University of Greenwich

School of Computing and Mathematical Sciences

Rare-Word Retention in Abstractive Summarisation

Sara Hodaei

Student ID: 001421104-1

A dissertation submitted in partial fulfilment of the requirements
for the degree of Master of Science in Data Science

Supervisor: Dr. Punitha Puttuswamy

Word Count: 13,753

September 2025

Acknowledgements

I would first like to express my sincere gratitude to my supervisor, Dr. Punitha Puttuswamy for her invaluable guidance, encouragement, and unwavering support throughout this project. Her thoughtful advice and steady mentorship have been instrumental in shaping this dissertation. I am also thankful to Dr. Atif Siddiqui for his valuable input during this important stage of my academic journey.

I am deeply grateful to my dearest mother, who has been my strongest source of motivation, and to my caring father, whose unwavering support has been my greatest strength. I also thank my little brother, for always standing by my side.

A very special thanks goes to Emilis Voropajevas, my partner, for his constant support and encouragement. He consistently pushed me to give my best and guided me forward with his help throughout this project. I am also grateful to my dear friend Aylin Jasas, who shared every step of this journey with me. Both, as fellow students at the University of Greenwich, made this experience truly rewarding.

Abstract

Abstractive summarisation has advanced rapidly with the advent of Transformer-based architectures, yet two persistent challenges remain: hallucination, where models generate unsupported facts, and rare-word omission, where critical entities such as names or numbers are excluded or misrepresented. This study addresses these issues through a hybrid copy-aware Transformer. A BART-base backbone was augmented during training with a pointer generator mechanism and coverage loss, enabling the model to balance generation with reliable copying of rare surface forms. At inference, span-aware continuation heuristics inspired by CopyNext and SeqCopyNet were introduced to reduce fragmented copying, and candidate outputs were reranked to privilege entity faithfulness.

The system was evaluated on the CNN/DailyMail corpus using both content-overlap metrics (ROUGE-1/2/L) and entity-level measures (precision, recall, F1, and UCER). Results show that the hybrid model achieves modest but consistent gains in ROUGE and significantly improves entity precision and recall, while reducing hallucination as measured by UCER. In particular, at 6k training steps, the hybrid system reduced UCER by over 80% relative to the baseline, demonstrating safer factuality without sacrificing fluency.

This dissertation makes three contributions: (i) integrating pointer–coverage into a pre-trained Transformer in a lightweight manner, (ii) enforcing span-aware copying at decoding without additional training cost, and (iii) reframing evaluation around entity metrics to better capture factual fidelity. The findings confirm that combining copy mechanisms with pre-trained generative models provides a practical and extensible path to more faithful abstractive summarisation.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background and Motivation

Automatic text summarisation aims to produce concise, coherent renditions of long documents while preserving their essential meaning (Mani, 2001). In current research, automatic summarisation is a recognised problem in natural language processing (NLP) and has been studied extensively for decades as a benchmark for evaluating models' ability to capture meaning, condense information and generate coherent text (Mani, 2001; Nenkova and McKeown, 2012; Allahyari et al., 2017). It is most commonly framed as a supervised sequence-to sequence learning task, in which models are trained on large datasets of document-summary pairs to learn mappings from long-form text to shorter abstractive renditions. Benchmark corpora such as the XSum dataset (Narayan et al., 2018) and the Gigaword headline generation dataset (Rush et al., 2015) have been widely adopted in this line of research, supporting the development of models ranging from recurrent encoder–decoders (Nallapati et al., 2016) to Transformer-based architectures such as BART (Lewis et al., 2020) and PEGASUS (Zhang et al., 2020). Among these resources, the CNN/DailyMail corpus (Hermann et al., 2015; See et al., 2017) is another additional corpus used as a benchmark for multi-sentence abstractive summarisation, and it forms the primary dataset employed in this study.

The evolution of summarisation models reflects broader advances in machine learning for language (i.e., the application of statistical and neural methods to learn patterns, structure and meaning from text data (Jurafsky and Martin, 2023)). Early extractive approaches focused on hand-crafted heuristics such as word frequency, cue phrases, and sentence position (Luhn, 1958; Edmundson, 1969), followed by graph-based centrality methods such as LexRank and TextRank (Erkan and Radev, 2004; Mihalcea and Tarau, 2004). Although effective at highlighting salient sentences, extractive methods are limited to verbatim extraction and are unable to paraphrase or generalise beyond the source text. In particular, summaries produced by extractive approaches "do not have the same lexical flow or coherence as summaries manually produced by humans" (Giarelis et al., 2023).

This began to change with the rise of neural sequence-to-sequence architectures. Initially developed for machine translation, recurrent encoder–decoders with attention (Sutskever et al., 2014; Bahdanau et al., 2015) were soon adapted to summarisation (Nallapati et al., 2016), enabling abstractive systems that could generate novel phrasing. An early example is the attention-based encoder–decoder of Rush et al. (2015), which applied neural sequence-to-sequence modelling to the Gigaword headline generation task. Their model demonstrated, for the first time, that neural networks could learn to compress full sentences into fluent, abstractive headlines rather than relying solely on extraction, producing paraphrases that generalised beyond the source text.

The subsequent introduction of the Transformer (Vaswani et al., 2017) enabled more efficient parallel training and better modelling of long range dependencies through multi-head self-attention, overcoming the vanishing gradient and sequential bottlenecks that limited recurrent architectures(Vaswani et al., 2017; Bahdanau et al., 2015). This innovation led to powerful pre trained models such as BART (Lewis et al., 2020) and PEGASUS (Zhang et al., 2020), which significantly outperformed earlier recurrent systems on summarisation benchmarks (Raffel et al., 2020).

More recently, large language models (LLMs) such as GPT-3 (Brown et al., 2020) and GPT-4 have demonstrated the ability to perform summarisation in a zero-shot setting, producing summaries without any task-specific fine-tuning and in a few-shot setting, where only a handful of in-context (Brown et al., 2020; OpenAI, 2023). Unlike task-specific encoder–decoder models trained directly on summarisation datasets, LLMs are trained on massive general corpora using language modelling objectives and can adapt to summarisation purely from instructions or a handful of examples. This capability illustrates how summarisation has become a central evaluation point for generative modelling more broadly, summarisation tests whether models can not only generate coherent text but also condense information faithfully, a skill that generalises across domains. Moreover, the persistence of issues such as factual inconsistency and hallucination in LLM outputs shows that summarisation remains a valuable benchmark for probing the strengths and limitations of generative language models at scale (Maynez et al., 2020; Pagnoni et al., 2021).

However, despite these advances, two persistent challenges remain: (i) hallucination, where models generate unsupported or fabricated information, and (ii) rare-word and entity omission, where critical surface forms such as names, numbers and locations are omitted or transcribed incorrectly (Maynez et al., 2020; Pagnoni et al., 2021). The present study focuses on the news domain, as news articles are particularly dense in named entities (PERSON, ORG, GPE) whose accurate reproduction is critical for preserving factual fidelity (See et al., 2017).

This motivates the present work of exploring copy-aware extensions of Transformer summarisation through practical implementation, specifically by using a pre-trained BART-base model with a pointer–generator mechanism and coverage regularisation during training (See et al., 2017), and incorporating span-aware decoding heuristics inspired by SeqCopyNet (Zhou et al., 2018) and CopyNext (Singh et al., 2020). The overarching aim is to retain the fluency and abstraction of a Transformer-based summariser while improving factual precision and minimising hallucination. Having established this motivation, the structure of the dissertation is outlined below.

1.2 Roadmap of the Dissertation

The remainder of this dissertation is structured as follows:

- **Chapter 2** reviews related work on extractive and abstractive summarisation, copy mechanisms, hallucination, and evaluation metrics.
- **Chapter 3** describes the dataset and preprocessing pipeline, including cleaning, deduplication, and tokenisation.
- **Chapter 4** presents the methodology, including architecture, training protocol, decoding procedures, and efficiency strategies.
- **Chapter 5** reports evaluation results with ROUGE and entity metrics, along with length profiles and qualitative analyses.
- **Chapter 6** concludes with key findings, limitations, and future research directions.

2 Related Work

Research in summarisation has developed in close alignment with the evolution of evaluation methods. Summarisation systems generate open-ended text so progress is difficult to measure directly (Nenkova and McKeown, 2011). The field has therefore relied on automatic metrics both to benchmark system performance and to guide model development. Early work standardised around ROUGE (Lin, 2004), which emphasises lexical overlap with reference summaries and remains the most effective benchmark for system comparison. However, subsequent analyses have shown that ROUGE is limited in capturing semantic adequacy or factual consistency (Fabbri et al., 2021). In response, newer approaches incorporate entity-level metrics (Maynez et al., 2020; Pagnoni et al., 2021) that evaluate whether names, organisations, places, and numbers are faithfully reproduced. An aspect that is particularly important in domains such as news, where factual accuracy of entities is central to meaning.

This chapter reviews the landscape of related work. Section 5.1 introduces the evaluation metrics used in this study, tracing their origins and motivations. We then turn to developments in extractive and abstractive summarisation in Section 2.2, followed by copy mechanisms and approaches explicitly designed to mitigate hallucination within this study (Section 2.3).

2.1 Metrics

Evaluation in summarisation has historically relied on automatic metrics designed to quantify how closely a system-generated summary matches a human-written reference. Because summarisation is an open-ended generation task, evaluation remains non-trivial: the same article may admit multiple valid summaries with different wording (Nenkova and McKeown, 2011; Fabbri et al., 2021). Metrics therefore fall broadly into two categories: (i) content-overlap measures, which assess adequacy by comparing lexical or sequence matches with the reference (Lin, 2004; Papineni et al., 2002), and (ii) factuality-oriented measures, which capture whether critical entities are preserved or hallucinated (Maynez et al., 2020; Pagnoni et al., 2021). This study employs both, balancing fluency/coverage with factual precision.

ROUGE

The Recall-Oriented Understudy for Gisting Evaluation (ROUGE) was introduced by Lin (2004) and has since become the de facto standard for automatic summarisation evaluation. ROUGE measures overlap between a candidate summary C and a reference summary R at different granularities. The most widely used variants are:

- **ROUGE-1**: unigram overlap, reflecting basic content coverage.
- **ROUGE-2**: bigram overlap, capturing local coherence.

- **ROUGE-L**: based on the longest common subsequence (LCS), rewarding sequence-level matches.

Formally, for an n-gram set G_n:

$$\text{ROUGE-}n_{\text{recall}} = \frac{|G_n(C) \cap G_n(R)|}{|G_n(R)|}, \quad \text{ROUGE-}n_{\text{precision}} = \frac{|G_n(C) \cap G_n(R)|}{|G_n(C)|}. \tag{2.1}$$

The F1 score balances the two:

$$\text{ROUGE-}n_{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{2.2}$$

For ROUGE-L, the score is based on the length of the longest common subsequence $LCS(C, R)$:

$$\text{ROUGE-L}_{\text{recall}} = \frac{LCS(C,R)}{|R|}, \quad \text{ROUGE-L}_{\text{precision}} = \frac{LCS(C,R)}{|C|}. \tag{2.3}$$

with the F1 score defined analogously.

ROUGE remains attractive due to its interpretability, efficiency, and comparability across decades of work. However, it has been criticised for focusing on surface overlap rather than semantic equivalence, often underestimating the quality of abstractive systems (Fabbri et al., 2021). In this study, ROUGE serves as a benchmark for *content adequacy*, enabling comparability with prior abstractive systems such as BART and PEGASUS (Lewis et al., 2020; Zhang et al., 2020).

Entity Precision, Recall and F1

ROUGE does not directly capture whether the content is factually correct. Entity-level metrics, introduced in factuality evaluations such as Maynez et al. (2020) and formalised in benchmarks like FRANK (Pagnoni et al., 2021), provide a complementary view by explicitly tracking named entities.

Entities are extracted from both candidate and reference using named entity recognition (NER). Let E_C be the set of entities in candidate summary C and E_R the set in reference R. We compute:

$$\text{Entity Precision (entP)} = \frac{|E_C \cap E_R|}{|E_C|}, \tag{2.4}$$

$$\text{Entity Recall (entR)} = \frac{|E_C \cap E_R|}{|E_R|}, \tag{2.5}$$

$$\text{Entity F1 (entF1)} = \frac{2 \cdot \text{entP} \cdot \text{entR}}{\text{entP} + \text{entR}}. \tag{2.6}$$

Entity precision quantifies hallucination avoidance (higher $\Rightarrow$ fewer unsupported entities), entity recall quantifies coverage of reference facts, and entity F1 balances the two. These measures

are particularly relevant in the news domain, where omission or alteration of PERSON, ORG, or GPE entities directly undermines factual fidelity.

Unsupported Core Entity Rate (UCER)

To complement token-level metrics, a system-level measure of factual risk is also employed: the Unsupported Core Entity Rate (UCER). Inspired by factuality audits in summarisation (Maynez et al., 2020), UCER measures the proportion of summaries that introduce at least one unsupported *core entity*—a PERSON, ORG, or GPE that does not appear in the reference:

$$\text{UCER} = \frac{\#\{C \in \mathcal{C} : E_C^{\text{core}} \setminus E_R^{\text{core}} \neq \emptyset\}}{|\mathcal{C}|}, \tag{2.7}$$

where $\mathcal{C}$ is the set of candidate summaries, and E^{core} denotes the set of core entities. Lower UCER indicates safer models: even if average entP is high, a single hallucinated name can render a summary factually unreliable.

Summary

Together, ROUGE and entity-based metrics form a complementary evaluation suite. ROUGE quantifies content adequacy through lexical overlap, while entity precision/recall/F1 and UCER capture factual precision and coverage. This dual perspective directly addresses the challenges motivating this study. Hallucination and rare-entity omission by reflecting not only how much content is preserved but also whether that content is factually correct and reliable (See et al., 2017; Maynez et al., 2020; Pagnoni et al., 2021).

2.2 Related Work

Abstractive summarisation research has long faced the dual challenge of producing fluent, concise text while remaining faithful to the source. Despite sustained progress, factual inconsistency and rare-word omission remain unsolved problems, motivating continued architectural innovation. Early neural approaches, built on recurrent sequence-to-sequence models with attention (Rush et al., 2015; Nallapati et al., 2016) demonstrated that abstractive summaries could be learned directly from data rather than assembled through handcrafted heuristics. These systems produced novel phrasings but often failed when critical low-frequency tokens such as names or numbers were required. Reliance on fixed vocabularies meant that rare surface forms were either replaced with placeholders or paraphrased incorrectly, undermining factual fidelity. Reinforcement learning extensions such as Paulus et al. (2018) sought to stabilise training, but recurrent architectures remained computationally expensive, struggled with long-range dependencies, and never offered a robust solution to the unknown-word problem.

The introduction of the Transformer (Vaswani et al., 2017) replaced recurrent computation with multi-head self-attention, removing sequential bottlenecks and enabling global context mod-

elling. This innovation underpinned a new generation of large-scale pre-trained encoder–decoders such as BART (Lewis et al., 2020), PEGASUS (Zhang et al., 2020) and T5 (Raffel et al., 2020). BART in particular showed that denoising pre-training enabled a single Transformer to generalise effectively across generation tasks. Fine-tuned on CNN/DailyMail, BART produced summaries that were markedly more fluent and coherent than recurrent models. Yet fluency gains were not matched by consistent factual reliability: independent evaluations found hallucinations in around 30% of outputs (Cao et al., 2018; Kryściński et al., 2020), an error rate incompatible with high-stakes domains such as news or law. The architectural leap to Transformers therefore improved surface quality but left unresolved the central tension between abstraction and factuality.

The persistence of factual errors led researchers to revisit and extend copy mechanisms, originally motivated by the need to handle rare words in recurrent models. Vinyals et al. (2015) first introduced the pointer network, in which the decoder does not select a token from a fixed vocabulary but instead outputs an index over the input sequence through an attention distribution. This formulation allowed seq2seq models to copy unseen symbols by directly selecting their position in the source. Building on this foundation, Gu et al. (2016) and Gulcehre et al. (2016)incorporated hybrid architectures that combined a standard generator with a pointer module, enabling the decoder at each step to either produce a token from its vocabulary or copy one from the source. This mixture distribution directly addressed the unknown-word problem, as it meant that rare entities and numerical expressions could be reproduced faithfully even if absent from the model's vocabulary. Merity et al. (2017) refined this approach with the pointer sentinel mixture model, which introduced a sentinel token in the attention distribution to decide when the model should fall back to generation. This provided a more stable mechanism for interpolating between copy and generation probabilities, avoiding degenerate behaviours where the pointer was over- or under-used.

A decisive advance came with the Pointer-Generator Network of See et al. (2017), which formalised the mixture of copy and generation through a gating scalar learned at each decoding step. This gate determined the balance between the standard vocabulary distribution and the attention-based pointer distribution, effectively letting the model decide when to generate novel phrasing and when to extract verbatim tokens. To mitigate the tendency of seq2seq decoders to repeat phrases, they paired this with a coverage mechanism, which accumulates past attention weights and penalises repeated focus on the same source positions. Evaluated on CNN/DailyMail, this pointer-generator with coverage (PGC) became a widely adopted baseline, consistently reducing hallucination rates and better capturing factual details such as names, dates and locations. Importantly, it demonstrated that abstractive models could retain fluency while adopting an explicitly extractive bias. Yet its operation at the token level left limitations unresolved: multi-word entities and phrases were often fragmented, as each token's copy decision was made independently. Thus, while PGC represented a turning point in abstractive summarisation by

integrating copying and coverage into a single, effective framework, it also exposed the need for mechanisms capable of enforcing span-level consistency in copied expressions.

The challenge of fragmented copying stimulated further research into span-level mechanisms that could reproduce contiguous sequences from the source rather than isolated tokens. Zhou et al. (2018)introduced SeqCopyNet, which extends the pointer-generator framework by predicting the start and end boundaries of a span to be copied as a whole. Once a span is selected, the decoder copies it in its entirety, ensuring that multi-word expressions such as named entities or technical terms are preserved intact. To maintain decoder state coherence after multi-token copying, they proposed the CopyRun procedure, which advances the recurrent state over the internal tokens of the span, avoiding the discontinuities that arise when spans are inserted wholesale. Empirically, SeqCopyNet demonstrated substantial gains on sentence summarisation tasks, confirming that span-level modelling alleviates entity fragmentation. However, its architecture introduced significant complexity, requiring specialised span prediction heads and custom training dynamics, and its benefits were less clearly established for longer multi-sentence inputs such as CNN/DailyMail. This raised open questions about how span copying could be integrated with more scalable Transformer architectures without incurring prohibitive training overheads.

Building on the same intuition, Singh et al. (2020) proposed CopyNext, which simplified span copying by introducing an explicit copy-next action into the decoder's decision space. Once a source token was copied, the decoder could remain in a continuation state that automatically advanced to the next source token until copying was terminated. This encoded a contiguity prior: if copying has started, the most probable continuation is the next token in the source. Unlike SeqCopyNet's boundary prediction and CopyRun updates, CopyNext implemented span continuation as a lightweight recurrent decision, avoiding additional modules and reducing complexity. Although primarily evaluated in the context of named entity recognition and structured prediction tasks rather than long-form summarisation, its principle is directly relevant. Without such contiguity bias, pointer-generator models risk copying only fragments—for example, "United" without "States"—resulting in incomplete or misleading entities. CopyNext demonstrated that even simple architectural signals can strongly encourage complete span reproduction. Yet its limited evaluation scope and reliance on training-time modifications meant that its utility in large-scale summarisation, especially within pretrained Transformer backbones, remained under-explored. This gap—between the clear need for span-continuous copying and the difficulty of integrating it into modern summarisation models—provides a key motivation for approaches that enforce span awareness at decoding without altering training regimes, as pursued in this work.

As pre-trained Transformers became dominant, researchers sought to reintroduce copying in ways compatible with their architectures. Xu et al. (2020) proposed Self-Attention Guided

Copy (SAGCopy), which leverages the self-attention graph of the encoder to compute centrality scores for source tokens. These scores bias the copy distribution toward tokens deemed important by the model's internal attention patterns. The motivation is that Transformer self-attention already encodes a notion of salience, and aligning the copy module with this structure should ensure that high-importance tokens such as entities are not overlooked. Experiments on CNN/DailyMail demonstrated gains in ROUGE and factual inclusion, showing that attention-guided copying could complement pretrained encoders. Yet the method remained word-level and did not explicitly address span fragmentation.

Li et al. (2021) introduced the Correlational Copy Network (CoCoNet) to further stabilise copying behaviour by conditioning each copy decision on the history of previous copies. Standard pointer mechanisms treat each time step independently, which can result in incoherent copying of disjoint words. CoCoNet instead models semantic and positional correlations between copied words, encouraging the model to continue copying related words from the source. This design produced more coherent reproduction of multi-word phrases and reduced factual inconsistencies. Importantly, it demonstrated that copy history, what has already been copied—can guide future decisions, an insight directly relevant to entity-level faithfulness.

While these extensions improved copying fidelity, they still permitted models to hallucinate entities not present in the source, since vocabulary generation remained available at every step. Xiao and Carenini (2022) directly confronted this limitation with an Entity-Based SpanCopy mechanism. Their model focuses explicitly on named entities: a span copier trained to replicate entity mentions from the source, gated by a mechanism that decides when to invoke it. This ensures that entities in the summary are drawn directly from the input document, thereby reducing unsupported entity hallucinations. Crucially, their evaluation demonstrated that SpanCopy improved entity-level factual consistency with almost no change in saliency or ROUGE, highlighting that entity-aware design can improve faithfulness without sacrificing coverage. The work also showed that hallucinated entities are not always "wrong" in a semantic sense—for instance, replacing "Portsmouth" with "Hampshire" may seem plausible—but they nonetheless undermine factual alignment, and SpanCopy substantially reduced such errors. This finding reinforces the need to treat entity fidelity as a first-class objective rather than a side effect of general copying.

Parallel to these architectural advances, there has been growing recognition that evaluation metrics shape model development. For years, ROUGE (Lin, 2004) dominated summarisation evaluation, measuring n-gram overlap between system and reference summaries. While convenient, ROUGE correlates poorly with factual consistency, as it rewards lexical overlap regardless of whether details are accurate. Maynez et al. (2020) and Pagnoni et al. (2021) showed that systems with high ROUGE often hallucinate unsupported facts. Consequently, new metrics focusing on factuality have been proposed. Kryściński et al. (2020) introduced FactCC, a BERT-based entailment classifier for detecting contradictions between summary and source. Maynez et al.

(2020) proposed QA-based metrics (QAGS), which generate questions from the summary and test whether the source provides consistent answers. While these metrics provide finer-grained signals, they require auxiliary models and can be brittle. At the entity level, Nan et al. (2021) proposed simple yet effective metrics measuring precision and recall of named entities in system summaries against source and reference texts. Precision against the source reveals hallucinations, while recall against the reference highlights omissions. These entity metrics directly reflect the challenges this project seeks to address: hallucination and rare-entity omission. They are also transparent, interpretable, and align with the real-world requirement that every entity in a news summary must be supported by the source.

The emphasis on entity-level evaluation reflects a broader shift in current thinking. Researchers increasingly recognise that summarisation quality cannot be fully captured by overlap metrics, and that faithfulness must be explicitly measured and optimised. However, evaluation alone does not resolve the architectural gap. Even when entity metrics are applied, models like pointer-generator networks tend to fragment multi-word names, while large pretrained Transformers without copy mechanisms continue to hallucinate. This reveals a systemic limitation: existing architectures either copy too narrowly (word-by-word without span control) or generate too freely (risking unsupported entities).

Data quality has also emerged as a limiting factor. Lee et al. (2022) showed that large pre-training corpora contain substantial duplication, encouraging memorisation and plausible but unsupported generations. Lv et al. (2024) further observed that Transformers acquire reliable copying ability only late in training, in a "grokking"-like dynamic, long after pre-training loss appears converged. This suggests that relying solely on implicit copying capabilities of large models is risky: without explicit mechanisms, entity copying may emerge too late or too inconsistently for reliable use.

Taken together, these strands of research illustrate both the progress and the gaps in abstractive summarisation. From recurrent seq2seq to pre-trained Transformers, from token-level pointers to entity-aware span copiers, the field has steadily advanced toward models that balance fluency and faithfulness. Yet each solution leaves open issues. Pointer-generator networks reduce hallucinations but fragment spans. SeqCopyNet and CopyNext mitigate fragmentation but add architectural complexity or are not tailored to large-scale summarisation. SAGCopy and CoCoNet integrate with Transformers but remain word-level and do not guarantee entity accuracy. Entity-Based SpanCopy directly enforces entity fidelity but requires additional modules and training costs. Pretrained Transformers deliver fluency but, without constraints, continue to hallucinate. Evaluation metrics now expose these limitations more clearly than ROUGE alone, but they also highlight the absence of a model that simultaneously addresses span completeness, entity fidelity and abstractive quality.

This analysis establishes the gap motivating the present work. Existing approaches either com-

promise on factual consistency, require substantial retraining, or fail to integrate span-aware copying with powerful pretrained Transformers in a cost-effective manner. The design adopted here—augmenting a BART-base summariser with pointer-generator and coverage mechanisms during training, while imposing span-aware constraints only at decoding—directly responds to these limitations. It builds on evidence that copying improves entity fidelity (See et al., 2017; Zhou et al., 2018; Singh et al., 2020; Xiao and Carenini, 2022) but diverges by placing span continuation in the inference stage, avoiding additional training complexity. By doing so, this work seeks to retain the fluency of BART while addressing the most pressing unresolved issue in abstractive summarisation: the faithful retention of rare and entity-level details.

Table 2.1: Neural summarisation models and copy mechanisms (from token to span/entity level).

Model / family	**Backbone**	**Copy / faithfulness mechanism**	**Key takeaways (per your review)**
Early seq2seq + attention (Rush et al., 2015; Nallapati et al., 2016)	RNN (attentive encoder–decoder)	None (fixed vocab)	Fluent but struggles with rare names/numbers; unknown-word issue persists.
Pointer-Generator (PGC) (See et al., 2017)	RNN	Token-level pointer + coverage	Big drop in hallucinations; better entities/dates/locations, but fragments multi-word entities.
SeqCopyNet (Zhou et al., 2018)	RNN	Span start/end + CopyRun state updates	Copies contiguous spans intact; more complex; evidence strongest on short/single-sentence inputs.
CopyNext (Singh et al., 2020)	RNN/seq models	"Copy-next" continuation action (span contiguity prior)	Lightweight span continuation; reduces fragmentation; limited large-scale summarisation evals.
SAGCopy (Xu et al., 2020)	Transformer-compatible	Self-attention-guided copy salience	Improves ROUGE/entity inclusion vs. plain Transformers; still word-level, not span-aware.
CoCoNet (Li et al., 2021)	Transformer-compatible	Copy decisions conditioned on copy history	Encourages coherent multi-word reproduction; lowers inconsistencies; adds training complexity.
Entity-based SpanCopy (Xiao and Carenini, 2022)	Transformer-compatible	Entity-focused span copier + gate	Cuts unsupported entity hallucinations with minimal ROUGE change; extra module/training cost.
BART / PEGASUS / T5 (Lewis et al., 2020; Zhang et al., 2020; Raffel et al., 2020)	Transformer (pretrained)	None by default (generation-only)	Strong fluency/coherence; still hallucinate without copy/constraints; basis for decoding-time span control.

Table 2.2: Factuality concerns, metrics, and data-quality factors shaping model behaviour.

Theme	Representative work	Implications from your review
Limitations of ROUGE overlap	(Lin, 2004; Maynez et al., 2020; Pagnoni et al., 2021)	ROUGE rewards lexical overlap but correlates weakly with factual consistency; models can be fluent yet hallucinate.
Entity-level evaluation focus	(Pagnoni et al., 2021; Maynez et al., 2020)	Precision/recall of entities is transparent and aligns with the need to avoid unsupported entities and omissions.
Transformer fluency vs. faithfulness	(Vaswani et al., 2017; Lewis et al., 2020; Zhang et al., 2020; Raffel et al., 2020)	Pretrained encoder–decoders greatly improve fluency/coherence but still hallucinate without copy/constraints.
Copying to reduce hallucinations	(See et al., 2017; Zhou et al., 2018; Singh et al., 2020; Xiao and Carenini, 2022)	Token-level pointers reduce errors but fragment spans; span-aware/entity-aware methods better preserve names/phrases.
Data duplication & memorisation risks	(Lee et al., 2022)	Duplicate pretraining data encourages plausible but unsupported generations; cleaning improves reliability.
Late-emerging copy competence ("grokking")	(Lv et al., 2024)	Reliable copying may emerge late in training; relying on implicit copying alone is risky without explicit constraints.

2.3 This Work

The review of prior research highlights a clear gap: no single model simultaneously secures the fluency of large-scale pre-trained Transformers and the entity-level fidelity delivered by copying mechanisms, while doing so in a manner that remains computationally efficient. This work directly responds to that gap through the design of a hybrid summariser that integrates a pointer–generator and coverage mechanism into a BART-base model during training, while imposing span-aware constraints at decoding. The approach is deliberately modular. It seeks not to replicate the full architectural complexity of SeqCopyNet or CoCoNet, which require custom span predictors and additional parameters, but rather to leverage BART's strengths as a fluent generator and layer on targeted mechanisms where they matter most: the retention of rare and entity-level details.

The central design choice is to couple the pre-trained BART encoder–decoder with a pointer generator network. This augmentation reintroduces the ability to directly copy from the source, addressing the issue of rare word and entity omission that continues to affect Transformer baselines. Following (See et al., 2017), the pointer–generator interpolates between the decoder's vocabulary distribution and a copy distribution derived from encoder attention, governed by a learned gate. By retaining the coverage penalty originally proposed to reduce repetition, the model also gains robustness against one of the common failure modes of abstractive generation. The result is an encoder–decoder that can still paraphrase and compress through its generative channel, but which now has an explicit extractive pathway to ensure critical surface forms are preserved. Unlike early RNN-based implementations, the hybridisation is performed within a modern pre-trained backbone, ensuring that gains in factual fidelity are not offset by losses in fluency.

Where this work diverges from most previous extensions is in its treatment of span-level copying. Instead of introducing additional span prediction modules, as in SeqCopyNet (Zhou et al., 2018) or CopyNext (Singh et al., 2020), span awareness is deferred entirely to the decoding

stage. The decoding process monitors candidate outputs for partial matches against source entities and enforces continuation until the span is complete, effectively biasing beam search to maintain entity integrity. This decision was motivated by both practical and theoretical considerations. From a practical perspective, inference-time span constraints avoid the overhead of modifying training objectives and parameters, making the approach lighter and more reproducible. From a theoretical perspective, constraining the decoder rather than retraining the encoder allows the model to remain maximally fluent while still being prevented from producing fragmented or unsupported entities. In this respect, the method aligns with the objectives of CopyNext, which sought to encode contiguity priors, but achieves them without requiring structural changes to the model or additional loss terms.

Further, this work prioritises entity fidelity even where it diverges from maximising ROUGE. Prior literature often reported improvements primarily in overlap metrics, but evaluations such as those of (Nan et al., 2021) demonstrate that high ROUGE does not imply factual reliability. In the present design, checkpoint selection and model validation explicitly used entity precision, recall and F1 alongside UCER as criteria, ensuring that the final model optimised for factual correctness. This reflects a deliberate departure from the convention of selecting models solely on ROUGE. The strategy accepts that abstractive systems may sacrifice marginal overlap gains if doing so prevents hallucination of unsupported entities, and it reframes evaluation priorities around reliability rather than surface similarity.

A further point of comparison is with the Entity-Based SpanCopy model of (Xiao and Carenini, 2022), which is conceptually close to this work in its focus on reducing entity hallucination. Their system introduces a dedicated span copier and gating module at training time, and additionally relies on a filtered dataset where reference summaries share entities with the source, thereby reducing the frequency of unsupported supervision. By contrast, the present study retains the pointer–generator and coverage as its only training-time augmentation, and does not require dataset curation. Instead, span fidelity is imposed entirely at decoding through constrained continuation of entity mentions. This distinction highlights the novelty of the approach: it demonstrates improvements in entity-level faithfulness without either architectural expansion or reliance on filtered supervision, making it more lightweight, generalisable, and applicable to real-world noisy datasets such as CNN/DailyMail.

Complementing the pointer–generator and span-aware decoding is a reranking strategy informed by the observation that Transformer decoders often produce a mixture of outputs of varying fidelity. Drawing inspiration from reranking methods explored by (Falke et al., 2019) and others, this work integrates diverse beam search with a lightweight reranker. Diverse beam search expands the candidate space to include summaries with varied phrasings and entity choices, while the reranker selects the candidate least likely to contain unsupported entities. By privileging factual correctness in reranking, the decoding pipeline reduces the risk that the

final output contains hallucinated content. Importantly, this is achieved without invoking heavy external models such as entailment classifiers or QA modules, preserving computational efficiency.

Together, these elements form a system that is distinct from prior work in several respects. Unlike pointer–generator implementations in recurrent architectures, the model leverages the linguistic fluency and contextual breadth of BART. Unlike SeqCopyNet and CopyNext, span consistency is imposed post hoc at inference rather than learned through additional modules. Unlike SAGCopy and CoCoNet, the design avoids training-time modifications, sidestepping the need for new centrality estimators or correlation parameters. Unlike SpanCopy, the method does not require an entity recognition component during training, though it achieves similar improvements in entity fidelity by applying constraints at decoding. In combining these insights, the system reflects an integration of architectural pragmatism and empirical priorities: it adapts proven mechanisms from the literature, applies them where they are most impactful, and measures success by factual precision rather than lexical overlap alone.

The novelty of this work lies not in proposing an entirely new copying algorithm, but in demonstrating how existing mechanisms can be strategically combined within a pre-trained Transformer to resolve the particular weaknesses of current summarisation systems. The adoption of pointer–generator and coverage restores reliable copying of rare words, span-aware decoding prevents fragmentation and hallucination of entities, and reranking ensures that the final output is selected for factual consistency. Each component directly addresses a gap identified in Section 2.2: the token-level limitation of PGC, the training complexity of span-copying networks, and the factual unreliability of unconstrained Transformer decoders. The resulting hybrid summariser retains the fluency of BART while achieving substantially improved entity-level faithfulness, positioning it as a practical contribution to the ongoing challenge of abstractive summarisation.

3 Corpus and Data Preparation

3.1 Dataset Description and Licensing

This study employs the non-anonymised CNN/DailyMail article–highlights corpus, originally introduced for machine reading comprehension with anonymised entities (Hermann et al., 2015) and later popularised for abstractive summarisation in its non-anonymised article–highlights form (See et al., 2017). Each example comprises a full news article and professionally written highlights used as the reference summary, giving an ideal pairing for training and evaluating abstractive systems. Copy-aware methods on this dataset are motivated by the need to handle names, numbers and other surface forms that are difficult to generate purely from a fixed vocabulary (See et al., 2017).

The news domain is entity-dense (persons, organizations, places, dates, quantities), which places particular demands on a model's ability to handle rare or out-of-vocabulary tokens, an area where copy-aware mechanisms are particularly effective (See et al., 2017). The corpus has been widely adopted by copy-aware Transformer variants, including models that integrate self-attention guided copying (Xu et al., 2020) and span-based copying strategies such as CopyNext (Singh et al., 2020). It has also supported models that track copying history (Li et al., 2021) and training-strategy work such as noisy self-knowledge distillation (Liu et al., 2021). Collectively, these studies establish CNN/DailyMail as solid benchmark for evaluating abstractive, copy-aware methods directly aligning with the aims of this project.

By contrast, XSum has emerged as another widely used summarisation benchmark, and results for models such as BART and PEGASUS are routinely reported on the XSum test set (Lewis et al., 2020; Zhang et al., 2020; Rothe et al., 2021). However, XSum's targets are single sentence, highly abstractive summaries, described by (Narayan et al., 2018) as capturing the 'aboutness' of the document, a design that minimises source overlap but also reduces opportunities for copy-aware behaviour. As a result, models trained and evaluated on XSum showcase less opportunity and weaker pressure for copy-aware behaviour. Studies also report substantially higher factual error/hallucination rates on XSum than on multi-sentence datasets such as CNN/DailyMail (Maynez et al., 2020; Pagnoni et al., 2021). Given the present focus on rare-entity retention and copy/coverage analysis, XSum was therefore not adopted. Instead, CNN/DailyMail's multi-sentence, editor written highlights provide higher lexical/entity overlap and thus a stronger benchmark for copy mechanisms aligned with the aims of this study.

The study uses version 3.0.0 with the official splits (Train = 287,113; Validation = 13,368; Test = 11,490). Articles originate from professionally produced journalism from the late-2000s to mid-2010s; licensing/attribution requirements and typical topic/geographic biases of news corpora are acknowledged and addressed in Section 3.5.

3.2 Pre-processing Pipeline

Having established the dataset and scope, a conservative pre-processing pipeline was applied to reduce noise and leakage while preserving the linguistic content used for training and evaluation. The protocol comprised the four following elements executed in a fixed linear order:

- Structural normalisation of the raw text.
- Removal of exact and near-duplicates including cross-split overlaps.
- Excision of boilerplate such as bylines, update stamps and credit lines.
- Removal of outlet/source tags where present.

The rationale was to minimise spurious variation and prevent train evaluation contamination while leaving genuine editorial signal intact. All the aforementioned steps were employed based on the insights gained from the train set only, and then applied on the validation and test set.

Structural Normalisation (Artifacts, Unicode, White-Space/Punctuation)

Structural cleaning was applied prior to any duplicate screening to standardise both the articles and highlights while preserving meaning. All line breaks were converted to spaces, Unicode was normalised (NFKC) with zero-width and non-breaking spaces being removed. Residual HTML was unescaped and tags were stripped. Legacy tokenisation artefacts were corrected (e.g., "@-@" to "-") (See et al., 2017). Numerical artefacts were repaired (e.g., 1,000 to 1000) and malformed punctuation collapsed (runs of "?!/.,;:" reduced to a single mark). Finally, white-spaces were standardised to single spaces with trailing spaces removed, producing a clean, canonical text representation for subsequent deduplication and boilerplate/source-tag passes.

Exact and Near-Duplicate Removal (Method and Thresholds)

Exact duplicates were identified (before and after structural cleaning) at the article–highlight pair level. Text was first converted into a canonical form, ensuring consistent white-space while retaining the original casing. Then a stable hash was computed over the concatenated article and highlights, with the first occurrence being retained and subsequent identical pairs removed to avoid over-representation of a single instance. Using this policy, 3098 exact article–highlight duplicates were removed from the training split, none from validation, and two from test. This step reduced the risk of rote memorisation and prevented repeated strings from exerting disproportionate influence during optimisation (Lee et al., 2022).

Conflicting pairs instances where the same article appeared with different highlights were identified in the training split (n = 10). Each case was reviewed and variants with trivial edits or factual inconsistencies were dropped to avoid conflicting supervision and superficial memorisation. The rejected differences fell into two types: (i) factual errors (e.g., "32" vs "33" years) and

(ii) near-duplicate repetitions that added no new information relative to the retained highlight.

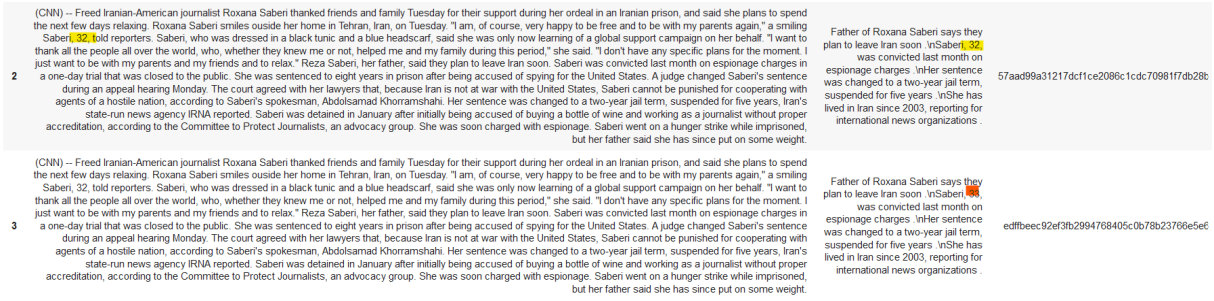

Figure 3.1: Conflicting pair: identical article with highlights differing only in age (32 vs. 33). The source-consistent variant was retained, while the inconsistent variant was removed.

Cross-split overlap was checked using the same procedure. One verbatim article was found to appear in both the training and validation splits. To avoid train–evaluation leakage—known to inflate reported performance, the training copy was removed while the validation instance was preserved, ensuring that the held-out distribution remained untouched and that validation retained its role as an unbiased measure of generalisation (Maynez et al., 2020; Carlini et al., 2021).

To reduce memorisation and leakage while preserving natural phrasing, near-duplicates were removed using a two-stage process grounded in MinHash/LSH from training set articles (Broder, 1997; Indyk and Motwani, 1998). Text was first normalised (Unicode NFKC, white-space collapsed), with punctuation and case preserved to avoid damaging named entities and predictable news phrasing. Exact duplicate lines were removed before approximate matching.

Documents were then segmented into overlapping 7-token n-grams. 128-permutation MinHash sketches were computed and indexed with locality-sensitive hashing (LSH threshold 0.95) to efficiently retrieve high similarity candidates. Candidates were confirmed only if a secondary token-sort similarity reached $\geq 96/100$ (tolerant of minor punctuation and spacing differences). Short texts ($n < 7$ tokens) were handled with a conservative fallback (no approximate match). The pass ran in 50k-row blocks (progress logging every 20k rows), retaining stop-words to reflect real natural phrasing.

A fixed, non-cryptographic 64-bit hash with a constant seed ensured reproducibility. This design follows standard MinHash-which sketches Jaccard set similarity (Broder, 1997; Indyk and Motwani, 1998; Lee et al., 2022) and locality-sensitive hashing for approximate neighbour retrieval (Indyk and Motwani, 1998), and it is motivated by evidence that deduplication reduces memorised copying and improves evaluation fidelity (Lee et al., 2022).

Overall, 126 training items were detected by the near-duplicate screen and recorded in the decisions log. Manual review indicated no fully paraphrastic duplicates; the matches reflected

recurring outlet templates, reused sentence stems, and minor stylistic variations rather than distinct stories being paraphrased. Therefore, all items were removed.

Boilerplate Removal

Boilerplate refers to repeatedly reused, non-semantic strings at article edges—navigation prompts, copyright notices, disclaimers, and templated lines—often treated as unwanted repeated segments that distract from the core content of news texts (Bose, 2019). Removing such text does not harm learning because it carries no narrative or factual content, rarely appears in the editor-written highlights, and otherwise inflates token frequencies with irrelevant patterns. Boilerplate is widely considered linguistically unattractive and noise in corpora creation (Schäfer, 2016).

Evidence from deduplication studies shows that near-duplicate and long repetitive substrings bias models toward memorisation, which supports removing predictable template strings to protect generalisation (Lee et al., 2022). Reducing repetitive boilerplate reduces the chance of copying unhelpful patterns and allows the copying signal to focus on contentful tokens (See et al., 2017; Gu et al., 2016).

The detector scanned the first and last three sentences of each article (sentences shorter than five characters were ignored) and, in parallel, mined edge word n-grams of length 3–6 (trigrams to six-grams). Long stock lines were flagged by the sentence channel, while shorter templated fragments were captured by the n-gram channel. The detailed boilerplate/source-tag lexicon, regex rules, and removal counts are provided in Appendix **??** (Tables A.1–3.3).

Source-tag removal

A source tag is a short outlet marker such as "(CNN)" embedded at the start of an article or inline. It was not detected by the boilerplate pass since source tags tend to appear inline and are often glued to punctuation, falling outside those channels.

A separate rule-based pass therefore removed "(CNN)" tokens at the beginning of articles and when they appeared mid-sentence, together with any immediately attached punctuation (e.g., "(CNN)The …", "(CNN)–Officials …"). An exception preserved an immediately following quotation mark to keep punctuation well-formed (e.g., "(CNN)'It was…'" → "'It was…'"). Since near-duplicate detection relies on contiguous token overlap, even a single extra token such as "(CNN)" shifts alignment and prevents otherwise identical articles from meeting the similarity threshold; removing the tag restores the true content overlap and lets near-duplicates be detected correctly.

3.3 Descriptive Statistics & EDA

The cleaning pipeline left the evaluation splits intact and reduced the training and test splits only through duplicate handling (Table 3.1). As shown in Figure 3.2 (token lengths estimated via whitespace segmentation on NFKC-normalised text) and Figure 3.3 (rare-word counts), the distributions before and after cleaning are essentially indistinguishable, indicating that preprocessing did not compress articles or deplete rare tokens. Figure 3.4 shows that entity frequencies (PERSON, ORG, GPE, etc.) extracted using spaCy were also preserved, so the entity-rich character that motivates copy-aware modelling remains intact.

Table 3.1: Dataset sizes before and after cleaning. Values show number of article–highlight pairs.

Split	Original	Cleaned
Train	(287113, 3)	(283861, 3)
Validation	(13368, 3)	(13368, 3)
Test	(11490, 3)	(11488, 3)

Using a 50k random sample is enough to show the true curves for token length and rare-word counts without wasting compute. The Dvoretzky–Kiefer–Wolfowitz (DKW) inequality states that the empirical distribution from n samples stays uniformly within ε of the full-data distribution with very high probability (Dvoretzky et al., 1956):

$$P\left(\sup_x \left|F_n(x) - F(x)\right| > \varepsilon\right) \leq 2e^{-2n\varepsilon^2}. \tag{3.1}$$

For $n = 50{,}000$ and $\varepsilon = 0.01$:

$$P\left(\sup_x |F_n - F| > 0.01\right) \leq 2e^{-2\cdot 50{,}000\cdot 0.01^2} \tag{3.2}$$

$$= 2e^{-10} \tag{3.3}$$

$$\approx 9.08 \times 10^{-5}. \tag{3.4}$$

Thus, the confidence is:

$$\text{Confidence} \;\approx\; 1 - 9.08 \times 10^{-5} \;=\; 0.99991 \;(\approx 99.99\%). \tag{3.5}$$

Therefore, plotting all 287k items would not materially change the figures, but would add unnecessary cost and latency.

To quantify cross-split phrase reuse and diversity, the Jaccard overlap of unique 3-grams and 4-grams was measured between splits (Figure 3.5). Jaccard overlap $|A \cap B|/|A \cup B|$ shows what fraction of phrases two splits share. Three-word sequences capture stock news phrasing that

mixes function and content words, while four-word sequences are more specific mini-templates. Unigrams and bigrams are too generic, while higher-order n-grams are too sparse and mostly reflect near-verbatim reuse. Lower overlap signals greater lexical/phrastic diversity (harder to "win" by rote reuse), whereas some overlap is desirable so models must decide when to copy a familiar fragment and when to generate novel text—exactly the copy/coverage behaviour this project evaluates. The cleaned corpus still contains enough shared phrasing for copy mechanisms to operate realistically, yet plenty of novel four-word spans to require genuine abstraction and entity handling (see Figure 3.5).

Table 3.2: Cross-split Jaccard overlap of unique n-grams (before $\rightarrow$ after). Values show fraction of shared unique phrases between splits.

Split pair	**3-gram**	**4-gram**
Train–Validation	0.254 $\rightarrow$ 0.257	0.111 $\rightarrow$ 0.113
Train–Test	0.257 $\rightarrow$ 0.260	0.111 $\rightarrow$ 0.112
Validation–Test	0.159 $\rightarrow$ 0.160	0.0696 $\rightarrow$ 0.0702

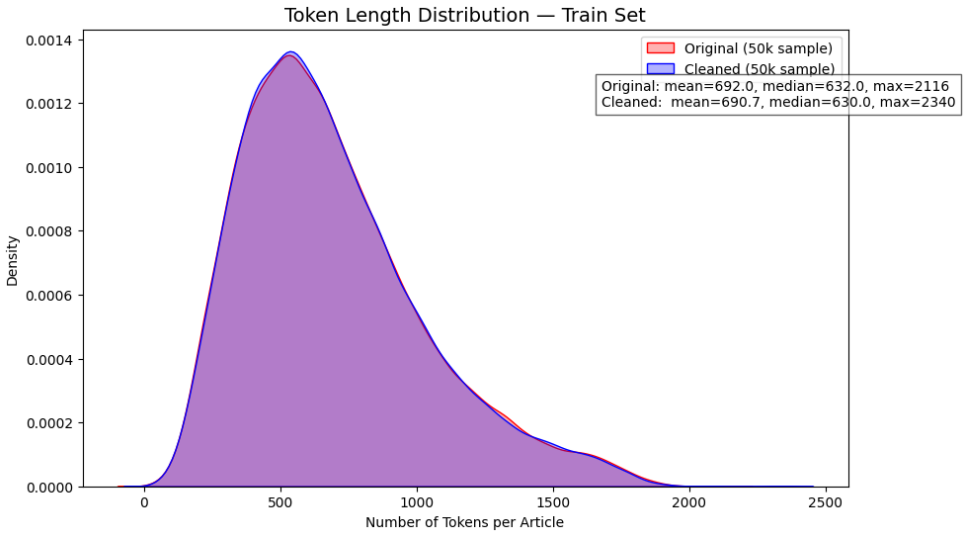

Figure 3.2: Token length distribution (train, 50k sample): original vs. cleaned curves overlap almost perfectly.

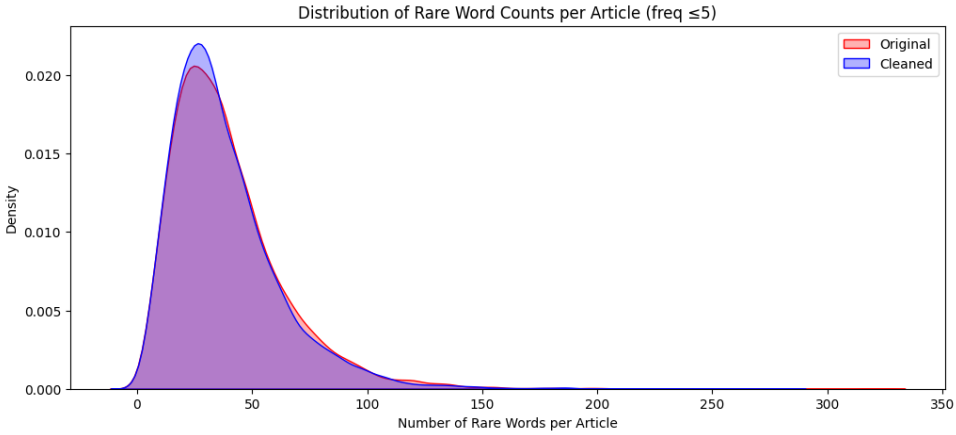

Figure 3.3: Rare-word counts per article (frequency ≤ 5): heavy-tailed shape preserved before and after cleaning.

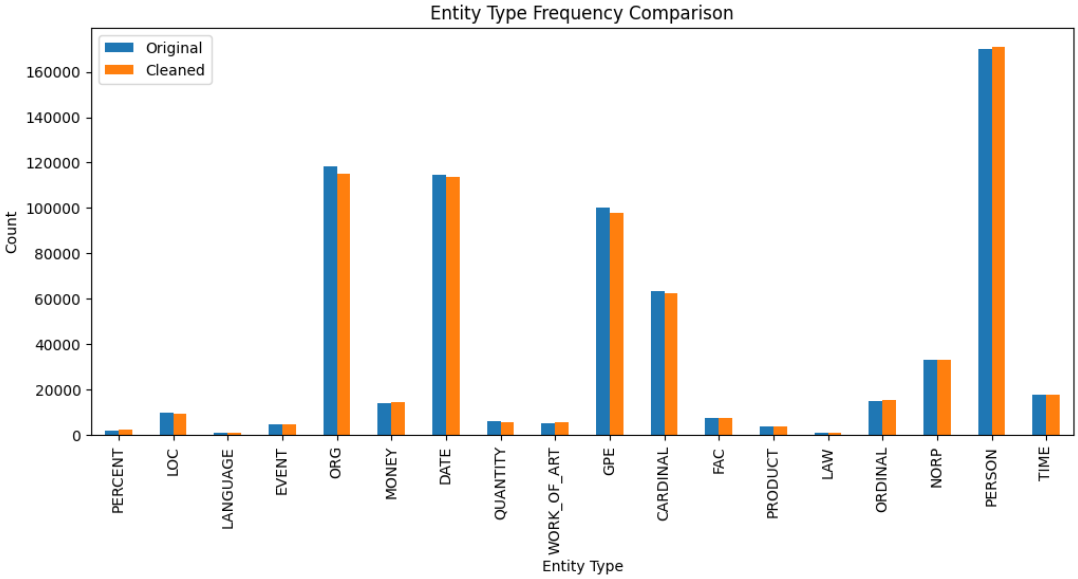

Figure 3.4: Entity-type frequencies: dominant types (PERSON, ORG, GPE) unchanged, confirming preservation of entity-richness.

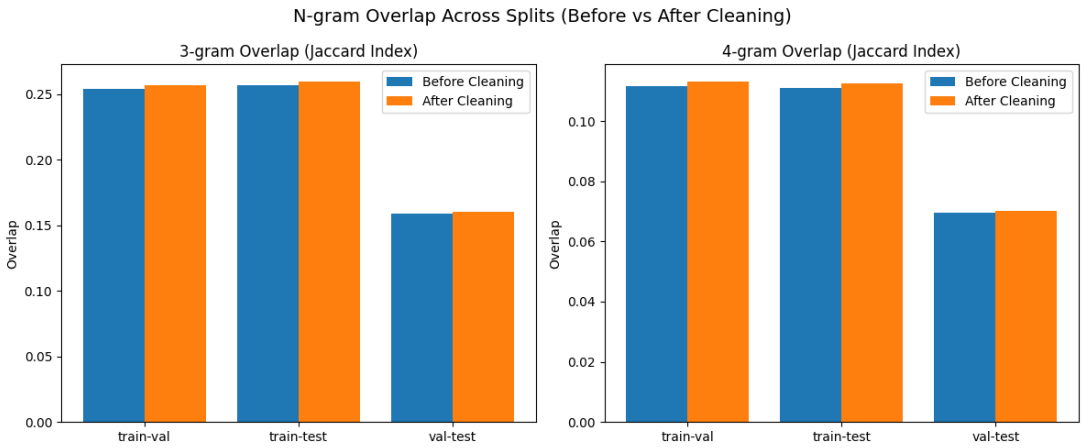

Figure 3.5: Cross-split 3-gram and 4-gram Jaccard overlap (before vs. after): only marginal increases, confirming stable diversity and minimal distributional drift.

3.4 Tokenisation & encoding choices

3.4.1 BART Native Tokenizer/Embeddings

All experiments were tokenized with the pre-trained BART-base tokenizer, which uses byte-level BPE in the same family as RoBERTa and GPT-2 (Lewis et al., 2020; Liu et al., 2019; Radford et al., 2019), building on subword BPE principles (Sennrich et al., 2016). Operating at the byte level eliminates out-of-vocabulary symbols while retaining subword compression via learned merges (Radford et al., 2019; Liu et al., 2019; Sennrich et al., 2016). Because the tokenizer's vocabulary is coupled to BART's pre-trained embedding matrix, no additional embedding training was required. Cleaned source articles were segmented to a maximum of 1,024 tokens, and reference highlights to 256 tokens with end-of-sequence markers appended. Full architectural details of BART are deferred to Section 4.

3.4.2 SentencePiece (unigram) tokenizer (trained but not used in this study)

For reproducibility and future model-agnostic scaling, a SentencePiece unigram tokenizer (Kudo and Richardson, 2018) was trained on sampled CNN/DailyMail source-side articles only. Key settings were: vocabulary 32,000; character coverage 0.9995; normalisation nmt_nfkc_cf; byte fallback enabled; maximum sentence length 40,000 characters; input sample 1.5 M lines from a capped corpus of 2.0 M lines. This tokenizer was not used in the reported BART experiments (to preserve compatibility with BART's pre-trained embeddings) but is retained to enable comparative studies with non-BART architectures. Key training parameters are summarised in Table 3.3. A complete configuration table is provided in Appendix **??**.

3.4.3 Rationale, Implications and Future Use

Selecting BART's native byte-level BPE ensured strict compatibility with the model's pre-trained embeddings and preserved comparability with BART baselines (Lewis et al., 2020).

Table 3.3: Configuration of the trained SentencePiece unigram tokenizer.

Parameter	Value
Vocabulary size	32,000
Character coverage	0.9995
Normalisation	nmt_nfkc_cf
Byte fallback	Enabled
Max sentence length	40,000 characters
Input sample	1.5 M lines (from 2.0 M capped corpus)
Model type	Unigram

The trained SentencePiece unigram model provides a leakage-safe, reusable option for future experiments across alternative architectures; it was trained exclusively on source articles, consistent with standard summarisation practice to avoid target-side leakage via the tokenizer. In effect, the strategy supports two aims: (i) faithful replication of BART-based results with robust byte-level segmentation ,and (ii) forward-looking portability using a consistent, source-only tokenizer for non-BART models.

3.5 Legal, Social, Ethical and Professional Issues

Dataset Provenance and Lawful Basis

This study uses the public, non-anonymised CNN/DailyMail corpus strictly for research purposes. Raw publisher texts are not redistributed. Only short quoted excerpts appear in the dissertation for criticism and review with citation. The non-anonymised variant is widely used in modern summarisation work (See et al., 2017).

Bias, representativeness, and duplication

Under-representation of communities and language varieties in NLP datasets is recognised as a systemic problem (Bender and Friedman, 2018; Lee et al., 2022). Assessing or correcting such imbalance is not the focus here. Instead, this project emphasises over-representation that arises from structural duplication and repeated strings, which can overweight certain patterns and encourage memorisation. In line with evidence that deduplication improves generalisation and reduces memorised copying, this project removes exact and near-duplicates, together with outlet boilerplate, while preserving genuine editorial content and the integrity of evaluation splits (Lee et al., 2022). Copy coverage is also analysed as a modelling choice (see Chapter 4). Post-cleaning diagnostics showed near-identical curves for token lengths and rare-word counts, and unchanged entity-type frequencies, confirming that the intervention reduces repetition without distorting the dataset's editorial signal.

Faithfulness and potential misuse

Abstractive summarisation is prone to generating unsupported content. Faithfulness was monitored through entity-level support analyses and the use of copy-aware and coverage mechanisms to discourage unsupported entities and repetition (see Chapter **??**), in line with recommendations from recent evaluations (Maynez et al., 2020). The system is a research prototype; generated summaries are treated as model outputs rather than verified facts and are not intended for deployment in sensitive domains.

Transparency and documentation

Ethical transparency is strengthened by adopting the principles of *Data Statements* (documenting data characteristics and caveats) (Bender and Friedman, 2018), *Datasheets for Datasets* (motivation, composition, collection, recommended uses) (Gebru et al., 2021), and *Model Cards* (intended use, limitations, reported metrics) (Mitchell et al., 2019). Concretely, deterministic preprocessing, manifest files per split, and clear licensing/attribution notes are provided to support replicability.

Normative stance

A proportionate ethics position is taken, following scholarship that recognises a rebuttable presumption of academic freedom in research, variation in legal and ethical norms across jurisdictions, and the need to focus on concrete harms rather than overly broad gatekeeping (Tsarapatsanis and Aletras, 2021). Given that the dataset is composed of public journalism, incremental privacy risk from processing is limited, though not dismissed. Access-controlled storage, restrained quotation, and the content-preserving cleaning choices above are justified as harm-reduction measures that also improve evaluation fidelity.

4 Methodology

This chapter outlines the methodological framework adopted in the study. The system is based on the Transformer encoder–decoder architecture of BART (Vaswani et al., 2017), instantiated using the publicly released Hugging Face `facebook/bart-base` checkpoint (Lewis et al., 2020; Hugging Face, 2020). To improve rare-word retention and reduce repetition, the baseline is extended during training with a pointer–generator mechanism and a coverage loss, following See et al. (2017). At inference, a span-continuation heuristic adapted from span-copying methods such as SeqCopyNet (Zhou et al., 2018; Singh et al., 2020) is applied to encourage contiguous copying of source-aligned spans. These modifications result in a hybrid configuration that combines copy-aware training with span-aware decoding.

The remainder of this chapter is organised as follows. Section 4.1 presents a high-level overview of the full pipeline. Section 4.2 describes the model architecture, beginning with the base encoder-decoder (§4.2.1) and extending to the pointer-generator with coverage (§4.2.2). Section 4.3 details the training protocol, including supervision signals and the phased parameter schedule. Section 4.4 outlines the inference framework, covering grouped diverse beam search (§4.4.1), span-aware continuation (§4.4.2), and reranking (§4.4.3).

4.1 System Overview

The system targets abstractive summarisation of CNN/DailyMail with emphasis on rare-word retention and factual consistency. It is instantiated through BART-base (Lewis et al., 2020), with the decoder augmented during training by a pointer–generator head and coverage regulariser (See et al., 2017) to balance copying and generation while discouraging repetition. At inference, span-continuation is enabled through a CopyNext-style heuristic (Zhou et al., 2018; Singh et al., 2020), while grouped diverse beam search promotes candidate diversity (Vijayakumar et al., 2016). Outputs are then re-ranked to prioritise factuality and coherence.

Together, these components produce a hybrid configuration that merges copy-aware training with span-aware decoding. Both baseline and hybrid systems share the same preprocessing pipeline, training schedule, and decoding hyperparameters to ensure comparability. The abstract flow of this framework is shown in Figure 4.1, where the left branch represents the training-time extension with the pointer–generator and coverage loss, and the right branch depicts the inference-time pipeline incorporating diverse beam search and span-aware re-ranking. The detailed architectural mechanisms and decoding procedures are presented in later sections.

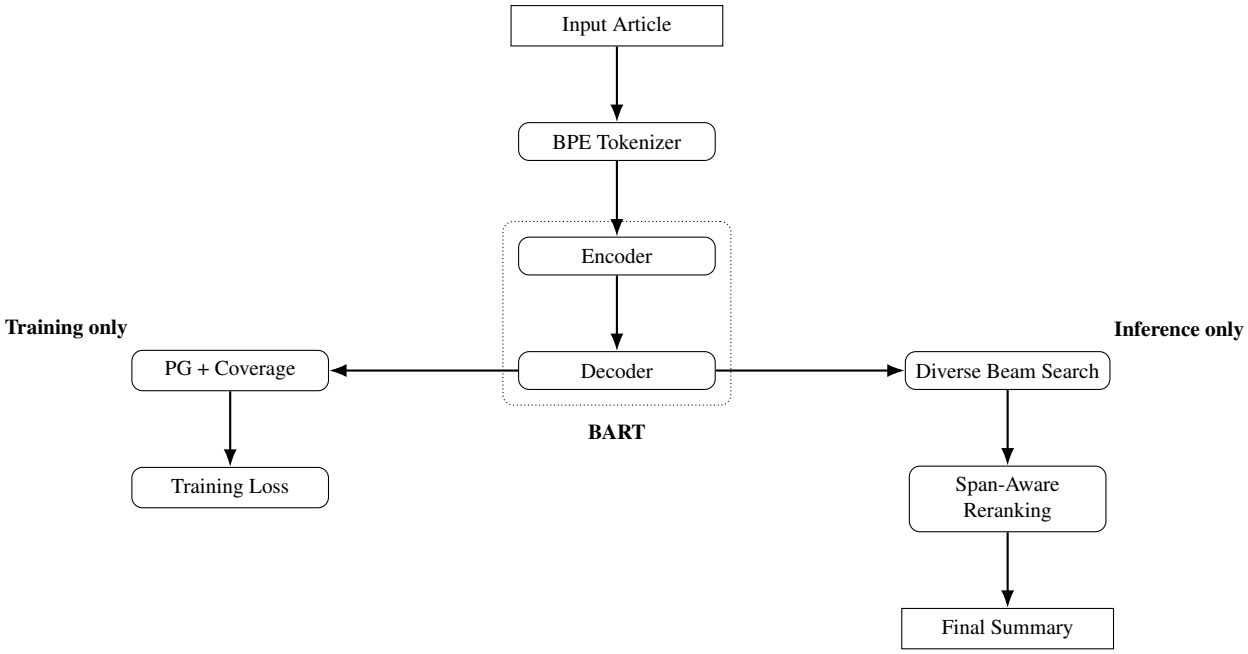

Figure 4.1: Abstract system overview. Input articles are tokenised, encoded and decoded by BART. Training augments the decoder with a pointer–generator and coverage loss, while inference applies diverse beam search with span-aware re-ranking to produce the final summary.

4.2 Model Architecture

This study employs a Transformer encoder–decoder backbone (Vaswani et al., 2017) instantiated as BART-base (Lewis et al., 2020). BART "uses the standard sequence-to-sequence Transformer architecture," replaces ReLU with GELU (Hendrycks and Gimpel, 2016), applies parameter initialisation from $\mathcal{N}(0,\, 0.02)$, and in its base configuration has six encoder and six decoder layers (Lewis et al., 2020). In contrast with BERT, BART's decoder includes a cross-attention sublayer to the encoder's final hidden layer and omits BERT's additional pre-LM feed-forward network.

As shown in Figure 4.2, the Transformer encoder consists of a stack of identical layers, each with two sublayers: multi-head self-attention and a position-wise feed-forward network. Each sublayer is wrapped with residual connections and layer normalisation (Vaswani et al., 2017). The decoder mirrors this structure but adds (i) masked self-attention over the target prefix and (ii) encoder–decoder cross-attention that conditions generation on encoder outputs.
BART instantiates this encoder–decoder design with a bidirectional encoder and an autoregressive decoder (Lewis et al., 2020). The encoder attends freely over the full input sequence, while the decoder generates left-to-right, attending to both its own past outputs and to encoder representations (Figure 4.3).

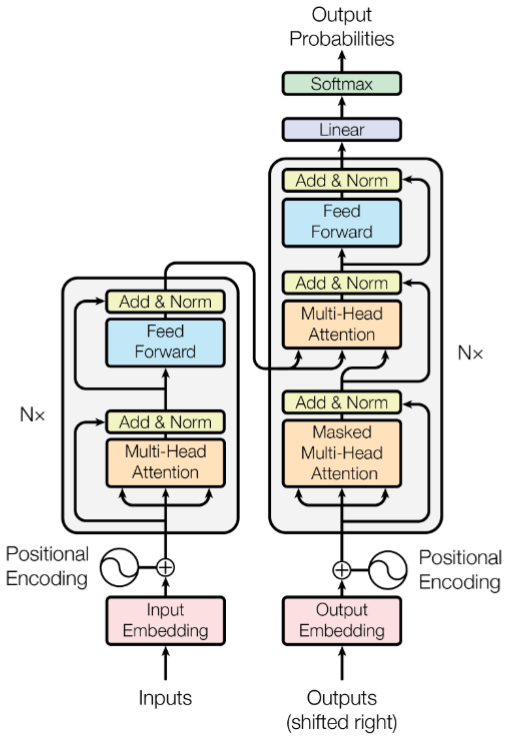

Figure 4.2: Transformer encoder–decoder architecture. Reproduced from Figure 1 in Vaswani et al. (2017).

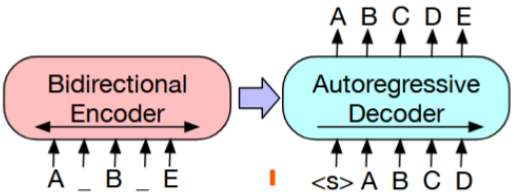

Figure 4.3: BART architecture. Adapted from Figure 2 in Lewis et al. (2020).

4.2.1 Base Encoder-Decoder

Tokenisation, Embeddings and Positions

Inputs are tokenised using the RoBERTa byte-level BPE vocabulary adopted by BART (Liu et al., 2019; Lewis et al., 2020). Positional information is injected by adding positional embeddings to the input representations "at the bottoms of the encoder and decoder stacks" (Vaswani et al., 2017). In BART, these embeddings are learned and absolute rather than sinusoidal (Lewis et al., 2020).

Attention

As illustrated in Figure 4.4, attention operates by projecting queries, keys, and values into multiple subspaces, computing attention scores through scaled dot-product attention, and then concatenating the resulting context vectors across heads followed by a linear transformation.

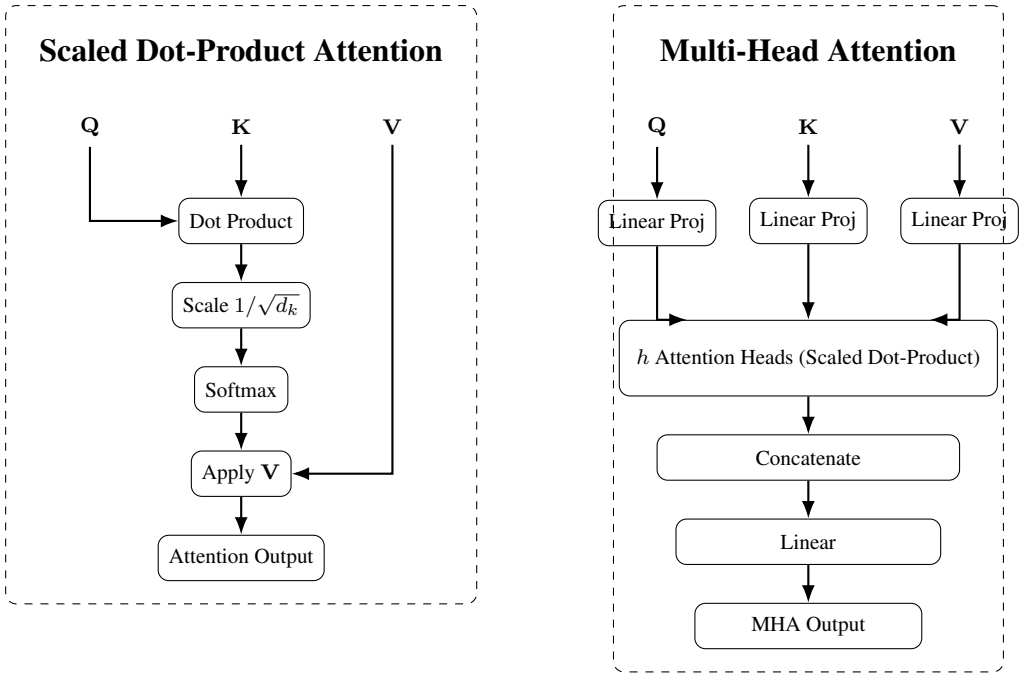

Figure 4.4: Figure 3 – (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel. Reproduced from Figure 3 in Vaswani et al. (2017).

Inside each attention block, queries, keys, and values are first obtained through linear projections of the inputs (Vaswani et al., 2017). The scaled dot-product attention mechanism is then defined as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{4.1}$$

where Q, K, and V are the query, key, and value matrices respectively, and d_k is the key dimensionality used to scale the dot product.

To improve expressivity, multi-head attention applies this operation in parallel across multiple learned subspaces. Formally, with W_i^Q, W_i^K, W_i^V, and W^O denoting the learned projection matrices, it is given by

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O, \\ \text{where} \quad \text{head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V). \end{aligned} \tag{4.2}$$

Each position in the sequence is further transformed by a two-layer feed-forward network (FFN), applied independently to every position (Vaswani et al., 2017):

$$\text{FFN}(x) = \max(0,\, xW_1 + b_1)W_2 + b_2. \tag{4.3}$$

Activation

BART follows this formulation but replaces the ReLU activation in the FFN with the Gaussian Error Linear Unit (GELU) (Hendrycks and Gimpel, 2016). GELU is defined as

$$\text{GELU}(x) = x\Phi(x) = x \cdot \frac{1}{2}\Big[1 + \text{erf}\Big(\frac{x}{\sqrt{2}}\Big)\Big], \tag{4.4}$$

where $\Phi(x)$ is the cumulative distribution function of the standard normal distribution. An efficient approximation is also commonly used in practice:

$$\text{GELU}(x) \approx 0.5x\Big(1 + \tanh\big[\sqrt{2/\pi}\,(x + 0.044715x^3)\big]\Big) \;\approx\; x \cdot \sigma(1.702x). \tag{4.5}$$

BART-base uses this GELU activation in all feed-forward sublayers, with parameters initialised from $\mathcal{N}(0,\ 0.02)$ and six encoder and six decoder layers in total (Lewis et al., 2020).

Encoder and Cross-Attention Notation

For later reference, let the encoder's top-layer outputs be denoted by

$$H = \{h_i\}_{i=1}^{n}, \tag{4.6}$$

where h_i represents the contextualised vector for the ith input token. This notation aligns with the description by Vaswani et al. (2017) that *"the encoder maps an input sequence of symbol representations $(x_1, \ldots, x_n)$ to a sequence of continuous representations $z = (z_1, \ldots, z_n)$"* (§3).

At decoding step t, let s_t denote the top-layer decoder state. Cross-attention over H produces attention weights

$$a_t = \{a_{t,i}\}_{i=1}^{n}, \tag{4.7}$$

and the corresponding context vector is given by the weighted sum

$$h_t^* = \sum_{i=1}^{n} a_{t,i} h_i. \tag{4.8}$$

As defined earlier, scaled dot-product attention is

$$\text{Attention}(Q, K, V) = \text{softmax}\Big(\frac{QK^\top}{\sqrt{d_k}}\Big)V \quad (Vaswaniet\ al.,\ 2017). \tag{4.9}$$

In the context of cross-attention, the values V are taken to be the encoder outputs ($V = H$), while the queries Q come from the current decoder state s_t. Writing the attention output at step

t as a weighted sum of these values directly yields the equality for h_t^*. The compact form

$$h_t^* = \sum_i a_{t,i} h_i \tag{4.10}$$

is therefore introduced as a direct consequence of the scaled dot-product formulation. We use the tuple $(s_t,\ a_t,\ h_t^*)$ in the decoder formulation described in §4.2.2.

Output Projection (baseline head)

As noted by Vaswani et al. (2017), "we also use the usual learned linear transformation and softmax function to convert the decoder output to predicted next-token probabilities." Following this description, the baseline vocabulary distribution is written as

$$P_{\text{vocab},t} = \text{softmax}(W_{\text{lm}} s_t + b_{\text{lm}}), \tag{4.11}$$

where s_t is the top-layer decoder state at time step t, W_{lm} is the learned output projection matrix, and b_{lm} is a bias term. This produces a categorical distribution over the fixed vocabulary at each decoding step.

This head constitutes the plain Transformer decoder output, which serves as the *baseline*. In §4.2.2, it will be replaced by a pointer-generator mixture with coverage, enabling explicit copying from the source text in addition to generation from the fixed vocabulary.

Pretrained Checkpoint and Configuration

Training a sequence-to-sequence Transformer from scratch requires very large corpora, so a pretrained model was adopted. BART uses RoBERTa-style denoising pretraining; RoBERTa itself was trained on five English corpora totalling approximately 160 GB (BookCorpus, English Wikipedia, CC-News, OpenWebText, and Stories) (Liu et al., 2019). BART reports performance comparable to RoBERTa under similar resources (Lewis et al., 2020), meaning it already captures the structure and semantics of English while remaining amenable to task-specific adaptation.

This work uses the `facebook/bart-base` checkpoint, which implements the canonical BART encoder-decoder at a tractable scale. It models long-range dependencies and rare entities, yet is small enough to fine-tune our pointer-generator with coverage extensions on a single GPU. The larger `facebook/bart-large-cnn` checkpoint ($\sim$406M parameters) was avoided since it is already fine-tuned on CNN/DailyMail, risking domain leakage and confounding, added to this is it's higher compute cost which would hinder controlled ablations (Hugging Face, n.d.). Choosing `bart-base` keeps compute manageable and allows attribution of performance gains to the proposed copy-aware mechanisms rather than to inherited task-specific pre-training. Its configuration is summarised in Table 4.1.

Table 4.1: Configuration of the `facebook/bart-base` checkpoint used in this work (Hugging Face, 2020).

Component	Specification
Encoder layers	6
Decoder layers	6
Hidden size	768
Feed-forward dimension	3072
Attention heads	12 per layer
Vocabulary size	BPE (∼50k merges; RoBERTa vocab)
Positional embeddings	1024 maximum length
Total parameters	∼139M

Provenance: the mathematical definitions of attention, multi-head attention, feed-forward networks, and positional encodings follow Vaswani et al. (2017). BART's architectural choices (GELU activation, $\mathcal{N}(0, 0.02)$ initialisation, six encoder and six decoder layers, the decoder's cross-attention, and omission of BERT's additional pre-LM feed-forward block) follow Lewis et al. (2020). All concrete dimensions reported in Table 4.1 correspond to the Hugging Face `facebook/bart-base` checkpoint used in this study (Hugging Face, 2020).

4.2.2 Pointer–generator and Coverage (PGC)

The baseline vocabulary head defined in Section 4.2.1 is replaced by a pointer–generator mixture and augmented with a coverage regulariser, following the formulation of pointer–generator networks for summarisation (See et al., 2017). The encoder–decoder layers, attention computations, masking, and normalisation conventions remain as in Section 4.2.1.

Notation carried over. We reuse the same symbols introduced previously: $H = \{h_i\}_{i=1}^{n}$ denotes the encoder's top-layer outputs; s_t is the top-layer decoder state at step t; $a_t = \{a_{t,i}\}_{i=1}^{n}$ are the encoder–decoder (cross) attention weights over source positions at step t; and $h_t^* = \sum_i a_{t,i} h_i$ is the corresponding context vector.

Pointer–generator head (copy + generate)

A vocabulary distribution is first computed from the decoder state and context. In the original formulation of See et al. (2017), a two-layer affine projection with softmax is applied to the concatenation of $[s_t, h_t^*]$:

$$P_{\text{vocab}} \;=\; \text{softmax}\Big(V'\big(V\,[s_t, h_t^*] + b\big) + b'\Big). \tag{4.12}$$

A scalar generation probability $p_{\text{gen}} \in [0, 1]$ at time step t is then computed from the context, decoder state, and current decoder input embedding x_t (See et al., 2017):

$$p_{\text{gen}} \;=\; \sigma\big(w_h^\top h_t^* \;+\; w_s^\top s_t \;+\; w_x^\top x_t \;+\; b_{\text{ptr}}\big), \tag{4.13}$$

where w_h, w_s, w_x and b_{ptr} are learned parameters and σ denotes the sigmoid activation.
The final next-token distribution over the *extended* vocabulary (base vocabulary $\cup$ source tokens) is the convex combination (See et al., 2017):

$$P_t(w) \;=\; p_{\text{gen}}\, P_{\text{vocab}}(w) \;+\; \big(1 - p_{\text{gen}}\big) \sum_{i:\, w_i = w} a_{t,i}. \tag{4.14}$$

If w is out-of-vocabulary (OOV), only the copy term contributes; if w does not appear in the source, only the vocabulary term contributes. Concretely, the BART LM head produces logits that are converted to P_{vocab} via softmax (Vaswani et al., 2017; Lewis et al., 2020); the copy distribution is formed by scattering the final-layer cross-attention weights $a_{t,\cdot}$ onto the unique source-token types (with padding positions masked). The mixture in Eq. (4.14) *replaces* the baseline softmax head during both training and decoding.

Coverage

To discourage repetition, a coverage mechanism is applied. Past attention is accumulated as in See et al. (2017):

$$c^t \;=\; \sum_{t'=0}^{t-1} a^{\,t'}. \tag{4.15}$$

Overlap between the current attention and the accumulated coverage is penalised with

$$\text{covloss}_t \;=\; \sum_{i=1}^{n} \min\big(a_{t,i},\, c_i^t\big). \tag{4.16}$$

The per-step training objective thus becomes

$$\mathcal{L}_t \;=\; -\log P_t\big(w_t^*\big) \;+\; \lambda\, \text{covloss}_t, \tag{4.17}$$

where w_t^* is the gold token at step t and $\lambda > 0$ is a tuned weight. In this study, coverage is used purely as a *loss-only* regulariser on the cross-attention distributions. Teacher-forced training minimises $\sum_t \mathcal{L}_t$ over the mixed distribution $P_t(\cdot)$.

This pointer-generator with coverage formulation replaces the baseline projection head, enabling the model to flexibly generate novel tokens while retaining the ability to copy rare entities and numbers directly from the source. Coverage ensures that the decoder avoids over-focusing on the same source fragments across steps, reducing repetition and improving consistency.

4.3 Training Stage

Training was required to specialise the model to the CNN/DailyMail summarisation task under the PGC head (see §4.2.2). The pointer–generator introduces new parameters (the gating and projection layers) that are not pretrained and must therefore be learned from scratch. The gate must be calibrated so that tokens are copied when appropriate rather than hallucinated, and the coverage term must shape cross-attention to minimise repetition while maintaining adequate source coverage. Training also adapts the model to the corpus' compression ratio and length constraints.

A phased schedule of 7k updates was adopted to: stabilise the PGC head; calibrate copy versus generate behaviour; then consolidate with a conservative partial unfreeze. Source and target caps were fixed at 400 and 100 tokens respectively, matching the canonical PGC setup and its compression regime (See et al., 2017).

4.3.1 Training Signals

Several signals were logged throughout training and guided subsequent adjustments:

- **Cross-entropy loss (CE)**: Fit to gold summaries; steady decline indicated task learning.
- **Coverage loss**: Decline reflected healthier attention distribution and reduced repetition.
- **Mean gate statistics** $(\bar{P}_{\text{copy}}, \bar{P}_{\text{gen}})$: Captured how frequently the model copied versus generated; a controlled rise and plateau in copying indicated calibration.
- **Gradient norm**: Spikes suggested overly aggressive optimisation; stable norms indicated healthy training.
- **Throughput (tokens/s)**: Monitored efficiency and regressions after freezing or unfreezing encoder layers.

These signals, summarised in Table 4.2, provided the evidence base for decisions about when to enable gating, when to strengthen coverage, and when to freeze or unfreeze the encoder.

Loss signals across training. The CE loss began near 2.37 in the early steps and declined steadily, stabilising around 2.27 by the final phase. Coverage loss followed a consistent downward trajectory, from $\sim$0.59 in the first thousand updates to $\sim$0.32 at convergence, reflecting reduced repetition pressure. Mean copy probability, initially negligible ($\sim$0.18), rose gradually and plateaued near 0.25 by the last phase, establishing a balance where roughly one in four tokens were sourced directly from the input. Gradient norms, extremely high at the outset ($>$200k), dropped to single-digit values once warm-up was complete and remained stable thereafter. Taken together, these signals confirmed that the model had stabilised: summarisation loss had converged, coverage was healthy, and the pointer gate was calibrated.

Table 4.2: Evolution of loss signals across training.

Phase (updates)	CE loss ()	Coverage loss ()	Mean p_{copy}	Mean p_{gen}	Gradient norm ()
Initial (0–2k)	2.37 ↓ 2.34	0.59 ↓ 0.51	0.18 → 0.20	0.82 → 0.80	200k → 50k
Mid (3–5k)	2.31–2.29	0.45–0.38	0.22–0.24	0.76–0.78	∼5–10
Tail (6–7k)	2.29 → 2.27	0.33 → 0.32	0.24 → 0.25	0.76 → 0.75	∼2.4–2.6

4.3.2 Phase Schedule and Parameter Changes

Training was deliberately structured as a sequence of short phases, with parameters adjusted only when logged signals indicated a clear need. The exact values are listed in Table 4.3.

Phase-by-phase adjustments.

- **Initial (0–2k updates)**: End-to-end training with moderate coverage ($\lambda_{\text{cov}} = 1.0$) and gate off. A relatively high learning rate (3e–5) with a short warm-up (6% of steps) was used to help the decoder adapt quickly. Gradient norms fell from unstable values to stable ranges.
- **Top-up (2–3k)**: LR lowered (1e–5) with short warm-up. No gate. CE plateaued with stable gradients.
- **Nudge (3–4k)**: LR lowered (1e–5) with short warm-up. No gate. CE plateaued with stable gradients.
- **Main (4–5k)**: Encoder weights frozen; LR raised (7e–5) for decoder and PGC head; coverage weight relaxed (0.9). Gate sharpened while CE remained stable.
- **Main top-up (5–6k)**: LR reduced (1e–5); coverage restored (1.2) to suppress signs of repetition and ensure selective copying. CE plateaued; coverage loss fell further.
- **Tail (6–7k)**: LR very low (3e–6) with 100-step warm-up. Top two encoder layers unfrozen for gentle co-adaptation between encoder attention and the pointer gate without overwriting pretrained knowledge. CE stable at ∼2.27; coverage ∼0.32; p_{copy} plateaued near 0.25.

Implementation Constants

Table 4.4 lists the optimisation and implementation parameters that were held constant across all training phases. AdamW with weight decay was used, with gradient clipping (≈1.0) to maintain stability. "Eager" attention mode was required to expose cross-attentions for the PGC head. Random seeds were fixed to 0 for reproducibility.

Table 4.3: Phase-by-phase knobs during training.

Phase	Step range	Encoder	LR	Warm-up	λ_{cov}	λ_{gate}
Initial	0–2,000	unfrozen	3e–5	0.06	1.0	0
Top-up	2,000–3,000	unfrozen	1e–5	0.03	1.0	0
Nudge	3,000–4,000	unfrozen	1e–5	0.03	1.2	0.02
Main	4,000–5,000	frozen	7e–5	0.10	0.9	0.03
Main top-up	5,000–6,000	frozen	1e–5	0.03	1.2	0.03
Tail	6,000–7,000	unfroze last 2 layers	3e–6	100 steps	1.2	0.03

Table 4.4: Optimisation and implementation constants across all phases.

Item	Setting
Optimiser / decay / clip	AdamW with weight decay 0.01; gradient clipping ≈1.0
Attention implementation	`eager` (to expose cross-attentions for PGC head)
Random seeds	0
Libraries (pinned)	transformers v4.43.4, accelerate v0.34.2, datasets v2.20.0

4.4 Decoding

This section describes the inference pipeline used to produce the final summary for each article. As shown in Figure 4.5, candidates are first generated with *grouped diverse beam search* (DBS), with two length-control pools (MED and LONG). During decoding, two *span-aware* processors apply small logit biases to encourage coherent continuation of source-aligned spans. The union of baseline and hybrid candidates is then passed to a two-stage *reranking* framework that filters unsafe outputs and selects a final summary.

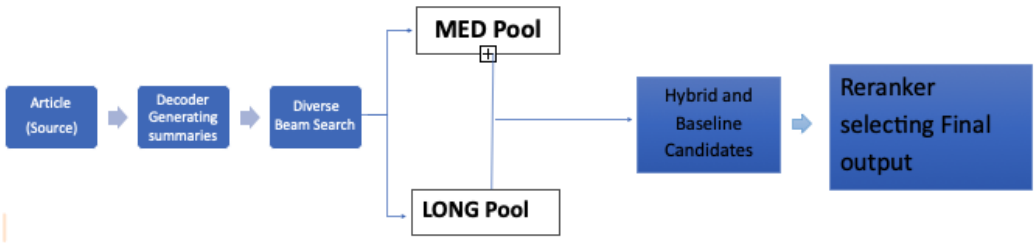

Figure 4.5: Decoding pipeline. Diverse beam search feeds two length-controlled pools (MED/LONG). Both the baseline head (§4.2.1) and the hybrid PGC head (§4.2.2) are decoded under identical controls; their candidates are merged and reranked to select the final output.

4.4.1 Diverse beam search (grouped)

A single article may admit several plausible summaries. Standard beam search often returns near-duplicates, which limits the usefulness of reranking and span-aware decoding. We adopt **Grouped Diverse Beam Search (DBS)** (Vijayakumar et al., 2016): the width-B beam is partitioned into G groups, and a dissimilarity term penalises later groups for reusing tokens or

n-grams chosen by earlier groups. This *doubly greedy* strategy keeps the efficiency of beam search while yielding a more varied candidate set.

A well-known drawback of wide beams is *length bias*, where short outputs are over-preferred (**??**). To counteract this, decoding is performed twice per article with distinct length controls: a **MED** pool (concise) and a **LONG** pool (fuller). The two pools maintain different minimum/maximum new-token limits and length normalisation, yet all other settings remain consistent. The pools are merged before reranking (§4.4.3), preserving both short and extended alternatives.

DBS hyperparameters.

Core DBS knobs and their qualitative effects are summarised in Table 4.5. Pool-specific decoding controls are listed in Table 4.6. Values were selected by validation tuning (§4.5); once fixed, the configuration was frozen for all test runs to ensure comparability.

Table 4.5: DBS hyperparameters.

Hyperparameter	**What it controls**	**Practical effect**
Beam width B	Total search budget	Wider exploration; interacts with length bias
Number of groups G	Partition of beam budget	More cross-group variety (group size $b = B/G$)
Group size b	Per-group beam width	Local exploration within each group
Diversity strength λ	Weight of dissimilarity	Larger $\lambda \Rightarrow$ stronger push away from earlier groups
Dissimilarity $\Delta(\cdot)$	Penalty shape	Token/n-gram overlap penalty per Vijayakumar et al. (2016)

4.4.2 Span-awareness

To reinforce factual retention without retraining, decoding was complemented with two *logits processors* that add a small *positive* bias to the next-token logit when a copy-consistent continuation is detected. The adjustment is additive (it never forces a choice), remains compatible with DBS, and respects global repetition constraints.

Table 4.6: Global decoding controls by pool (applied to both baseline and hybrid models).

Control	MED pool (concise)	LONG pool (fuller)	Notes
Min / max new tokens	Moderate floor / cap	Higher floor / higher cap	Hard bounds vs under/over-generation
Length normalisation / word reward	Mild	Stronger	Counters length bias in scoring
Early stopping	Enabled	Enabled	Efficiency control
No-repeat n-gram	3-gram	3-gram	Prevents local loops
Deduplicate identical beams (post-decode)	On	On	Removes exact duplicates before reranking

Processors

Two complementary mechanisms were used (Table 4.7).

- **Suffix-match continuation (CopyNext-style).** The current hypothesis suffix (up to 6 tokens) is compared against contiguous slices of the source. When a match is found, the logit of the next aligned source token is increased by a bias γ, subject to guards (word-like tokens only; no punctuation/special tokens; max span length). This implements the principle of CopyNext (Singh et al., 2020) as an inference-time bias.
- **Entity-aware continuation (CopyNext + NER).** A second processor conditions the bias on named-entity boundaries. For each article a copy-map over source k-grams is built with spaCy NER; at decode step t the longest matching suffix triggers a bias γ for generic spans or a stronger γ_{entity} inside entity spans (PERSON/ORG/GPE/DATE/MONEY), in the spirit of SeqCopyNet/SAGCopy (Zhou et al., 2018; Xu et al., 2020).

Table 4.7: Span-aware processors.

Processor	Input used	Bias rule	Guards applied	Motivation
Suffix-match continuation	Hypothesis suffix ($\leq$6 tokens) vs contiguous source slice	Add bias γ to next aligned source token	Punctuation/special-token block; word-like tokens only; max span length	Singh et al. (2020)
Entity-aware continuation	Pre-computed copy-map of source k-grams with NER spans	Add γ (generic) or stronger γ_{entity} (inside entity span)	Same as above	Zhou et al. (2018); Xu et al. (2020)

Safety mechanisms

To prevent runaway copying, the following safeguards remained active: (i) additive bias only (does not force selection); (ii) global *no-repeat 3-gram*; (iii) punctuation/word-filters; (iv) span length cap; and (v) downstream reranker penalties for unsupported entities (§4.4.3).

Tuning and final values.

Decoding parameters were tuned on the validation split by monitoring ROUGE-1/2/Lsum and entity precision/recall; once fixed, they were frozen for all test runs. Final values are listed in Table 4.8.

Table 4.8: Hyperparameters of span-aware processors.

Parameter	Value	Rationale
Generic bias γ	0.4	Encourages continuation of generic spans without overpowering base likelihood
Entity bias γ_{entity}	1.5	Stronger bias for within-entity continuations where factual accuracy matters most
Maximum span window	6 tokens	Limits suffix search to avoid long verbatim copying
Guards	Word-like filter; punctuation block	Prevents unnatural/degenerate copying
Global constraints	No-repeat 3-gram	Preserves diversity and prevents repetition loops

4.4.3 Reranking

DBS with dual pools yields diverse candidates, but some remain noisy (unsupported entities, fragmented copying, or drifting lengths). A two-stage reranking framework is therefore used: (i) a **Stage-A** per-article filter removes unsafe/structurally weak outputs; and (ii) a **linear reranker** (trained on validation and then frozen) selects the final summary from the remaining set.

Stage-A filtering (per article). For each candidate y, we compute a set of factual and structural features (Table 4.9) and combine them into a single composite heuristic score. The feature design was tuned empirically on validation data for this study (rather than copied from prior published formulae). As part of Stage-A, any candidate containing an *unsupported core entity* (PERSON/ORG/GPE) is rejected outright. The highest-scoring $M = 10$ candidates per article are retained for the final reranking pass; if all fail the core-entity rule, the rejection is relaxed to avoid empty sets.

The *length* component uses a Gaussian reward:

$$\mathrm{LengthReward}(y) \;=\; \exp\!\left(-\frac{(L(y)-L^*)^2}{2\sigma^2}\right), \tag{4.18}$$

where $L(y)$ is the candidate length, L^* an adaptive target estimated from article leads, and $\sigma = 20$.

Table 4.9: Stage-A features and rules.

Signal	Definition (per candidate)	Role in Stage-A
Strict entity score	NER overlap between article and candidate; core entities $\times 3$, non-core $\times 1$	Encourages supported entities
Hard fail (CORE)	Reject if any core entity occurs in candidate but not in article	Removes unsafe beams
Span contiguity (LCS)	Count of contiguous blocks from article$\rightarrow$candidate (length ≥ 2)	Rewards coherent copied spans
Length reward	Gaussian reward Eq. (4.18)	Discourages under/over-length
Non-core unsupported	Count of unsupported non-core entities (DATE, MONEY, etc.)	Penalises spurious facts

Base likelihood normalisation

To complement the heuristic features, we score each surviving candidate using the model's own likelihood. Given an article x and parameters θ, the *teacher-forced* average negative log-likelihood is

$$\ell(y \mid x) \;=\; -\frac{1}{T}\sum_{t=1}^{T} \log P_\theta\big(y_t \mid y_{<t}, x\big), \tag{4.19}$$

with $T = L(y)$ the candidate length in tokens. Since raw likelihoods tend to favour shorter outputs, we apply a length-normalisation exponent $\alpha = 0.3$ to obtain a plausibility score that puts different lengths on a more equal footing:

$$\mathrm{base\_norm}(y \mid x) \;=\; \frac{-\ell(y \mid x)}{L(y)^\alpha}. \tag{4.20}$$

Linear reranker (frozen on validation)

Final selection is performed with a linear scorer applied to the Stage-A survivors.[1]. Each candidate is represented by a feature vector (Table 4.10) combining factual overlap, structural continuity, model likelihood, and cross-candidate agreement. Within each per-article set, features are min–max normalised before scoring. The reranker computes

$$\mathrm{Rerank}(y \mid x) \;=\; w_0 \;+\; \sum_j w_j \, f_j(y \mid x), \tag{4.21}$$

where $f_j(y \mid x)$ are the normalised features and w_j the learned coefficients. Weights are trained once on the validation split and then frozen for all test runs. If the top-scoring candidate contains an unsupported *core* entity, a UCER-safe fallback selects the highest-scoring candidate without such an error.

Table 4.10: Final reranker features.

Feature	Definition	Purpose
Composite heuristic	Weighted entity, contiguity, length, penalty score (Stage-A)	Encodes factual & structural quality
Lead similarity	ROUGE-Lsum vs. article lead-3	Anchors summary to salient lead
Cross-candidate similarity	Mean pairwise ROUGE-Lsum vs. other beams	Lightweight centrality proxy
Base likelihood norm	Teacher-forced log-prob, length-normalised (Eq. 4.20)	Model plausibility signal
Entity consensus	Jaccard similarity of NER sets across beams	Promotes entity-consistent outputs
Unsupported non-core	Count of unsupported DATE/-MONEY/etc.	Penalises spurious facts

4.4.4 Precision-recall anchoring via λ (concept only)

To expose a controllable factuality policy in reranking, a scalar is defined to anchor selection toward entity precision (→1) or entity recall (→0). With micro-averaged entity precision/recall entP,entR*entP*, *entR*entP,entR computed per candidate, the entity term is

$$entity\_score(\lambda) = \lambda \, entP + (1 - \lambda) \, entR \tag{4.22}$$

This term may be combined with the existing features (length-normalised likelihood, span-contiguity, and length reward) in the linear scorer (§4.4.3) to select a single candidate. Intuitively, acts as a policy knob on faithfulness: higher values favour avoiding unsupported

[1]The reranker is intentionally pragmatic rather than theoretical. The chosen features reflect common concerns in abstractive summarisation—entity faithfulness, redundancy control, and length bias—while the exact feature set and coefficients are specific to this study.

entities; lower values favour retaining more reference entities.A small, diagnostic exploration of was run on 100 validation items using a wider decode than the final recipe (grouped DBS with 20 beams/10 groups and an entity-aware span processor). Because both the decoding and the sample size differ from the frozen MED/LONG setup, was not used in subsequent validation or test evaluations. Although excluded from the main evaluation, it helped to illustrate how precision–recall trade-offs can be steered, and is included here only as a potential tuning helper; this point is developed further in Chapter 6. Results are provided only as an illustration in Appendix X.

4.5 Validation experiments and tuning

This section reports how decoding length control (MED/LONG pools), model checkpoints, and final validation frameworks were selected. All experiments reuse the hybrid architecture and decoding stack defined earlier (§4.4), with grouped diverse beam search (DBS), span-aware continuation, and the two-stage reranking pipeline fixed unless otherwise noted.

4.5.1 Length control — experiments leading to the frozen MED/LONG pools

For each candidate setting, validation summaries were generated and their word-length distributions were compared to those of the references, alongside ROUGE.[2] Settings that consistently produced under- or over-length outputs were rejected. The smallest ranges that lay on the ROUGE plateau and aligned to references were then frozen for all subsequent experiments.

Table 4.11: Length tuning on validation. Rows correspond to different validation subsets (used deliberately to sample natural variability); absolute lengths differ across rows and were compared qualitatively to locate a stable operating region.

Beams	min_new	max_new	α (len-pen)	no-rep	R1 F1	R2 F1	RLsum F1	Pred len (mean/med) $\rightarrow$ Ref (mean/med)
5	26	60	3.0	4	0.3184	0.1194	0.2243	42.77 / 43.0 $\rightarrow$ 34.37 / 33.0
5	40	70	1.2	4	0.4092	0.1850	0.2799	50.17 / 51.0 $\rightarrow$ 58.37 / 55.0
5	40	120	1.2	3	0.4191	0.1928	0.2926	71.2 / 66.0 $\rightarrow$ 67.8 / 65.3*

* Different validation slices; comparisons interpreted qualitatively to find the ROUGE/length plateau.

Findings: the 26-60 window (even with $\alpha = 3.0$) produced over-length outputs and weaker ROUGE and was rejected; 40–70 improved ROUGE but tended slightly short versus references; probing 40–120 confirmed a ROUGE/length plateau at higher caps, but very high maxima risk long-tail outliers. To cover both concision and recall without long tails, we froze *two complementary pools* (Table 4.12) and reused them throughout.

Freeze statement

The MED (35–70, α=1.6) and LONG (55–100, α=2.0) pools were selected on validation by jointly inspecting ROUGE-Lsum and length statistics; these values were *frozen* and reused un-

[2]ROUGE (Recall-Oriented Understudy for Gisting Evaluation) measures n-gram and longest-common-subsequence overlap with references and is widely adopted in news summarisation for its correlation with human judgements and cross-paper comparability. We use ROUGE-Lsum as the primary selection signal on validation (Lin, 2004).

Table 4.12: Frozen length pools reused thereafter. DBS elsewhere: beams=10, groups=5, diversity penalty $\lambda_{\text{div}} = 0.3$; no-repeat n-gram$= 3$; early stopping on.

Pool	min_new	max_new	α	beams	groups	λ_{div}	Rationale
MED	35	70	1.6	10	5	0.3	Concise regime: mitigates under-length seen at 40–70 (α=1.2) by slightly lowering the minimum and raising α to resist premature EOS.
LONG	55	100	2.0	10	5	0.3	Coverage regime: anchored near the reference median; max reduced to 100 to prevent long-tail outliers.

changed in all subsequent experiments and in Chapter 5 test evaluation.

4.5.2 Checkpoint selection

We compared three checkpoints (steps 6k, 6.5k, 7k) on the same 200-item validation subset (seed=0). Decoding was held *identical* across checkpoints: the frozen pools above and the constants in Table 4.13. For each article, 20 candidates (10 MED + 10 LONG) were generated, Stage-A filtered (§4.4), scored with length-normalised likelihood, and reranked by the frozen pairwise LR model. The selection rule was fixed in advance: maximise ROUGE-Lsum; break ties by entity F1, then UCER (lower is better).

Table 4.13: Decoding constants used for checkpoint selection.

Component	Setting
DBS	beams=10, groups=5, diversity penalty $\lambda_{\text{div}} = 0.3$
Repetition control	no_repeat_ngram_size = 3
Early stopping	enabled
Source length cap	max_source_tokens = 400
Span-aware bias	$\gamma = 0.4$, $\gamma_{\text{entity}} = 1.5$, max_span = 6

Table 4.14 shows the hybrid model's metrics by checkpoint. Step 6500 achieves the **lowest UCER** and **highest entity F1** via a clear recall gain, making it the faithfulness-first option. Step 6000 remains very safe with low UCER and strong entity scores, so we retain both 6k and 6.5k as complementary points on the faithfulness/safety Pareto frontier. Step 7000 shows the **highest UCER** without compensating gains and was dropped.

Table 4.14: Hybrid validation metrics by checkpoint (green=best in column; red=worst; UCER: lower is better).

Checkpoint	R1 F1	R2 F1	RLsum F1	Entity P	Entity R	Entity F1	UCER
step 6000	**0.4233**	**0.1925**	**0.2970**	**0.4030**	**0.3810**	0.3917	0.0250
step 6500	0.4184	0.1885	0.2860	0.3896	**0.4143**	**0.4015**	**0.0200**
step 7000	0.4200	0.1882	0.2847	0.4107	0.3889	0.3995	**0.0300**

Effect of span bias

Holding checkpoint and decoding slice fixed (step 6500 on VAL-200), enabling the span-aware bias improves entity-level faithfulness and reduces UCER under otherwise identical conditions (Table 4.15).

Table 4.15: Effect of span bias at step 6500 (same slice, same decoding except span bias toggled).

System	RLsum F1	Entity P	Entity R	Entity F1	UCER
Baseline (no span bias)	0.2224	0.2455	0.3216	0.2785	**0.0400**
Hybrid (with span bias)	**0.2860**	**0.3896**	**0.4143**	**0.4015**	**0.0200**

4.5.3 Final frameworks (VAL-only, $N = 1{,}000$)

We fix two frameworks at steps 6000 and 6500 as the definitive validation settings going forward. Decoding remains *frozen* as in §4.5.1 and Table 4.13. Each framework decodes 1,000 validation items (20k candidates), which is sufficient to stabilise faithfulness/UCER trends while remaining computationally feasible; the pairwise LR reranker is trained on validation only and frozen for later tests.

Table 4.16: VAL-1000 results at step 6000. Hybrid improves entity precision/recall and reduces UCER, with small positive ROUGE shifts.

Variant	R1 F1	R2 F1	RL F1	Entity P	Entity R	Entity F1	UCER
Baseline	**0.3116**	**0.1184**	**0.2157**	**0.2324**	**0.3270**	**0.2717**	**0.179**
Hybrid	**0.3252**	**0.1233**	**0.2247**	**0.2564**	**0.3411**	**0.2927**	**0.031**

Table 4.17: VAL-1000 results at step 6500. Even with entity-aware ablated in the baseline, Hybrid still reduces UCER at roughly ROUGE parity; entity F1 dips slightly, reflecting a conservative bias against unsupported core entities.

Variant	R1 F1	R2 F1	RL F1	Entity P	Entity R	Entity F1	UCER
Baseline	**0.3225**	**0.1206**	**0.2227**	**0.2574**	**0.3490**	**0.2963**	**0.040**
Hybrid	0.3232	0.1201	0.2208	0.2549	0.3369	0.2902	**0.031**

4.6 Efficiency

This section details the engineering choices made to keep training, decoding, and evaluation efficient on a single NVIDIA T4–class GPU *accessed via Google Colab Pro*, while maintaining reproducibility and fidelity to the method in §4. The design prioritises (i) reduced memory footprint, (ii) higher computational efficiency, and (iii) minimal code complexity by relying on stable, well-supported libraries.

4.6.1 Training-time efficiency

Mixed-precision (AMP, fp16) with dynamic loss scaling

Training uses automatic mixed precision (AMP) with a `GradScaler` to avoid underflow and to keep matmuls on tensor cores, typically yielding $1.3{\sim}1.8\times$ speedups on Turing/Ampere GPUs while halving activation memory (Micikevicius et al., 2018; Paszke et al., 2019). In code, autocast wraps the forward pass, the loss is scaled, and gradients are unscaled before clipping.

Gradient accumulation

To emulate a larger effective batch without out-of-memory (OOM), the loop accumulates K micro batches before every optimiser step. This maintains stable optimisation while respecting device memory.

Efficient attention kernels (`attn_implementation=sdpa`)

The HF `transformers` loader enables PyTorch 2.x scaled dot-product attention (SDPA), which fuses softmax and matmul into a single kernel and chooses an efficient backend at runtime. This reduces both launch overhead and memory traffic compared to legacy Python-level attention implementations (Paszke et al., 2019; Wolf et al., 2020).

Gradient checkpointing (selective)

Checkpointing trades compute for memory by recomputing selected activations during backpropagation. It increases wall time per step slightly but enables larger batches or longer sequences on the same GPU budget (Chen et al., 2016). We enable it during later phases once the model stabilises.

AdamW with weight decay and clipping

We use decoupled weight decay AdamW (Loshchilov and Hutter, 2019) with gradient clipping (≈ 1.0) for robust optimisation. While fused AdamW kernels are available on recent CUDA stacks, the project notebook used the standard optimiser for portability; this keeps behaviour identical across machines while still benefiting from AMP and SDPA. A linear warmup/decay scheduler is used for stable early steps.

Memory–safe decoder training

Decoder `use_cache` is disabled during training (saves memory), and enabled only at inference where KV caching accelerates generation.

I/O Pipeline

Pinned host memory and non-blocking device transfers keep the input queue fed. The collator pads on CPU once and moves the whole batch with `tensor.to(device, non_blocking=True)` to avoid many small copies.

Table 4.18: Training efficiency knobs and observed/expected effects.

Knob	Mechanism / effect	Refs
AMP (fp16) + scaler	Tensor-core matmuls, smaller activations; typical $1.3{\sim}1.8\times$ speedup	Micikevicius et al. (2018)
SDPA attention	Fused, backend-optimised scaled dot-product attention	Paszke et al. (2019)
Grad accumulation	Larger effective batch without OOM	—
Grad checkpointing	Recompute-activations to cut memory, enable longer seqs	Chen et al. (2016)
AdamW + clipping	Stable, decoupled decay; protects against exploding grads	Loshchilov and Hutter (2019)
Disable `use_cache`	Frees decoder KV memory during training	Wolf et al. (2020)
Pinned & non-blocking I/O	Hides H2D copy latency, steadies throughput	Paszke et al. (2019)

4.6.2 Decoding-time efficiency

Batched generation with caching

Inference runs in fp16 under `autocast` with `use_cache=True` to reuse past keys/values, which reduces per-step cost for beam search substantially (Wolf et al., 2020). Articles are decoded in batches; the two length-controlled pools (MED/LONG; §4.4.1) are executed in batches rather than item-by-item.

Grouped Diverse Beam Search (DBS)

DBS preserves the $\mathcal{O}(B{\cdot}T)$ footprint of beam search while producing diversity through a simple dissimilarity term between G groups (Vijayakumar et al., 2016). Compared to sampling-based diversity, DBS keeps determinism and hardware-friendly vectorisation (Vijayakumar et al., 2016).

Lightweight span-aware processors

Span-continuation biases are implemented as *logits processors* that only add a small constant to selected next-token logits. They are additive (no sampling) and vectorisable, so they introduce negligible overhead compared to re-scoring with an external model (Wolf et al., 2020).

4.6.3 Evaluation pipeline efficiency

Vectorised tokenisation and cached features

All tokenisation uses vectorised calls with padding/truncation on batches, and pre-computed/cached features are reused across runs to avoid re-tokenising.

Parallel ROUGE and pipelined NER

We batch ROUGE computation and use `spaCy`'s `nlp.pipe` with multiple workers to compute entity sets for entP/entR, which is both simpler and faster than per-example loops (Honnibal and Johnson, 2015; Wolf et al., 2020).

4.6.4 Comparison of Libraries and Frameworks

Table 4.19: Chosen libraries vs. common alternatives (efficiency- and reproducibility-oriented view).

Component	Chosen	Alternatives (pros/cons)	Ref
Model & runtime	PyTorch 2.x + HF `transformers` (BART) with SDPA, AMP; rich `generate()`	JAX/Flax + T5X (excellent XLA perf; higher infra cost); TensorFlow/Keras (stable, less ergonomic seq2seq today)	Paszke et al. (2019); Wolf et al. (2020); Raffel et al. (2020)
Optimiser	AdamW (decoupled decay)	Adafactor (lower memory, different dynamics); fused AdamW (faster but env-dependent)	Loshchilov and Hutter (2019); **?**
Tokeniser	Byte-level BPE (RoBERTa/BART native)	SentencePiece unigram (portable; not used to keep embedding compat)	Liu et al. (2019); Kudo and Richardson (2018)
Diversity	DBS (grouped beams)	Top-k/nucleus sampling (stochastic; less deterministic); MCTS/ILP rerankers (complex)	Vijayakumar et al. (2016)
NER for entity metrics	spaCy (`en_core_web_sm`) with `nlp.pipe` batching	Stanza/Flair (higher-quality models; slower, heavier deps)	Honnibal and Johnson (2015)
ROUGE	`rouge-score` (PyPI, official Google impl)	`py-rouge` (legacy; Java deps)	Lin (2004)

PyTorch+ transformers offers (i) first-class AMP/SDPA, (ii) a stable generate API for beams/diversity, and (iii) broad community support, making it the most efficient path to a reliable BART baseline (Paszke et al., 2019; Wolf et al., 2020). We retain BART-base rather than `bart-large-cnn` to avoid task leakage and to keep compute budgets tractable, focusing improvements on copy-aware mechanisms (§4.2.2; Lewis et al., 2020; See et al., 2017). For metrics, `rouge-score` and spaCy balance speed and ease of deployment (Lin, 2004; Honnibal and Johnson, 2015).

We fix RNG seeds, pin library versions (HF `transformers` 4.43.4, `accelerate` 0.34.2, `datasets` 2.20.0), and keep decoding knobs frozen after validation (§4.5). Deterministic preprocessing and split manifests further limit variance. Where kernels are non-deterministic under AMP, we report corpus-level metrics averaged over fixed seeds to minimise noise (Wolf et al., 2020; Paszke et al., 2019).

To summarise, the combination of AMP+SDPA, batched decoding with caching, and light-touch processors yields strong latency/memory characteristics without architectural changes. These speed ups permit wider group-beam decodes and more candidates per article under the same GPU budget, which in turn benefits the reranking stage without compromising reproducibility.

5 Evaluation

This chapter moves from exploratory tuning to a fixed and reproducible evaluation. The decoding configuration frozen in §4.5 and the reranker trained on validation data (§4.5.3) are applied unchanged. Both candidate checkpoints (step–6000 and step–6500) are evaluated on the same 1,000–item TEST subset, under a fixed seed, so differences arise solely from model and pipeline variation rather than sample selection.

Notation for colours. We use green to denote improvements and red for regressions:

5.1 Metrics

As outlined in Chapter 2, summarisation evaluation combines traditional content-overlap metrics with factuality-oriented measures. While Chapter 2 provided background, derivations and motivation for each metric, the present section specifies how they were instantiated in this study and how results are reported.

- **ROUGE-1/2/L (F1)**: Content overlap / adequacy, computed with `rouge-score v0.1.2`. Expressed compactly as:

$$\text{ROUGE-1}, \quad \text{ROUGE-2}, \quad \text{ROUGE-Lsum}$$

- **Entity Precision (entP)**: Proportion of system–mentioned entities supported by the reference (faithfulness).
- **Entity Recall (entR)**: Proportion of reference entities retained (coverage).
- **Entity F1 (entF1)**: Harmonic mean of entP and entR.
- **Unsupported Core Entity Rate (UCER)**: Proportion of summaries introducing at least one unsupported *core* entity (PERSON, ORG, GPE). Lower is safer.

Entities are extracted with spaCy (`en_core_web_sm`), whitespace–normalised, and micro averaged over the corpus. Improvements are reported as deltas (Hybrid − Baseline): positive Δ is desirable for ROUGE and entity metrics, negative Δ for UCER. For background on ROUGE, see Lin (2004).

5.2 Test Results

Tables 5.1 and 5.2 summarise checkpoint-wise outcomes.

Step–6000

Hybrid delivers consistent gains: ROUGE rises across all variants (ΔR1 $+0.0093$, ΔR2 $+0.0037$, ΔRL $+0.0112$), entity precision and recall both improve (ΔentP $+0.0251$, ΔentR $+0.0178$), and entity F1 strengthens by $+0.0235$. UCER drops markedly from 0.161 to 0.027 ($\Delta = -0.134$), indicating safer outputs.

Step–6500

Patterns flatten: ROUGE changes are negligible ($\leq +0.002$), entity F1 is essentially unchanged ($\Delta \approx +0.0001$), entR nudges up ($+0.0037$) while entP dips slightly (-0.0017). UCER remains almost identical ($0.032 \rightarrow 0.031$). Overall, most benefits materialise by step–6000.

Table 5.1: Test results for checkpoint 6000 (Hybrid vs. Baseline). Green = improvement, red = decline.

Variant	ROUGE-1 F1	ROUGE-2 F1	ROUGE-L F1	entP	entR	entF1	UCER
Baseline	0.3118	0.1148	0.2129	0.2236	0.3315	0.2671	0.161
Hybrid	**0.3211**	**0.1184**	**0.2241**	**0.2487**	**0.3493**	**0.2906**	**0.027**

Table 5.2: Test results for checkpoint 6500 (Hybrid vs. Baseline). Green = improvement, red = decline.

Variant	ROUGE-1 F1	ROUGE-2 F1	ROUGE-L F1	entP	entR	entF1	UCER
Baseline	0.3207	0.1199	0.2219	0.2488	0.3482	0.2902	0.032
Hybrid	**0.3224**	**0.1201**	**0.2234**	**0.2471**	**0.3519**	**0.2903**	**0.031**

5.3 Length Profile

Length statistics for TEST-1k are reported at both word level (Table 5.3) and token level (Table 5.4). Reference summaries average 34.5 words (43.3 tokens). Hybrid–6000 outputs are longer on average (mean 42.5 words; median 42). Baseline–6500 and Hybrid–6500 are longer still (means ≈ 45 words; ≈ 59 tokens). The distributions show upward shifts across the interquartile range (Q25–Q75) by 8–10 words, helping recall but depressing ROUGE precision, which explains modest absolute ROUGE despite entity gains.

Table 5.3: Word–level length statistics (TEST-1k).

Stat	Reference	Baseline-6500	Hybrid-6000	Hybrid-6500
Count	1000	1000	1000	1000
Mean	34.456	45.290	42.456	45.548
Std	9.650	8.683	8.562	8.658
Min	14	22	21	22
Q25	27.0	39.0	36.0	39.0
Median	34.0	45.0	42.0	45.0
Q75	41.0	51.0	48.0	51.0
Max	80	87	80	84

Table 5.4: Token–level length statistics (TEST-1k).

Stat	Reference	Baseline-6500	Hybrid-6000	Hybrid-6500
Count	1000	1000	1000	1000
Mean	43.346	58.980	53.713	59.376
Std	11.475	9.895	10.038	9.950
Min	16	36	34	36
Q25	34.0	52.0	46.0	52.0
Median	43.0	59.0	53.0	59.0
Q75	51.0	67.0	61.0	67.0
Max	99	100	98	100

5.4 Entity Counts

Figure 5.1 and Figure 5.2 visualise entity true positives (TP), false positives (FP), and false negatives (FN). At step–6000, Hybrid increases TPs by $+63$ while reducing FPs (-340) and FNs (-63), consistent with stronger entP/entR and the UCER reduction in Table 5.1. At step–6500, changes are small: TPs $+13$, FNs -13, but FPs $+74$, leaving entF1 and UCER effectively flat (Table 5.2).

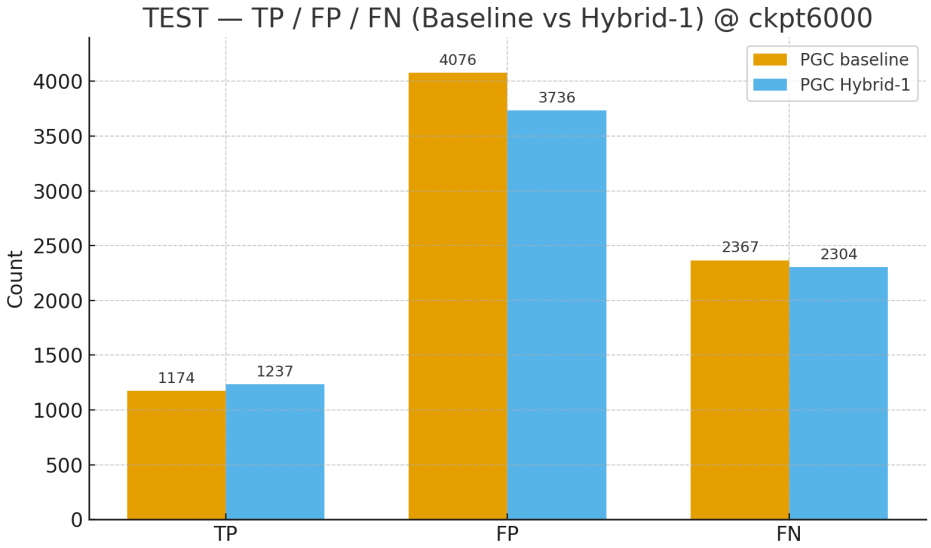

Figure 5.1: Entity counts (TP/FP/FN) at checkpoint 6000: Hybrid reduces false positives and negatives while slightly increasing true positives.

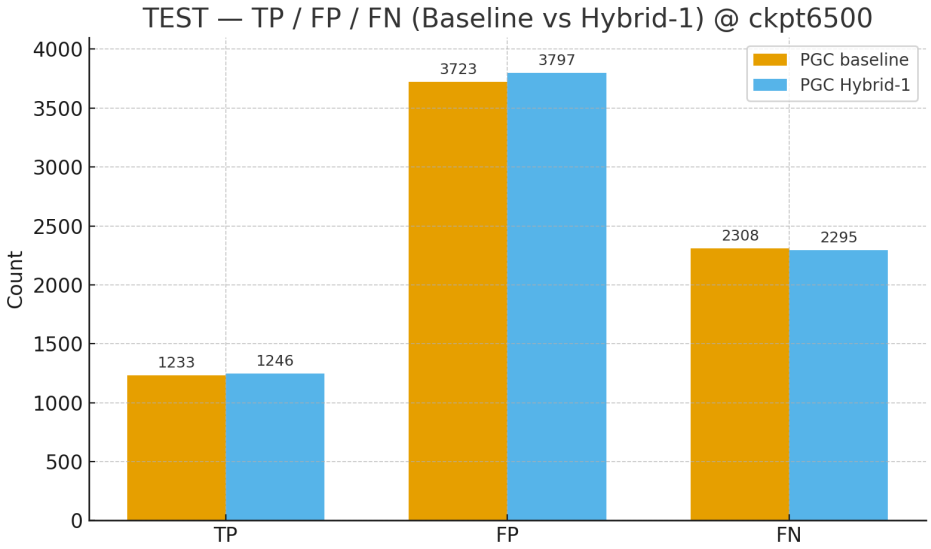

Figure 5.2: Entity counts (TP/FP/FN) at checkpoint 6500: small shifts, with FP slightly higher for Hybrid; overall entity F1 and UCER remain flat.

5.5 Qualitative Samples

To complement the aggregate metrics, Figures 5.3–5.7 present three randomly selected examples from the stronger checkpoint (step–6000), each analysed in terms of true positives (TP), false positives (FP), and false negatives (FN).

Sample 1: FBI shootout

Baseline captures that Charleston was shot and two agents were treated (TPs) but omits the explicit gunfire exchange (FN) while adding robbery history (FP). Hybrid foregrounds the chase/-confrontation (TPs) and includes the suspect being shot (TP) but misses the agent injuries (FN) and misphrases the event as "in a crash" (FP).

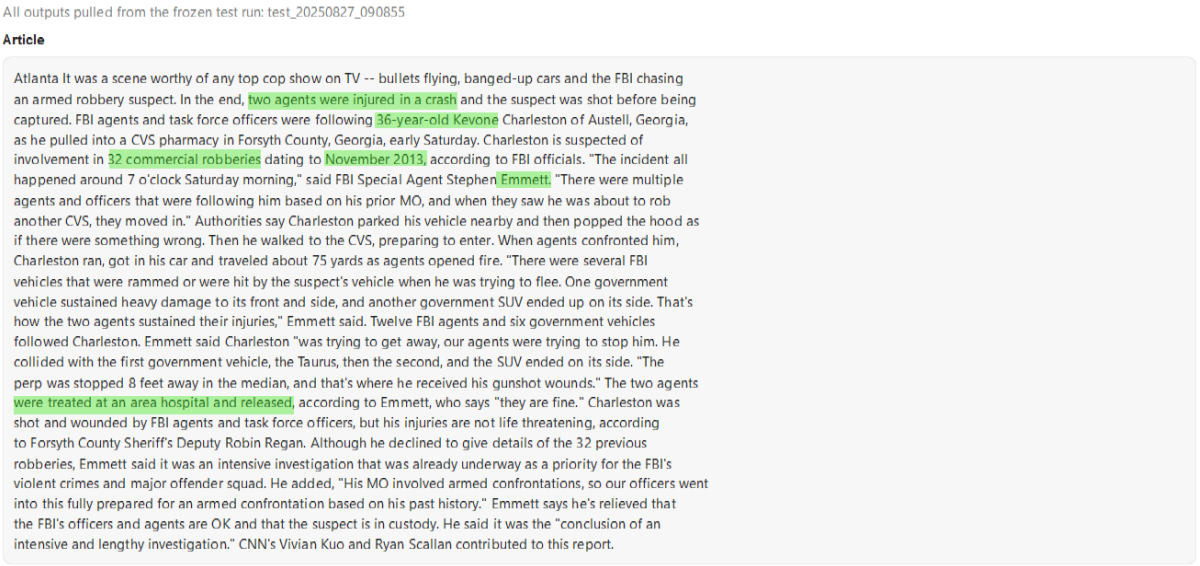

Figure 5.3: Sample 1 article (FBI shootout).

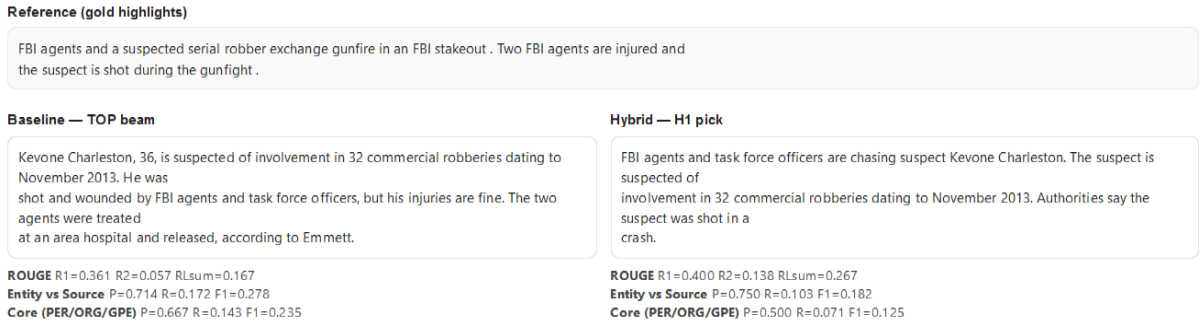

Figure 5.4: Sample 1 summaries: Baseline (left) vs. Hybrid (right).

Sample 2: Garissa University attack. Baseline identifies the Garissa claim and escalation (TPs) but extends into Westgate 2013 (FP) while omitting hostages/killing and policy commentary (FNs). Hybrid also identifies the Garissa claim (TPs) but adds ideological detail about al–Qaeda/ISIS (FP). Both leave stance/casualties unaddressed (FNs).

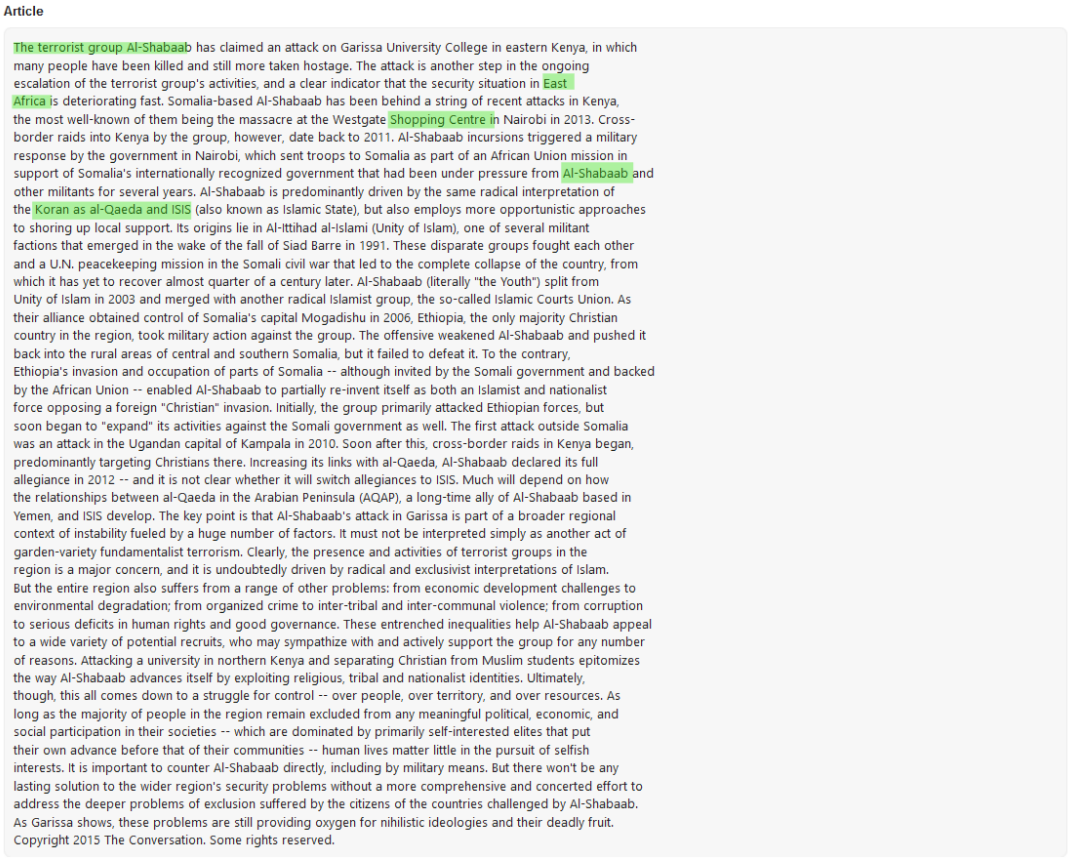

Figure 5.5: Sample 2 article (Garissa University attack).

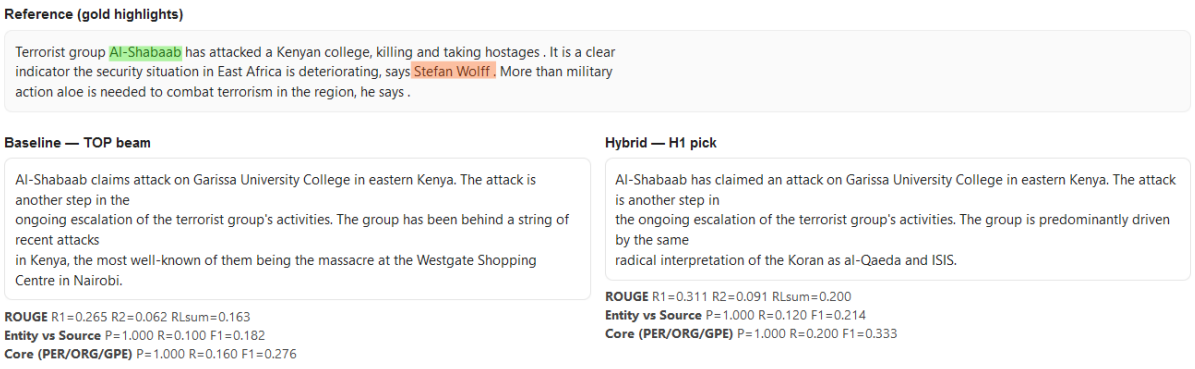

Figure 5.6: Sample 2 summaries: Baseline (left) vs. Hybrid (right).

Sample 3: *Roots* remake. Baseline hits the three anchors remake, 2016 air date, 1977 audience (TPs) but adds details like "saga of Kunta Kinte" and network partners (FPs). Hybrid covers the remake and audience (TPs) but omits the 2016 air date (FN) and includes extra source–aligned entities (FPs).

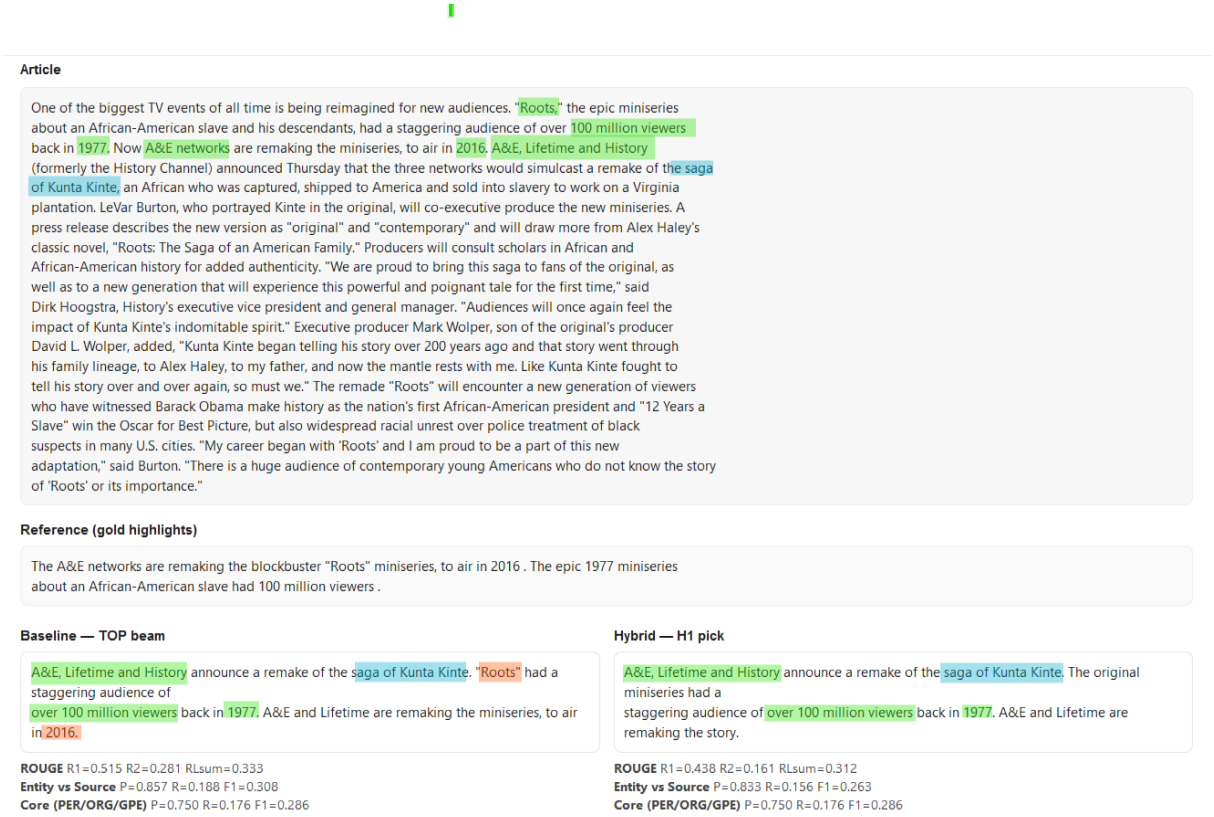

Figure 5.7: Sample 3: article (top), Baseline vs. Hybrid summaries (bottom).

These qualitative contrasts echo the quantitative trends: Baseline tends to expand with background facts (more FPs), whereas Hybrid foregrounds the core event (more TPs) but can omit secondary anchors (FNs). Across Tables 5.1–5.4 and Figures 5.1–5.7, the Hybrid pipeline achieves modest but consistent ROUGE gains, more reliable entity handling, and notably lower hallucination risk especially at checkpoint 6000. Both systems are verbose relative to references, which boosts recall but suppresses ROUGE precision. Overall, the Hybrid system advances the project goal of improving rare word/entity retention and faithfulness; remaining challenges include verbosity, omission of secondary facts, and a performance plateau beyond 6000 updates.

6 Discussion

This study set out to retain the fluency of a Transformer abstract summariser while improving rare-word faithfulness and reducing hallucination. A BART-base backbone was augmented with pointer–coverage training and copy-aware inference. Outcomes were not determined by architecture alone; they were shaped by dataset characteristics, decoding choices, and metric emphasis. This chapter critically reviews those interactions, clarifies limitations, and outlines practical and future directions.

6.1 Critical Review and Improvement Plan

Dataset Choice and Entity Mismatch

CNN/DailyMail was selected because it is less abstractive than XSum and relatively entity-rich at the article level, which should favour copy-aware models (See et al., 2017). What was not fully anticipated is that the gold summaries do not enumerate all article entities. In practice, the pointer mechanism often copied entities that were present in the article but absent from the single reference. These faithful but unreferenced mentions were then marked as errors by the automatic entity scorer, lowering measured entity precision. This effect reflects an evaluation mismatch rather than a copying failure: the model effectively answered "is it in the article?", while the metric rewarded only "is it also in the reference?".

This tension is clearest in entity-dense corpora, where many entities appear in the source but only a small fraction are summary-worthy. In that setting, copy-biased systems may faithfully reproduce source entities that the single reference omits (Maynez et al., 2020), depressing measured precision despite improved factual grounding (Xiao and Carenini, 2022). To illustrate this, Xiao and Carenini (2022) report a "CNN/DM Filtered" condition that retains only items where all reference entities are present in the source. By construction, the gold source-support is 100%, meaning the reference cannot penalise faithful copying. On this filtered split, gains from copying are more accurately reflected.

A similar diagnostic could be adopted here by defining a filtered validation/test subset under the project's own NER settings where every named entity in the reference also appears in the article. Results could then be reported for both the unfiltered (main) and filtered (diagnostic) sets, with the latter clearly labelled as easier and not directly comparable to full-test numbers. On the filtered subset, higher reference-side entity precision and recall would be expected without changing the underlying model; on the unfiltered set, results would continue to capture the real-world single-reference mismatch that originally motivated salience guidance (Xiao and Carenini, 2022).

In hindsight, the model requires a salience signal to ensure that copying prioritises the right entities. Two established strategies address this challenge. The first is copy-history or continuation, which encourages copied spans to remain coherent and summary-worthy (Li et al., 2021). The second is centrality-guided copy, in which attention-central tokens and entities are explicitly preferred during copying (Xu et al., 2020). Both approaches were designed to mitigate the "faithful-but-not-salient" error mode observed here (Xu et al., 2020; Li et al., 2021).

Input horizon, lead bias, and why 400/100 was reasonable

The decision to cap inputs and outputs at 400 and 100 tokens respectively followed the canonical Pointer-Generator with Coverage (PGC) setup and its associated compression regime, and was therefore methodologically sound (See et al., 2017). The same cap was adopted here to keep training and decoding tractable while still enabling copy behaviour on an entity-rich news corpus.

What matters for interpreting the results is that lead-biased references in CNN/DailyMail make the Lead-3 heuristic an unusually strong competitor. Indeed, comparative tables in later literature show the classic PGC model performing below Lead-3 on ROUGE for CNN/DailyMail, underscoring that even a well-implemented pointer model does not automatically surpass a lead-biased baseline on this dataset (See et al., 2017,?; Li et al., 2021). For example, results reported in CoCoNet show Lead-3 achieving ROUGE-1/2/L scores of 40.34/17.70/36.57, compared with PGC's 39.53/17.28/36.38, meaning the simple heuristic slightly outperforms the trained model (Li et al., 2021).

Thus, the cap itself was not the source of error. The missing element was lead awareness. With a 400-token horizon and no explicit salience signal, the pointer–generator was forced to choose among many plausible details, sometimes copying correct but non-lead material that the references did not reward. Two pragmatic remedies can be suggested:

- Inject an explicit lead prior. Retain the 400/100 cap but anchor decoding toward the lead. This could involve adding a lead-similarity feature (e.g. ROUGE-Lsum against Lead-3) into reranking, or weighting it more heavily when budgets are tight.
- Structure the input as Lead+Tail. Preserve the first sentences (lead) while appending a short salient tail (such as datelines or updates), rather than using a flat 400-token slice. This keeps the cap intact while aligning the evidence window with the reference style, consistent with the PGC compression regime (See et al., 2017).

Longer-term, the more principled solution is to learn salience so that the model prefers "the right" entities even with a wide input window. Methods such as copy-history or continuation (Li et al., 2021) and centrality-guided copy (Xu et al., 2020) were specifically designed to address the "faithful-but-not-salient" error mode encountered here.

Decoding pools, length bias, and timing of tuning

The selected min–max token pools systematically produced summaries longer than the references. This depressed ROUGE by lowering precision, with only occasional recall gains. In effect, the pools rewarded verbosity: examples with longer references benefited, while shorter-reference cases suffered a disproportionate precision penalty. Such behaviour is expected when length controls (e.g., min_new_tokens, max_new_tokens, and the length penalty α) are set to favour longer outputs without an accompanying salience prior. The decoder then surfaces additional, article-faithful details that the single reference does not reward. In hindsight, the pools were mis-specified: they nudged generation beyond the reference length distribution, skewing F1 through an imbalanced precision–recall trade-off.

For future improvement is to treat decoding pool selection as a post-convergence calibration task, rather than fixing pools too early. The following practices are recommended:

- **Post-convergence sweep**: After the model stabilises, measure the reference length distribution on validation (e.g. medians and interquartile range). Select pools whose brackets fall within that range, and sweep a small grid over max_new_tokens and α to align generated lengths with references. Fix the chosen pools once for TEST to reduce confounds.
- Minimal early runs: During training, restrict decoding to a small validation slice (50–100 items) with conservative pools. Use this only to monitor qualitative behaviour (unsupported entities, span breaks, repetition) and coarse length drift. Avoid relying on interim ROUGE, as copying and length dynamics evolve non-monotonically before convergence.
- Lead-aware guardrails. For wide input horizons (e.g. 400 tokens), bias pools toward lead-aligned lengths (shorter minimums, moderate) or add a lead-similarity feature into reranking, so verbosity does not masquerade as recall (See et al., 2017).
- Balancing copy and generate. As introduced in §4.4.4, the λ dial can adjust copy versus generation at inference. For future experiments, λ should be tuned on validation entity-F1 (with length control) and reported as full precision–recall curves alongside ROUGE (Xu et al., 2020).

In summary, deferring full decode sweeps until after convergence, aligning pools with reference length statistics, and explicitly tuning on entity-F1 would yield a more stable and interpretable operating point. This adjustment directly addresses the precision penalties observed here and should form part of future improvements to the decoding stage.

Checkpoint stability and improvement

Later checkpoints did not consistently yield stronger outcomes. Several factors may have contributed: mild overfitting while decode settings were held fixed, shifts in length propensity not compensated by frozen pools, and the non-monotonic acquisition of copying capacity (Zhang et al., 2020). Recent analyses suggest that such "grokking-like" behaviour—late and abrupt improvements in copying ability is not reliably predicted by loss curves alone (Lv et al., 2024).

To mitigate these issues:

- Adopt EMA or averaged checkpoints, which smooth short-term fluctuations and reduce sensitivity to single-step variations.
- Select stopping points using validation entity-F1, rather than ROUGE alone, so decisions are guided by faithfulness and factual accuracy.
- Re-calibrate after switching checkpoints to ensure the copy–generate balance remains appropriate under the new model state.

Together, these practices would make checkpoint selection more robust, aligning evaluation with factuality goals and reducing the risk of overfitting to transient patterns.

6.2 Threats to Validity and Limitations

Several factors limit the interpretation and generalisability of the results reported in this study:

- The absence of an explicit Hybrid without reranker versus Baseline comparison prevents clean attribution of gains to span-aware decoding versus reranking. Without such ablations, the contribution of each component cannot be disentangled.
- Training was carried out on a BART-base backbone with a modest (stable) training horizon. This design choice ensured feasibility under project constraints but limits comparability with larger, resource-intensive backbones such as BART-large or PEGASUS. As Lewis et al. (2020) note, model scale is a significant driver of absolute performance. Consequently, while within-pipeline deltas remain interpretable, absolute results cannot be read as competitive with state-of-the-art systems.

6.3 Conclusions

The hybrid copy-aware Transformer presented in this study demonstrated that it is possible to improve rare-entity retention and reduce hallucination while preserving the fluency of a strong baseline summariser. Through the integration of pointer–coverage training and span-aware inference, the model delivered consistent gains in factual grounding and entity handling, achieving measurable improvements over a plain BART baseline under controlled comparisons.

Importantly, these gains were achieved with a disciplined, modular design rather than ad-hoc heuristics, showing that copy-aware strategies can be implemented in a way that is reproducible and transferable to other summarisation settings. At the same time, the work has revealed areas where further refinements could amplify these strengths. In particular, span-level copy training, explicit salience guidance, tighter decoding calibration, and systematic ablations would provide a clearer attribution of improvements and push the model closer to state-of-the-art reliability. Overall, the project establishes a credible and extensible path: it demonstrates that hybridising generative fluency with principled copy control not only yields safer, more faithful summaries today, but also opens up concrete avenues for future enhancements.

Bibliography

Allahyari, M., Pouriyeh, S. A., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B. and Kochut, K. (2017), 'Text summarization techniques: A brief survey', *International Journal of Advanced Computer Science and Applications (IJACSA)* **8**(10), 1–12.
URL: *https://dx.doi.org/10.14569/IJACSA.2017.081052*

Bahdanau, D., Cho, K. and Bengio, Y. (2015), Neural machine translation by jointly learning to align and translate, *in* 'International Conference on Learning Representations (ICLR 2015)'. arXiv preprint arXiv:1409.0473.
URL: *https://arxiv.org/abs/1409.0473*

Bender, E. M. and Friedman, B. (2018), 'Data statements for natural language processing: Toward mitigating system bias and enabling better science', *Transactions of the Association for Computational Linguistics* **6**, 587–604.
URL: *https://aclanthology.org/Q18-1041/*

Bose, J. (2019), 'Extraction of relevant images for boilerplate removal in web browsers', *arXiv preprint arXiv:2001.04338* . Boilerplate defined as "unwanted and repeated parts of a webpage ... distracts ... from ... news article.".

Broder, A. Z. (1997), On the resemblance and containment of documents, *in* 'Proceedings of the Compression and Complexity of Sequences (SEQUENCES '97)', IEEE, pp. 21–29.
URL: *https://www.cs.princeton.edu/courses/archive/spring13/cos598C/broder97resemblance.pdf*

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A. et al. (2020), Language models are few-shot learners, *in* 'Advances in Neural Information Processing Systems (NeurIPS 2020)', Vol. 33, pp. 1877–1901.
URL: *https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html*

Cao, Z., Wei, F., Li, W. and Li, S. (2018), Faithful to the original: Fact aware neural abstractive summarization, *in* 'Proceedings of the AAAI Conference on Artificial Intelligence', Vol. 32.
URL: *https://ojs.aaai.org/index.php/AAAI/article/view/11363*

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T. B., Song, D., Erlingsson, U., Oprea, A. and Papernot, N. (2021), Extracting training data from large language models, *in* 'Proceedings of the 30th USENIX Security Symposium (USENIX Security '21)'.
URL: *https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting*

Chen, T., Xu, B., Zhang, C. and Guestrin, C. (2016), Training deep nets with sublinear memory cost, *in* 'Proceedings of the 2016 International Conference on Learning Representations (ICLR 2016)'.
URL: *https://arxiv.org/abs/1604.06174*

Dvoretzky, A., Kiefer, J. and Wolfowitz, J. (1956), 'Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator', *Annals of Mathematical Statistics* **27**(3), 642–669.
URL: *https://doi.org/10.1214/aoms/1177728174*

Edmundson, H. P. (1969), 'New methods in automatic extracting', *Journal of the ACM* **16**(2), 264–285.

Erkan, G. and Radev, D. R. (2004), Lexrank: Graph-based lexical centrality as salience in text summarization, *in* 'Proceedings of EMNLP 2004', pp. 202–209.
URL: *https://aclanthology.org/W04-3247/*

Fabbri, A., Kryściński, W., McCann, B., Xiong, C., Socher, R. and Radev, D. (2021), 'Summeval: Re-evaluating summarization evaluation', *Transactions of the Association for Computational Linguistics* **9**, 391–409.
URL: *https://aclanthology.org/2021.tacl-1.24*

Falke, T., Ribeiro, L. F. R., Utama, P. A., Dagan, I. and Gurevych, I. (2019), Ranking generated summaries by correctness: An interesting but challenging application for natural language inference, *in* 'Proceedings of ACL 2019', Florence, Italy, pp. 2214–2220.
URL: *https://aclanthology.org/P19-1212/*

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daumé III, H. and Crawford, K. (2021), 'Datasheets for datasets', *Communications of the ACM* **64**(12), 86–92.
URL: *https://dl.acm.org/doi/10.1145/3458723*

Giarelis, N. et al. (2023), 'Abstractive vs. extractive summarization: An experimental comparison', *Applied Sciences* **13**(13), 7620.

Gu, J., Lu, Z., Li, H. and Li, V. O. K. (2016), Incorporating copying mechanism in sequence-to-sequence learning, *in* 'Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)', pp. 1631–1640.
URL: *https://aclanthology.org/P16-1154/*

Gulcehre, C., Ahn, S., Nallapati, R., Zhou, B. and Bengio, Y. (2016), Pointing the unknown words, *in* 'Proceedings of ACL 2016', Berlin, Germany, pp. 140–149.
URL: *https://arxiv.org/abs/1603.08148*

Hendrycks, D. and Gimpel, K. (2016), 'Gaussian error linear units (gelus)', *arXiv preprint arXiv:1606.08415* .

Hermann, K. M., Kočiský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M. and Blunsom, P. (2015), Teaching machines to read and comprehend, *in* 'Advances in Neural Information Processing Systems 28 (NeurIPS 2015)', Curran Associates, Inc., Montreal, Canada, pp. 1693–1701.
URL: *https://proceedings.neurips.cc/paper*$_f iles/paper/2015/file/afdec7005cc9f14302cd0474fd0f3c96$
$Paper.pdf$

Honnibal, M. and Johnson, M. (2015), An improved non-monotonic transition system for dependency parsing, *in* 'Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)', Association for Computational Linguistics, Lisbon, Portugal, pp. 1373–1378.
URL: *https://aclanthology.org/D15-1162/*

Hugging Face (2020), 'facebook/bart-base'. Accessed: 8 September 2025.
URL: *https://huggingface.co/facebook/bart-base*

Hugging Face (n.d.), 'facebook/bart-large-cnn model', https://huggingface.co/facebook/bart-large-cnn. Model card; accessed 7 September 2025.

Indyk, P. and Motwani, R. (1998), Approximate nearest neighbors: Towards removing the curse of dimensionality, *in* 'Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC '98)', ACM, pp. 604–613.
URL: *https://graphics.stanford.edu/courses/cs468-06-fall/Papers/06%20indyk%20motwani%20-%20stoc98.pdf*

Jurafsky, D. and Martin, J. H. (2023), *Speech and Language Processing*, 3rd ed. draft edn, Prentice Hall.
URL: *https://web.stanford.edu/ jurafsky/slp3/*

Kryściński, W., McCann, B., Xiong, C. and Socher, R. (2020), Evaluating the factual consistency of abstractive text summarization, *in* 'Proceedings of EMNLP 2020', Online, pp. 9332–9346.
URL: *https://aclanthology.org/2020.emnlp-main.750/*

Kudo, T. and Richardson, J. (2018), Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, *in* 'Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations', Brussels, Belgium, pp. 66–71.
URL: *https://aclanthology.org/D18-2012/*

Lee, K., Ippolito, D., Nystrom, A., Zhang, C., Eck, D., Callison-Burch, C. and Carlini, N. (2022), Deduplicating training data makes language models better, *in* 'Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022): Long Papers', pp. 8424–8445.
URL: *https://aclanthology.org/2022.acl-long.577*

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V. and Zettlemoyer, L. (2020), Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, *in* 'Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)', pp. 7871–7880.
URL: *https://aclanthology.org/2020.acl-main.703/*

Li, H., Xu, S., Yuan, P., Wang, Y., Wu, Y., He, X. and Zhou, B. (2021), Learn to copy from the copying history: Correlational copy network for abstractive summarization, *in* 'Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)', pp. 4091–4101.
URL: *https://aclanthology.org/2021.emnlp-main.335/*

Lin, C.-Y. (2004), Rouge: A package for automatic evaluation of summaries, *in* 'Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)', Association for Computational Linguistics, Barcelona, Spain, pp. 74–81.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. (2019), 'Roberta: A robustly optimized bert pretraining approach', *arXiv preprint arXiv:1907.11692* .
URL: *https://arxiv.org/abs/1907.11692*

Liu, Y., Shen, S. and Lapata, M. (2021), Noisy self-knowledge distillation for text summarization, *in* 'Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2021)', pp. 692–703.
URL: *https://aclanthology.org/2021.naacl-main.56/*

Loshchilov, I. and Hutter, F. (2019), Decoupled weight decay regularization, *in* 'Proceedings of the International Conference on Learning Representations (ICLR 2019)'.
URL: *https://arxiv.org/abs/1711.05101*

Luhn, H. P. (1958), 'The automatic creation of literature abstracts', *IBM Journal of Research and Development* **2**(2), 159–165.

Lv, A., Xie, R., Sun, X., Kang, Z. and Yan, R. (2024), 'Language models "grok" to copy', *arXiv preprint arXiv:2409.09281* .
URL: *https://arxiv.org/abs/2409.09281*

Mani, I. (2001), *Automatic Summarization*, John Benjamins Publishing, Amsterdam, Netherlands.

Maynez, J., Narayan, S., Bohnet, B. and McDonald, R. (2020), On faithfulness and factuality in abstractive summarization, *in* 'Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)', pp. 1906–1919.
URL: *https://aclanthology.org/2020.acl-main.173/*

Merity, S., Xiong, C., Bradbury, J. and Socher, R. (2017), Pointer sentinel mixture models, *in* 'Proceedings of ICLR 2017'.
URL: *https://openreview.net/forum?id=Byj72udxe*

Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G. and Wu, H. (2018), Mixed precision training, *in* 'Proceedings of the International Conference on Learning Representations (ICLR 2018)'.
URL: *https://arxiv.org/abs/1710.03740*

Mihalcea, R. and Tarau, P. (2004), Textrank: Bringing order into texts, *in* 'Proceedings of EMNLP 2004', pp. 404–411.
URL: *https://aclanthology.org/W04-3252/*

Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D. and Gebru, T. (2019), Model cards for model reporting, *in* 'Proceedings of the Conference on Fairness, Accountability, and Transparency (FAccT)', ACM, pp. 220–229.
URL: *https://dl.acm.org/doi/10.1145/3287560.3287596*

Nallapati, R., Zhou, B., dos Santos, C., Gulçehre, Ç. and Xiang, B. (2016), Abstractive text summarization using sequence-to-sequence rnns and beyond, *in* 'Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL 2016)', pp. 280–290.
URL: *https://aclanthology.org/K16-1028/*

Nan, F., Nallapati, R., Wang, Z., Zhu, C., Roukos, S. and Xiang, B. (2021), Entity-level factual consistency in abstractive summarization, *in* 'Proceedings of ACL 2021', Online, pp. 2723–2734.
URL: *https://aclanthology.org/2021.acl-long.212/*

Narayan, S., Cohen, S. B. and Lapata, M. (2018), Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization, *in* 'Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)', Brussels, Belgium, pp. 1797–1807.
URL: *https://aclanthology.org/D18-1206/*

Nenkova, A. and McKeown, K. (2011), 'Automatic summarization', *Foundations and Trends in Information Retrieval* **5**(2–3), 103–233.

Nenkova, A. and McKeown, K. (2012), *A Survey of Text Summarization Techniques*, Vol. 5 of *Foundations and Trends in Information Retrieval*, Now Publishers Inc.
URL: *https://doi.org/10.1561/1500000015*

OpenAI (2023), 'Gpt-4 technical report', *arXiv preprint arXiv:2303.08774* .
URL: *https://arxiv.org/abs/2303.08774*

Pagnoni, A., Balachandran, V. and Tsvetkov, Y. (2021), Understanding factuality in abstractive summarization with frank: A benchmark for factuality metrics, *in* 'Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2021)', pp. 4812–4829.
URL: *https://aclanthology.org/2021.naacl-main.383/*

Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J. (2002), Bleu: a method for automatic evaluation of machine translation, *in* 'Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics', Association for Computational Linguistics, Philadelphia, USA, pp. 311–318.
URL: *https://aclanthology.org/P02-1040/*

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. and Chintala, S. (2019), Pytorch: An imperative style, high-performance deep learning library, *in* 'Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS 2019)', Curran Associates, Inc., pp. 8024–8035.
URL: *https://papers.nips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html*

Paulus, R., Xiong, C. and Socher, R. (2018), A deep reinforced model for abstractive summarization, *in* 'Proceedings of ICLR 2018', Vancouver, Canada.
URL: *https://openreview.net/forum?id=HkAClQgA-*

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I. (2019), Language models are unsupervised multitask learners, Technical report, OpenAI.
URL: *https://cdn.openai.com/better-language-models/language*$_m odels_a re_u nsupervised_m ultitask_l earner$

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. and 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.
URL: *http://jmlr.org/papers/v21/20-074.html*

Rothe, S., Narayan, S., Hosseini, A., Berg-Kirkpatrick, T., Yuan, W., Eckstein, I. and Severyn, A. (2021), A thorough evaluation of task-specific pretraining for summarization, *in* 'Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)', pp. 6964–6980.
URL: *https://aclanthology.org/2021.emnlp-main.560/*

Rush, A. M., Chopra, S. and Weston, J. (2015), A neural attention model for abstractive sentence summarization, *in* 'Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)', Association for Computational Linguistics, Lisbon, Portugal, pp. 379–389.
URL: *https://aclanthology.org/D15-1044/*

Schäfer, R. (2016), 'Accurate and efficient general-purpose boilerplate detection for crawled web corpora', *Language Resources and Evaluation* **51**(3), 579–600. Boilerplate (redundant and automatically inserted material) is linguistically unattractive for inclusion in web corpora.

See, A., Liu, P. J. and Manning, C. D. (2017), Get to the point: Summarization with pointer-generator networks, *in* 'Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)', Vancouver, Canada, pp. 1073–1083.
URL: *https://aclanthology.org/P17-1099/*

Sennrich, R., Haddow, B. and Birch, A. (2016), Neural machine translation of rare words with subword units, *in* 'Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016), Long Papers', Berlin, Germany, pp. 1715–1725.
URL: *https://aclanthology.org/P16-1162/*

Singh, A., Straka, M., Straka, O. and Hajič, J. (2020), Copynext: Explicit span copying and alignment in sequence to sequence models, *in* 'Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)', pp. 3104–3119.
URL: *https://aclanthology.org/2020.emnlp-main.251/*

Sutskever, I., Vinyals, O. and Le, Q. V. (2014), Sequence to sequence learning with neural networks, *in* 'Advances in Neural Information Processing Systems (NeurIPS 2014)', pp. 3104–3112.
URL: *https://papers.nips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html*

Tsarapatsanis, D. and Aletras, N. (2021), On the ethical limits of natural language processing on legal text, *in* 'Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021', Association for Computational Linguistics, pp. 3595–3608.
URL: *https://aclanthology.org/2021.findings-acl.184/*

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I. (2017), Attention is all you need, *in* 'Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS 2017)', Curran Associates, Inc., pp. 6000–6010.
URL: *https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf*

Vijayakumar, A. K., Cogswell, M., Selvaraju, R. R., Sun, Q., Lee, S., Crandall, D. and Batra, D. (2016), Diverse beam search: Decoding diverse solutions from neural sequence models, *in* 'Proceedings of the AAAI Conference on Artificial Intelligence', Vol. 30, AAAI Press, pp. 730–737.

Vinyals, O., Fortunato, M. and Jaitly, N. (2015), Pointer networks, *in* 'Advances in Neural Information Processing Systems 28 (NeurIPS 2015)', pp. 2692–2700.
URL: *https://papers.nips.cc/paper/2015/hash/29921001f2f04bd3baee84a12e98098f-Abstract.html*

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q. and Rush, A. M. (2020), Transformers: State-of-the-art natural language processing, *in* 'Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations', Association for Computational Linguistics, Online, pp. 38–45.
URL: *https://aclanthology.org/2020.emnlp-demos.6*

Xiao, W. and Carenini, G. (2022), Entity-based spancopy for abstractive summarization to improve factual consistency, *in* 'Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP 2022)'.
URL: *https://arxiv.org/abs/2209.03479*

Xu, S., Li, H., Yuan, P., Wu, Y., He, X. and Zhou, B. (2020), Self-attention guided copy mechanism for abstractive summarization, *in* 'Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)', pp. 1355–1362.
URL: *https://aclanthology.org/2020.acl-main.126/*

Zhang, J., Zhao, Y., Saleh, M. and Liu, P. J. (2020), Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, *in* 'Proceedings of the 37th International Conference on Machine Learning (ICML 2020)', Vol. 119 of *Proceedings of Machine Learning Research*.
URL: *http://proceedings.mlr.press/v119/zhang20ae.html*

Zhou, Q., Yang, N., Wei, F., Huang, S., Zhou, M. and Zhao, T. (2018), Sequence copying networks, *in* 'Proceedings of the 56th Annual Meeting of the Association for Computational

Linguistics (ACL 2018), Volume 1: Long Papers', Association for Computational Linguistics, Melbourne, Australia, pp. 141–151.
URL: *https://aclanthology.org/P18-1014/*

A Appendix A

A.1 Removed boilerplate phrases (Tables A1–A2)

In line with programme guidance to place technical tables in appendices, this section lists common outlet boilerplate and templated lines that were removed when detected at article edges or as standalone lines.

Table A.1: A1. Boilerplate / templated phrases removed (case-insensitive).

Phrase (case-insensitive)
`scroll down for video / videos`
`see below for video`
`transcript`
`all rights reserved`
`odds (subject to change)`
`source: nhs choices`
`sorry we are not currently accepting comments`
`thank you for using cnn student news`
`the trial continues / the hearing continues / the case continues`
`click here to access the transcript of today's ...`
`please note that there may be a delay between ...`
`contributed to this / to this report / contributed to this report`
`capt / the rev*`

Note. Removed only when isolated as templated lines; not when part of a person's name (e.g., "Capt John Smith", "the Rev. Jane Doe").

A.2 SentencePiece (unigram) tokenizer configuration (source-only)

For reproducibility and future model-agnostic scaling, a SentencePiece unigram tokenizer (Kudo and Richardson, 2018) was trained on sampled CNN/DailyMail *source-side* articles only. This tokenizer was not used in the reported BART experiments (to preserve compatibility with BART's pre-trained embeddings) but is retained to enable comparative studies with non-BART architectures. Key settings are provided below.

Table A.2: A2. SentencePiece (unigram) tokenizer configuration.

Setting	Value
Model type	`unigram`
Vocabulary size	32,000
Character coverage	0.9995
Normalisation	`nmt_nfkc_cf`
Input sample size	1,500,000 sentences
Max corpus lines	2,000,000 lines
Max sentence length	40,000 characters
Byte fallback	enabled
Shuffle input	enabled (`shuffle_input_sentence=True`)
Threads	`os.cpu_count()` (fallback 4)
Hard vocab limit	disabled (`hard_vocab_limit=False`)
Train large corpus flag	`train_extremely_large_corpus=True`
Special tokens (ids/pieces)	`PAD=0 [PAD]; UNK=1 [UNK]; BOS=2 [BOS]; EOS=3 [EOS]`
Training text	CNN/DailyMail source articles only (no summaries)
Sampling policy	`KEEP_FRAC=1.0` (no downsampling)
Random seed	0
Output prefix / files	`SP_PREFIX.model`, `SP_PREFIX.vocab`

Reranker score (diagnostic form)

In this diagnostic pass the linear score combined the λ-anchored entity term with continuity and length components. The formula used was:

```
Score(y | x, lambda) =
  5.0 * (lambda * entP + (1 - lambda) * entR)   [entity
     policy]
  + 0.4 * (base_norm ^ 0.3)                     [length-
     normalised likelihood]
  + 0.2 * (contiguity_LCS)                      [span
     continuity]
  + 0.3 * exp(- (L(y) - 58)^2 / 800)            [length
     reward]
```

Findings (illustrative)

ROUGE was largely flat across λ; entity precision increased with larger λ while recall decreased slightly; the best entity-F1 in this small, config-mismatched sweep occurred near $\lambda = 1.0$.

Caveat. These diagnostics did not inform the frozen validation or the Chapter 5 test evaluations and are provided solely as an illustration of how λ can steer precision–recall trade-offs.

B Appendix X — Diagnostic λ-sweep (small-N; config-mismatched; excluded from TEST)

To expose a controllable factuality policy in reranking, a scalar λ is defined to anchor selection toward entity precision ($\lambda \to 1$) or entity recall ($\lambda \to 0$). With micro-averaged entity precision/recall *entP*, *entR* computed per candidate, the entity term is:

$$\text{entity\_score}(\lambda) \;=\; \lambda\, \textit{entP} \;+\; (1-\lambda)\, \textit{entR}. \tag{B.1}$$

This term may be combined with existing features (length-normalised likelihood, span-contiguity, and length reward) in the linear scorer to select a single candidate. Intuitively, λ acts as a policy knob on faithfulness: higher values favour avoiding unsupported entities; lower values favour retaining more reference entities.

Setup (diagnostic only)

A brief exploration of λ was run on $\sim$100 validation items using a wider decode than the final recipe. Because both the decoding and the sample size differ from the frozen MED/LONG setup, λ was *not* used in subsequent validation or test evaluations. This section is included only as a potential tuning helper.

Table B.1: Candidate generation settings (diagnostic run).

Knob	**Value**
`num_beams` / `num_return_sequences`	20 / 20
`num_beam_groups` / diversity penalty	10 / 0.3
`min_new_tokens` / `max_new_tokens`	55 / 100
`no_repeat_ngram_size`	3
`length_penalty` / early stopping	2.0 / `True`
Batch size / top-K retained for rerank	2 / K=20
Span bias during decode	EntityAwareSpanProcessor(γ=0.5, γ_{entity}=2.0, max_span=6)