Sentence Splitting: Encoder-Based vs Decoder-Based Approaches

Archit Rastogi **Ejaz Ahmed**
Sapienza University of Rome
rastogi.1982785@studenti.uniroma1.it ahmed.1988401@studenti.uniroma1.it

1 Introduction

Sentence boundary detection is a token-level classification task: given a sequence of tokens, predict whether each token ends a sentence. We compare two paradigms on Italian literary text (I Promessi Sposi): encoder-based discriminative models that classify token boundaries directly, and decoder-based generative models that predict boundaries via prompting. We evaluate in-domain (validation) and out-of-domain (OOD, Pinocchio) generalization, with F1 as the primary metric given severe class imbalance (96.7% non-boundary tokens).

2 Methodology

2.1 Encoder Approaches

We evaluate three encoder strategies: (1) **BERT fine-tuning**: Italian BERT (Schweter, 2020) with a classification head for binary token prediction; (2) **BERT+CRF**: adding a Conditional Random Field layer to model label dependencies; (3) **XGBoost with embeddings**: gradient boosting (Chen and Guestrin, 2016) on contextual embeddings (Italian BERT, multilingual MiniLM (Wang et al., 2020)) combined with hand-crafted features. We chose fine-tuning over frozen encoders to learn domain-specific boundary patterns in literary Italian.

2.2 Decoder Approaches

For decoders, we test Llama-3.1-8B, Llama-3.2-1B, Llama-3.2-3B (Dubey et al., 2024), and Qwen3-8B (Yang et al., 2025) using inference-only prompting (no fine-tuning). We evaluate seven strategies, focusing on the two best-performing: (1) **Marker Insertion**: the model rewrites text inserting explicit <EOS> markers at sentence boundaries; (2) **Chain-of-Thought**: explicit reasoning over punctuation, context, and syntax before prediction. Other strategies (sliding window, next-token probability, JSON output, few-shot, iterative refinement) performed significantly worse (see Appendix).

2.3 Design Rationale

The fundamental difference is *task-architecture alignment*. Sentence boundary detection is inherently a token-level classification problem. Encoders produce per-token representations that map directly to binary labels no parsing, no ambiguity. Decoders must "translate" classification into text generation, introducing *output format brittleness*: any formatting error, hallucination, or marker misalignment breaks the prediction.

Additionally, encoders were **fine-tuned** on training data, learning Manzoni's specific patterns (dialogue markers, literary punctuation). Decoders used **zero/few-shot prompting** without parameter updates, relying entirely on pre-existing knowledge that may not transfer to this domain.

3 Experiments

Dataset. The Manzoni dataset contains 74,765 training tokens (2,447 boundaries, 3.3%) and 9,344 validation tokens (324 boundaries, 3.5%). OOD evaluation uses Pinocchio (1,524 tokens). Class imbalance makes F1 the primary metric.

Training. Encoders: AdamW optimizer, learning rates 1e-5 to 1e-4 (Optuna-tuned), batch size 8–32, early stopping, FP16. XGBoost: max depth 10, 500 estimators, 384-dim embeddings + hand-crafted features. Decoders: inference-only with chunked input (50–100 tokens) to fit context windows.

4 Results

Table 1 shows OOD results. The best encoder (XGBoost+Italian-BERT, 0.960 F1) outperforms the best decoder (Llama-3.1-8B Marker Insertion, 0.837 F1) by 12.3 points absolute. This gap is consistent: even the weakest fine-tuned encoder (BERT-Italian, 0.678) approaches the best decoder.

Why encoders win. Two factors explain the gap. First, *task-architecture alignment*: encoders output

Family	Model	OOD F1
Encoder (fine-tuned)		
	XGB+Italian-BERT	**0.960**
	XGB+MiniLM	0.946
	XLM-RoBERTa+CRF	0.866
	BERT-Italian	0.678
Decoder (inference-only)		
	Llama-3.1-8B + Marker	**0.837**
	Llama-3.1-8B + CoT	0.503
	Llama-3.2-1B + CoT	0.418
	Llama-3.1-8B + JSON	0.472

Table 1: OOD (Pinocchio) F1 scores. Encoders significantly outperform decoders. Best encoder: XGBoost+Italian-BERT (0.960); best decoder: Llama-3.1-8B with Marker Insertion (0.837).

per-token logits directly, while decoders must generate structured text that requires parsing. Marker Insertion works best among decoders precisely because <EOS> markers are easy to extract, minimizing parsing errors. Second, *fine-tuning*: encoders learned domain-specific patterns (Italian literary conventions, Manzoni's style) that zero-shot prompting cannot capture.

Decoder strategy analysis. Marker Insertion (0.837) dramatically outperforms other strategies. Chain-of-Thought (0.503) suffers from verbose reasoning that increases error opportunities. JSON output (0.472) is brittle to formatting. Sliding window and few-shot essentially fail (F1 $\approx$ 0), indicating that small models cannot reliably perform token-level classification via prompting.

Embedding quality matters. Among encoders, XGBoost+Italian-BERT (0.960) outperforms XGBoost+MiniLM (0.946), confirming domain-specific embeddings generalize better. TF-IDF completely fails (0.017), proving semantic context is essential.

Error patterns. Decoders systematically underpredict boundaries (low recall) in literary contexts with unconventional punctuation. Encoders maintain balanced precision/recall (0.963/0.957 for XGBoost+Italian-BERT). The CRF layer helps in-domain but hurts OOD, learning dataset-specific transitions.

5 Conclusion

For sentence boundary detection, encoder-based approaches decisively outperform decoder-based prompting (0.960 vs 0.837 F1). The gap stems from task-architecture alignment: classification tasks naturally fit encoder per-token predictions, while decoders introduce output format brittleness. Fine-tuning on domain data provides additional gains that inference-only prompting cannot match. For practitioners, we recommend embedding-based classifiers (XGBoost+BERT) for robustness and efficiency, reserving decoder approaches for scenarios where fine-tuning is infeasible.

References

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.

Abhimanyu Dubey, Aniket Jauhri, Abhishek Pandey, Abhinav Kadian, Ahmad Al-Dahle, Anton Letman, Arthur Li, Arjun Agrawal, Anna Wang, Anthony O'Horo, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Stefan Schweter. 2020. Italian BERT and ELECTRA models.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2140–2151.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.