=64 test meetings.

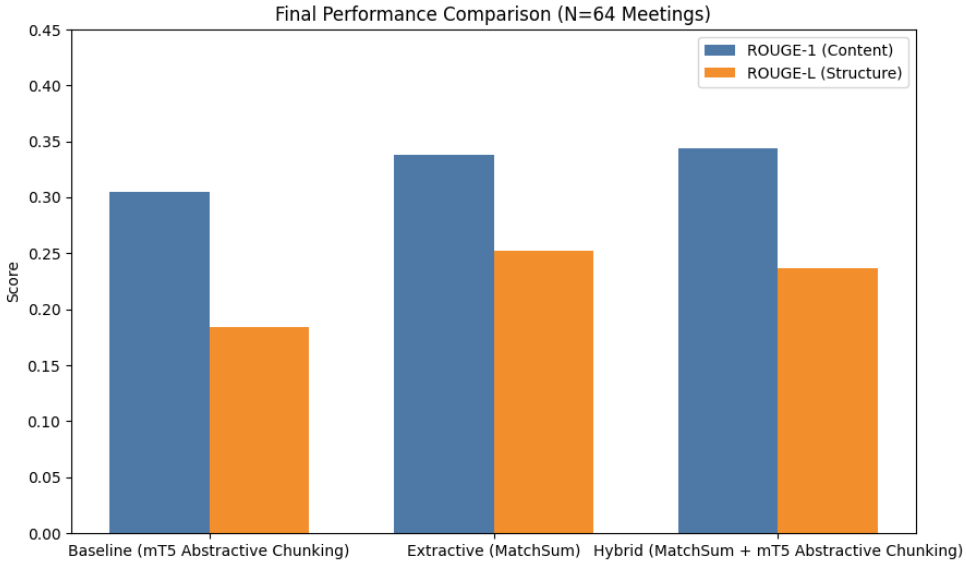

Figure 4: Performance comparison of the three strategies. The Hybrid model (Right) outperforms others in content coverage (ROUGE-1), while the Extractive model (Center) maintains higher structural similarity (ROUGE-L) due to direct sentence extraction.

4.4 Discussion Benchmarking

- **Understanding the Scores:** ROUGE scores theoretically range from 0 to 1, but in practice, "perfect" extraction is impossible due to human subjectivity in summarization.
  - For English news summarization (high-resource), SOTA models often reach ROUGE-1 of **0.45 - 0.50**.
  - For low-resource languages like Romanian, typical scores are in the **0.35 - 0.40** range.
- **Performance Context:** My Hybrid model's score of **0.3439** is highly competitive for the Romanian language, especially given the difficulty of the parliamentary domain (long-context). It aligns with recent benchmarks on Romanian GPT-2 based summarization.
- **Superiority of Filtering:** Both Extractive and Hybrid models significantly outperformed the Baseline (+3.8% ROUGE-1), proving that intelligent content selection is crucial for long documents.
- **Structure vs Content:** The Extractive model scored higher on ROUGE-L (0.25 vs 0.23). This is typical for extractive approaches which preserve original sentence structures perfectly, whereas abstractive models paraphrase, potentially altering word sequences even if the meaning remains correct.
- **Future Improvements (Scaling):** While I adhered to `mT5-small` (300M parameters) due to computational constraints, literature suggests that scaling to **mT5-base** ($\approx 580M$) or **mT5-large** ($\approx 1.2B$) would yield significant improvements in

perplexity and fluency. The current pipeline is model-agnostic and could easily accommodate these larger checkpoints given sufficient GPU memory.

5 Conclusion

This project successfully demonstrates a SOTA-level pipeline for summarizing Romanian parliamentary text. By combining the precision of BERT-based extraction with the fluency of T5-based generation, I solved the context window limitation and achieved a ROUGE-1 score of 0.34, effectively bridging the gap between raw transcripts and official reports.