Decoupling Segmentation and Scoring: A Unified Framework for Summary-Source Alignment

Sophie Harren[1], Lars Braubach[1], Tom Vincent Peters
[1]Hochschule Bremen, Germany
Correspondence: soph.harren@gmail.com

Abstract

Tracing generated text back to its source is essential for validating the reliability of abstractive summaries. We address this by decomposing the alignment process into unit segmentation and correspondence scoring. This modularity allows for a systematic investigation into how different text granularities interact with various scoring functions. Our evaluation reveals that while large language models (LLMs) offer high semantic precision, their application at scale is limited by their computational costs. To solve this, we propose a hybrid system: an embedding-based pre-filter removes irrelevant pairs before an LLM calculates the final score. This approach effectively balances semantic accuracy with processing efficiency, providing a scalable framework to evaluate the true information origins of LLM outputs.[1]

1 Introduction

Large Language Models (LLMs) have significantly advanced abstractive summarization, enabling the automated distillation of long texts. However, as the length of the summarized texts increases, verifying the accuracy of the generated information becomes exceedingly difficult. This poses an issue, as LLMs have been shown to exhibit positional biases when processing long texts, for example not being able to reliably retrieve statements (Liu et al., 2024) or under-representing information (Ravaut et al., 2024) from the middle of the source material. To validate that a model has utilized the correct information, analyse potential biases, and ensure that generated statements can be traced back to the input document, robust methods for mapping summaries to their sources are required. These so-called *summary source alignments* can also be used to automatically derive training datasets for tasks like grouping information, detecting salient content and fusing source sentences (Ernst et al., 2024).

Tracing the generated text back to the source however can be challenging due to the abstractive nature of summaries generated by LLMs, which hinders the utilization of lexical measures due to their sensitivity to paraphrasing. Meanwhile, "white-box"-interpretation methods are often inadequate: attention weights are rarely available for proprietary models and fail to actually explain results (Jain and Wallace, 2019), while perturbation-based methods (Ribeiro et al., 2016) are computationally prohibitive. Although attempts have been made to approach summary source alignment as a supervised classification task (Ernst et al., 2021), utilize LLMs to generate citations (Gao et al., 2023) or employ Natural Language Inference (NLI) techniques (Zha et al., 2023), research evaluating the information used in summaries usually relies on heuristics based on lexical measures to pick the closest sentences from the source text (Ravaut et al., 2024; Chhabra et al., 2024).

In this paper, we address these limitations by providing a formalization of the summary-source alignment task, offering a unified framework to categorise existing methodologies by decomposing the process into two distinct sub-tasks: segmentation and scoring. Unlike prior research that typically evaluates alignment as a single, fixed system, we systematically decouple these components, allowing for a controlled investigation into how different combinations of segmentation granularity and scoring logic interact. To this end, we evaluate representative methods from both traditional and modern paradigms. Our findings indicate that the optimal choice of granularity can shift depending on the characteristics of the source domain. Finally, we leverage these empirical insights to derive a hybrid system designed to mitigate the inherent trade-offs between computational efficiency and semantic precision.

[1]Source code and dataset used are publicly available at https://github.com/apfelphine/decoupling-summary-source-alignments.

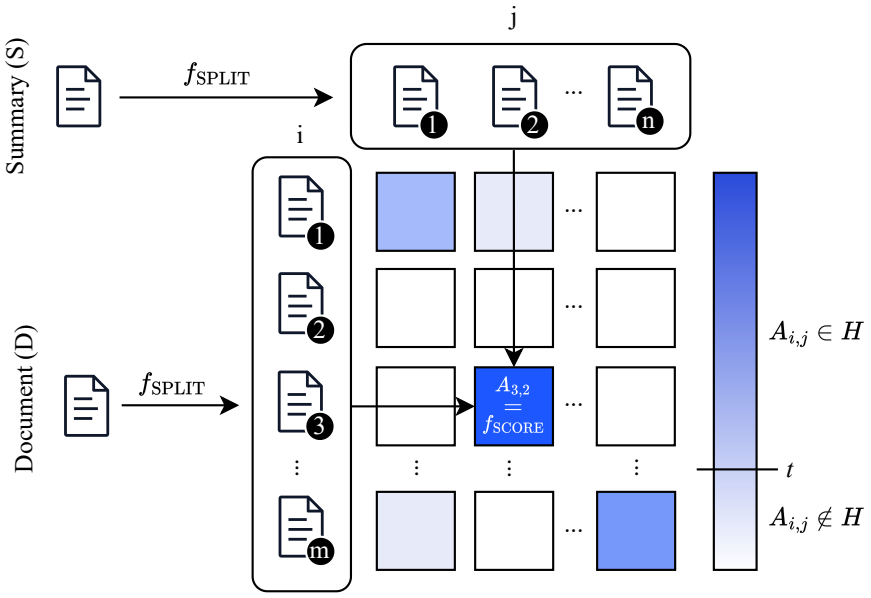


Figure 1: Procedure for calculating summary source alignments.

2 Formalising the Task

We define summary source alignment as a task consisting of two distinct functional stages: (1) unit segmentation and (2) correspondence scoring. The process is illustrated in Figure 1. To determine which parts of the source document D provide evidence for the information included in the summary S, the document is first split into a sequence of m units d using a segmentation function f_{SPLIT}:

$$\{d_1, d_2, \ldots, d_m\} = f_{\text{SPLIT}}(D)$$

These units are used as the basis for identifying the source of information. For every unit d_i it is checked whether any of its information is reflected in the summary. Since abstractive summaries can be lengthy and often fuse multiple information units into one text, comparing the summary in its entirety is prone to noise. Therefore the summary S is also partitioned into a sequence of n units s using f_{SPLIT}:

$$\{s_1, s_2, \ldots, s_n\} = f_{\text{SPLIT}}(S)$$

Every summary unit s_j is compared to every document unit d_i using a scoring function f_{SCORE}. This function is typically directional and asymmetric ($f_{\text{SCORE}}(d_i, s_j) \neq f_{\text{SCORE}}(s_j, d_i)$): it evaluates the degree to which a source unit d_i provides evidence for a summary unit s_j, mirroring tasks like NLI, where the source acts as the premise and the summary as the hypothesis. Comparing all units pairwise results in an alignment matrix $A \in \mathbb{R}^{m \times n}$, where $A_{i,j} = f_{\text{SCORE}}(d_i, s_j)$ represents the alignment between the i-th source unit and the j-th summary unit:

$$A_{i,j} = f_{\text{SCORE}}(d_i, s_j)$$

From this representation we can derive an alignment set H of all the correlated pairs as defined by Ernst et al. (2024) by applying a threshold t for pairs or a top-k value per summary unit to calculate binary labels from the continuous alignment values:

$$H = \{(d_i, s_j) \mid A_{i,j} \geq t \vee i \in \text{argmax}_k(A_{\cdot,j})\}$$

Within this framework, f_{SPLIT} and f_{SCORE} are viewed as independent modules, allowing for a variety of implementations with unique characteristics and trade-offs.

2.1 Segmentation

Segmentation varies primarily by granularity: text can be split by paragraph, sentence or even atomic, contextualized claims, a concept rooted in the Summary Content Units (SCUs) introduced for Pyramid evaluation by Nenkova and Passonneau (2004). To realise these splits, a variety of different means such as structural characteristics of the text (such as syntax, line breaks or captions) or specialised machine learning models may be used. For instance, to approximate SCUs for the Pyramid evaluation, Nawrath et al. (2024) propose relying either on

Abstract Meaning Representation (AMR) parsing or LLMs. The granularity of the source units defines how precisely sources of information may be identified, as the proposed system can only ever attribute information to a specific unit rather than a sub-portion of it. More granular splits therefore allow for easier validation. When splitting the texts into larger units consisting of several sentences and propositions, minimum or maximum limits for the length may be defined to control this granularity. These limits may be specified in different units such as characters, tokens or sentences.

Additionally, the resulting units may be contextualised (meaning additional context is added to the raw text). This can be especially important, if several similar topics are discussed within the source material. A statement about the publishing time of a book may be incorrectly attributed to another book, if the information what book is being discussed is missing from the unit. In order to avoid loss of context, units may also overlap or alternatively be entirely disjoint.

Finally, while our formalization implicitly assumes a uniform segmentation approach for both summary and source, asymmetric segmentation is also possible. We define this as using two distinct segmentation strategies for the summary and source texts, respectively. For example, the source may be split into higher level paragraphs while the summary is split into detailed propositions. Asymmetric splitting can be beneficial when applying scoring functions made for texts of different lengths or when dealing with concise summaries of long input texts.

2.2 Scoring

In order to score a pair of summary and source units, they may be compared using various types of measures. Following the definition of Jalili Sabet et al. (2020), the output of f_{SCORE} may be a continuous score or a label. One option for calculating a continuous score is to utilize traditional reference-based metrics used for evaluating the quality of generated texts by calculating text similarity. Lexical metrics such as ROUGE (Lin, 2004) or other statistical measures may however perform worse when evaluating abstractive texts generated by LLMs, since they are sensitive to paraphrasing (Zhang et al., 2020). To counteract this, semantic measures like the BERTScore (Zhang et al., 2020) or embedding similarities in general may be employed. Approaches using embeddings can be differentiated by the model and the specific distance or similarity measure used. Alternatively to text similarity measures, specialised classification models for summary source alignment can be utilized. While f_{SCORE} is defined as a function of two discrete units, the implementation of such models can be either context-independent (comparing units in isolation) or context-aware. In context-aware scoring, the function f_{SCORE} incorporates surrounding units of D or S to resolve co-references and provide context.

Moving beyond these dedicated models, the LLM-as-a-judge paradigm (Liu et al., 2023) can also be applied to evaluate the relationship between text units leveraging their understanding of natural language. For these models, prompt design is essential in order to receive accurate results. It should be noted that such instructions can also be applied when using specialized instruct-embeddings. These can take an additional instruction to frame the input influencing the resulting embedding (Su et al., 2023). Further, when scoring unit pairs using complex models such as LLMs the high computational cost must be considered. A granular segmentation may, depending on the length of the data, result in a large number of unit pairs, which may require significant processing times and can lead to high token usages and associated costs when using proprietary models. To avoid these costs, requests should be minimized if possible. This can for example be done by batching multiple comparisons in one request, which may also benefit results, as additional context is provided. Alternatively, a hybrid approach, where expensive models are only utilized on a pre-filtered subset of the unit pairs, may be adopted to mitigate potential expenses.

3 Related Work: Existing Approaches to the Task

Using the newly established taxonomy, existing approaches to summary source alignment can be systematically classified, as summarized in Table 1.

Identifying which parts of the source contributed to a generated summary has long been relevant in summarization research. When training extractive models, prior work approximated alignments by mapping abstractive summaries back to source sentences in order to derive gold-standard labels. Chen and Bansal (2018), Lebanoff et al. (2019), and Nallapati et al. (2017) employ heuristic procedures based on ROUGE similarities to select source

Approach	f_{SPLIT}		f_{SCORE}	Alignment Set (H)
	Source	Summary		
Nallapati et al., 2017	Sentence	Full Text	ROUGE (Heuristic)	Greedy cumulative max
Chen and Bansal, 2018	Sentence		ROUGE (Heuristic)	Top 1–2 most similar
Lebanoff et al., 2019	Sentence		ROUGE (Heuristic)	Top 1–2 (dynamic*)
Ernst et al., 2021	Claims		Supervised Classifier	Binary classification
Zha et al., 2023	350-token chunk	Sentence	Unified NLI Model	Binary classification
Gao et al., 2023	100-word chunk	Sentence	LLM	In-generation or Post-Hoc citations
Amplayo et al., 2023	Sentence		Soft-matching (N-gram / Model)	Maximum match score
Landes and Di Eugenio, 2024	Sentence (AMR)		Max-Flow Network	Bipartite graph

*Removes overlapping words dynamically during selection.

Table 1: Classification of existing automatic summary-source alignment approaches within the proposed taxonomy.

sentences. While these approaches do not explicitly formalize alignment as a stand-alone task, but rather use it as an auxiliary step for dataset construction, the methods can be categorized using the proposed taxonomy.

With the emergence of models that can generate cohesive abstractive summaries, the focus shifted from generating training data to evaluating the information used. This shift is reflected by more sophisticated approaches to solve the task, such as training or fine-tuning dedicated models such as SuperPAL (Ernst et al., 2021, 2024) for classification. Models like AlignScore (Zha et al., 2023) leverage cross-task pre-training, incorporating NLI and fact-verification data. Similarly, SMART (Amplayo et al., 2023) frames factual evaluation around sentence-level alignments. It employs string-based or model-based soft-matching to score pairs, mapping each generated summary sentence back to the highest-scoring source sentence to identify evidence. Within our proposed taxonomy, this serves as an example of evaluating varying implementations of f_{SCORE} (string-based vs model-based) while keeping the f_{SPLIT} constant at the sentence level. Going beyond purely text-based sequence modeling, approaches like CALAMR (Landes and Di Eugenio, 2024) represent both source and summary sentences as AMR graphs. Based on these graphs, CALAMR frames the task as an information flow problem, utilizing a max-flow algorithm across a bipartite graph to trace component overlap. Alternatively, frameworks exemplified by ALCE (Gao et al., 2023) frame alignment as an automated citation task, prompting LLMs to score and link generated summary sentences back to specific source chunks either directly during or after generation. While generating the citations directly during generation has been shown to lead to a higher coverage in comparison to post-hoc approaches as proposed here (Saxena et al., 2025), this however introduces a risk of a methodological circularity, as the validation process relies on the same internal capabilities that are the subject of the investigation.

More broadly, summary-source alignment is closely related to factual consistency verification of generated text. Whereas verification aims to determine if a claim is supported by the source at all (e.g., via question-answering (Kryscinski et al., 2020; Wang et al., 2020; Fabbri et al., 2022; Min et al., 2023)), alignment focuses specifically on tracing the exact origin of that information. Due to the different nature of the goal, these approaches are not represented by the proposed taxonomy.

4 Evaluating the Task

While existing approaches propose complete alignment pipelines or only evaluate different scoring functions (Amplayo et al., 2023; Ernst et al., 2021), previous work lacks a systematic evaluation of the

underlying segmentation and scoring functions as independent modules.

To address this, we evaluate combinations of representative f_{SPLIT} and f_{SCORE} functions separately. Rather than an exhaustive study, this serves as a empirical baseline evaluation demonstrating the necessity of considering these components individually. This setup allows us to qualitatively analyse the resulting alignment matrices and explicitly evaluate the trade-offs between computational efficiency and alignment precision.

4.1 Experimental Framework

To systematically compare these modular components, we first define the experimental framework, consisting of the dataset and evaluation metrics used.

4.1.1 Dataset

Since granular alignment is inherently subjective, we construct a curated, manually annotated dataset to serve as a qualitative gold standard for evaluation. It contains eight multi-document summaries with an average length of 650 tokens generated via various LLMs based on Wikipedia articles. The summaries span across two distinct topics: Five summaries are based on four dog breeds ($\sim$ 13k tokens) and three summaries are based on four Northern German cities ($\sim$ 39k tokens). Restricting the input to two general topics simplifies manual verification. Furthermore, using lengthy Wikipedia articles forces the models to synthesize diverse information into concise summaries. The long input thus serves as a representative use case that necessitates automatic alignment, since reading the entire source to verify information is time-consuming.

All summaries are manually annotated by two annotators using a graphical user interface which enables fine-grained annotations at the character-level. Given the long source documents, this precise mapping requires significant cognitive effort and time, making large-scale crowdsourcing unfeasible and necessitating a highly curated expert dataset. Because this data is used exclusively for evaluation rather than training, we prioritize this annotation depth over sheer volume. These detailed annotations allow for mapping to any level of granularity for f_{SPLIT}. In total, across both annotators and the eight summaries, 775 alignments are annotated. The number of resulting positive and negative samples depends on the f_{SPLIT} used. For example, when splitting summary and source by sentence, a robust evaluation set of 1058 correlated and $130\,566$ non-correlated sentence-pairs is derived. A more coarse f_{SPLIT} method will result in a smaller number of pairs.

Inter-annotator agreement (Cohen's Kappa) on the alignment matrix at sentence-granularity for summary and source is 64.5%, indicating fair agreement following Fleiss (1981). Manual verification revealed that the disagreements stem largely from an asymmetry in human annotation behaviour: while annotators precisely highlighted sub-sentences in the generated summaries, they frequently selected entire paragraphs in the source documents as the supporting context, if the entire paragraph discusses the relevant topic. Further, information sometimes naturally occurs redundantly across source documents, with annotators either marking all or the most relevant occurrence. Both of these issues are characteristic for summaries in this domain, since information is highly aggregated and often pulled from multiple different positions.

To analyse, how our findings generalize beyond a single domain, we evaluate against a secondary dataset. For this cause, we utilize the subset of the MultiNews (MN) dataset (Fabbri et al., 2022) that was annotated by Ernst et al. (2021), comprising nine summaries with approximately 2.3k input tokens and 300 token in the summary each. For these summaries Ernst et al. (2021) provide 362 span-level annotations (approx. 40 per summary).

4.1.2 Metrics

To evaluate different implementations and combinations for f_{SPLIT} and f_{SCORE}, the calculated alignments for each combination are compared to the gold standard. To achieve this, the detailed gold standard annotations are mapped to the units resulting from f_{SPLIT}. The corresponding alignment score is calculated as the average across all available annotators. A pair is considered aligned if at least one human annotator marked text for each unit in that pair. This ensures that valid alignments missed by a single annotator are still captured.

Automatic evaluation is performed in two dimensions: classification performance and efficiency. To evaluate the classification performance, the alignment set H is assessed using precision, recall, and the F_1 score. The goal is to determine whether the automated method identifies the same aligned pairs as a human evaluator. To calibrate a threshold t or a top-k value for deriving H from the alignment matrix A, the dataset is partitioned into a calibra-

tion and a test set using a 60/40 split. If the f_{SCORE} method applied does not require calibration, only the test set is used regardless to allow for a fair comparison of different methods.

In addition to these performance metrics, the average processing time per scored pair is also considered. This is essential for ensuring that a method can scale effectively when evaluating long summaries and large datasets.

To complement these automatic measures and understand common error patterns, we also conduct a manual qualitative evaluation of a subset of the results. This qualitative analysis is essential in the context of summary-source alignments, as such mappings are inherently subjective, particularly when information is repeated multiple times within the source text. Performing a manual evaluation thus avoids discounting an alignment system due to issues or inconsistencies in the human annotations.

4.2 Comparing Segmentation and Scoring Approaches

Using the defined experiment setup, we systematically compare combinations of different approaches for f_{SPLIT} and f_{SCORE}, by calculating annotations for all summaries in the datasets.

4.2.1 Defining the Search Space

As described in section 2, there are many possible implementations for the two sub-tasks of summary source alignment. All segmentation functions can further be combined with all scoring functions, making up a large search space for a efficient and scalable approach. For this reason, we evaluate a representative subset.

Segmentation Functions

Segmentation is performed on different levels of granularity: proposition, sentence, paragraphs and chunks of four sentences each. In order to split the source into propositions, we use Gemma 4 26B A4B to generate atomic claims from groups of sentences, similar to the LLM-generated semantic units (SGUs) proposed by Nawrath et al. (2024). Sentence splitting is implemented via spaCy.[2] Paragraphs and four-sentence-chunks are contextualized using the hierarchical captions, if available in the source, by utilizing Docling.[3] In addition to symmetric splitting for each granularity, asymmetric segmentation, where the source is split into chunks, while the summary is split into sentences, is evaluated.

Scoring Functions

For scoring, a mixture of simple lexical and more complex semantic approaches is employed. In order to allow for comparison with existing approaches, we evaluate ROUGE (Chen and Bansal, 2018), dynamic ROUGE while removing overlapping words during selection (Lebanoff et al., 2019) and SuperPAL (Ernst et al., 2021). To evaluate semantic similarity we calculate the BERTScore (Zhang et al., 2020) using the model microsoft/deberta-xlarge-mnli.

Additionally, we implement f_{SCORE} as the cosine similarity of embeddings using Qwen3 8B Embedding, while framing the summary unit as the "question" using an instruction. Since embedding models are mainly optimized for asymmetric inputs (short question, long document), we always split the summary into sentences and use the maximum similarity score for sentence to document unit as the score of the entire summary unit.

Finally, an LLM (we use Qwen3.5 9B) is utilized to judge whether information from a given source unit is included somewhere in the summary. We group all possible summary units per document unit in one request, to reduce the number of requests and speed up processing time. To improve scoring, the model is instructed to identify supported and unsupported information following a Chain-Of-Thought approach before providing the final score.

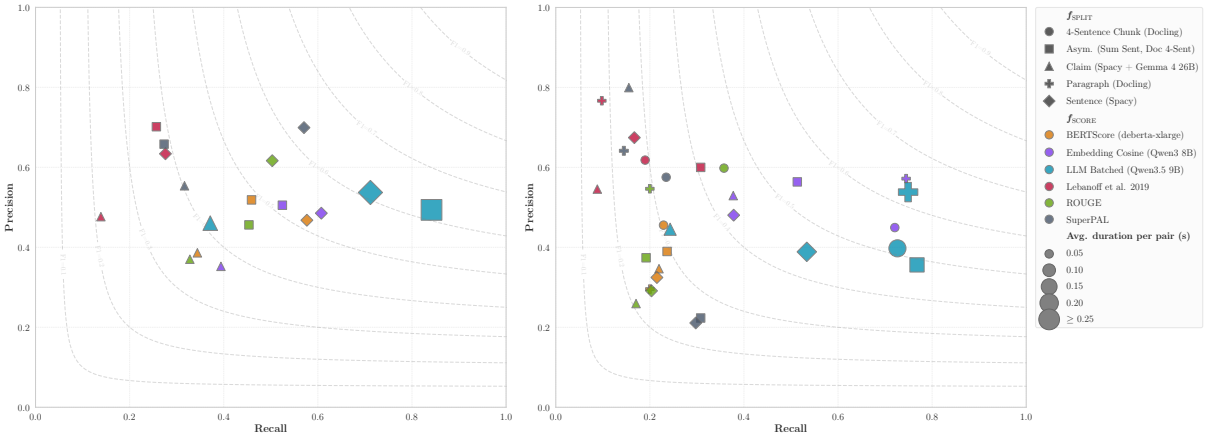
Calibration of the threshold t and top-k is performed for all f_{SCORE} implementations yielding a continuos score with no inherent meaning (ROUGE, BERTScore, and embeddings). For other methods, parameters are fixed: LLM-based scoring uses $t = 0$ by instruction, and SuperPAL uses $t = 0.5$ because it outputs alignment probabilities. Finally, dynamic ROUGE Lebanoff et al. (2019) requires no t or top-k, as it inherently limits summary units to two sources.

4.2.2 Exploring the Search Space

In total, by combining the five segmentation approaches and six different scoring implementations, 30 different systems are evaluated. In Figure 2 the trade-off between precision, recall and processing time is visualized. We find that depending on the f_{SPLIT} and f_{SCORE} function used, results

[2] https://github.com/explosion/spaCy

[3] https://github.com/docling-project/docling



(a) MultiNews (b) Wikipedia

Figure 2: Performance of different f_{SPLIT} and f_{SCORE} combinations.

vary greatly. This demonstrates the importance of analysing f_{SPLIT} and f_{SCORE} implementations and carefully selecting a fitting approach.

Since the MN dataset is rather short with only one paragraph per input document and in the summary, segmenting into large chunks would result in every pair being a valid annotation, therefore not having a sensible application. For this reason, segmentation by paragraphs and segmentation by four-sentence-chunks are excluded for this dataset. Splitting by sentence, however, yields notably better results for MN than for Wikipedia considering most f_{SCORE} functions. This is due to the different nature of the two datasets: While Wikipedia articles often include long paragraphs covering many details that are aggregated in the summary, the news articles of MN are already highly dense in information. This suggests the need for different f_{SPLIT} functions depending on the domain. When evaluating texts dense in information, f_{SPLIT} implementations that result in short chunks should be utilized.

When considering different implementations for f_{SCORE}, LLM-as-a-judge and embedding similarity perform consistently well in terms of their precision and recall regardless of the dataset or splitting method. For these scoring functions, splitting by sentences (for the Wikipedia dataset) and claims yields the worst results respectively. Splitting by paragraph on the other hand results in the best alignments for Wikipedia articles, while asymmetric splitting achieves good results overall, notably achieving the highest recall for both datasets when pairing with scoring using an LLM. Manual validation shows, that while embedding achieves slightly higher scores for most f_{SPLIT} implementations on the Wikipedia dataset, many of the calculated annotations are due to general topical overlap, rather than the document unit being the actual information source. Further, specifically when splitting by sentence, for both methods some errors could be observed due to the missing context: for example the models attributed some statements in the summary to an incorrect dog breed. In terms of the processing time per pair however, LLMs are considerably slower thus scaling worse to long inputs.

In terms of the other f_{SCORE} functions, ROUGE, BERTScore and SuperPAL perform better on MN, achieving a low recall for the summaries of Wikipedia articles. Good performance for SuperPAL using sentence segmentation on MN is expected, since this is highly similar to the MN propositions the model was trained on. Dynamic ROUGE scoring following Lebanoff et al. (2019) on the other hand achieves low recall independent of f_{SPLIT} or the dataset. In addition to the limitations of heuristic measures, this can be explained by filtering out word overlap, leaving only few possible matches per summary unit.

4.3 Designing a Hybrid System to Balance Trade-Offs

While scoring via LLM yields the best results overall, employing LLMs for summary source alignments is not scalable to long inputs and summaries due comparatively high processing times per pair. In order to reduce processing time while retaining good performance, we propose a hybrid system combining the LLM with an embedding-based prefilter: First, all pairs are scored using the embedding. Only pairs above a certain threshold are then

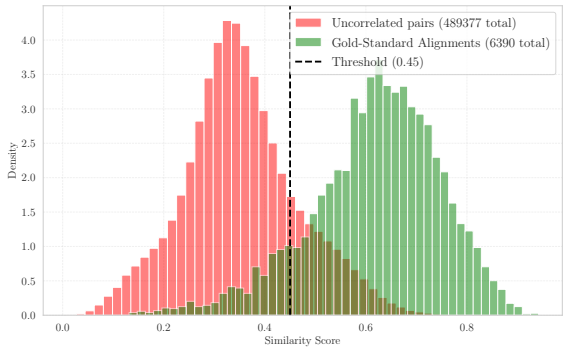


Figure 3: Total score distribution for f_{SCORE} = Qwen3 8B Embedding.

classified by the LLM to filter out matches due to general topical overlap.

Since the recall for any two-step system is limited by the recall of the pre-filter, when picking a threshold the recall must be prioritized higher than when calibrating for usage as a primary scoring function. For this reason, we consider the score distribution of the embedding-similarity calculated on all pairs of all splitting functions as seen in Figure 3 and pick a threshold so that most correct pairs are still considered while filtering out as many unrelated pairs as possible.

To evaluate the benefits of this hybrid approach, its performance is compared to the LLM and the embedding-based approaches on both datasets. For this, the asymmetric splitting is utilized, since this method showed consistent results on both datasets when scored using an LLM. As seen in Table 2, the processing time is reduced by more than half in comparison to using the LLM on its own. It is notable that in addition to the processing time decreasing, the precision increased by approximately 0.1. This can be explained by the fact that per document unit, there are fewer candidate summary units which could distract the LLM. If f_{SCORE} were implemented context-unaware by scoring every pair individually, the precision could not increase. The increase in precision, however, comes at the expense of a slight decrease in recall. This drop is expected, as any aligned pairs mistakenly discarded by the pre-filtering stage are permanently excluded from the LLM's evaluation. By the means of our hybrid system, we therefore achieve an improved F1-score at less than half the cost.

4.4 Verifying against Naive Baselines

To verify the performance of the proposed hybrid system and investigate potential positional biases, we evaluate the performance of two naive baselines on our manually annotated Wikipedia dataset. First, we introduce a lead baseline to quantify whether our annotators disproportionately rely on the beginning of a source document. Since the dataset consists of multi-document summaries, rather than just aligning the very first source unit, this baseline aligns the first source unit of every input document to the entire summary. Second, a random baseline is employed, which assigns alignments uniformly at random to establish an absolute lower bound for performance.

When aggregated across all f_{SPLIT} implementations, both baselines yield low overall performance. The lead-per-document baseline achieves an F_1 score of only 0.057, consisting of a precision of 0.092 and a recall of 0.050. This very low recall demonstrates that the annotators did not predominantly rely on the lead segments of the source documents. The random baseline performs slightly worse, with an F_1 score of 0.040. While it achieves a relatively high recall of 0.489, thus identifying roughly half of the actual annotations by chance, it suffers from extremely low precision at 0.022. This can be explained by long input texts, where random assignment generates a vast number of unrelated alignment pairs. It has to be noted however, that the lead baseline achieves a better precision than random alignment. A potential reason for this, is the inherent structure of Wikipedia articles, which usually start with a short summary of the topic.

Overall the verification against the baselines shows, that our manually annotated dataset has no clear positional biases. Furthermore, the results show that all evaluated f_{SCORE} functions, including the worst performing ROUGE based methods, substantially outperform the naive lower bounds. This suggests that while lexical overlap already provides a much stronger performance than structural position, the alignment task remains complex. Our hybrid system manages to address this underlying complexity yielding the strongest overall performance.

5 Conclusion

By decoupling summary source alignment into unit segmentation and correspondence scoring and systematically evaluating combinations of these subfunctions, we demonstrated that while LLMs provide the semantic precision needed to overcome limitations of traditional lexical heuristics, their

f_{SCORE}	Parameters	Avg. duration per summary	F_1	P	R
Qwen3 8B Embedding	$t_{\text{calibrated}} = 0.6, k_{\text{calibrated}} = 5$	3.69s	0.495	0.584	0.429
Qwen3 8B Embedding	$t_{\text{pre\_filter}} = 0.45$	3.69s	0.211	0.120	0.902
Qwen3.5 9B	$t_{\text{fixed}} = 0.0$	399.17s	0.525	0.393	0.791
Hybrid	$t_{\text{fixed}} = 0.0$	117.13s	0.602	0.502	0.752

Table 2: Performance of the proposed hybrid system with f_{SPLIT} = Asymmetric.

computational costs restrict their application at scale. To address this shortcoming, we proposed a hybrid system utilizing an embedding-based prefilter before applying an LLM to calculate the final score to balance semantic precision with processing efficiency.

Additionally, we demonstrated that the choice of granularity for an appropriate splitting function can depend on the density of the text, thus being domain-dependent. Source units that align with an exceedingly high number of summary units are likely too broad and lack necessary specificity. Building on this insight, document segmentation could be formulated as an optimisation problem designed to minimize the number of summary alignments per source unit. By dynamically penalizing broad alignments, future systems could ensure that alignments remain highly precise and do not contain unnecessary context.

Our modular framework provides a scalable foundation for evaluating information origins of LLM outputs after generation. Future work should leverage these empirical insights to refine evaluations and formally expand the search space to include larger datasets, additional models for scoring and a broader pool of human annotators.

Limitations

While the proposed framework offers a systematic approach to evaluating summary-source alignment, this study has several limitations. Mainly, while our manual evaluation relies on a dense, highly curated gold-standard benchmark, it is restricted to two specific domains, input lengths of up to 13k tokens and 14 summaries. However, these summaries capture a broad variance: different LLMs, prompting strategies and human written texts. Given this density and summary diversity, we anticipate that merely scaling the dataset's size within these domains would likely yield diminishing returns and not fundamentally alter our primary findings. Further, the empirical evaluation covers a representative, but non-exhaustive subset of segmentation and scoring functions, and we only consider open-source models for proposition extraction and scoring. Therefore, the performance of the proposed hybrid system may vary when applied to different models, highly specialized domains, or significantly longer inputs, necessitating further validation on broader, multi-domain datasets.

Finally, any post-hoc approach, as proposed here, evaluates the final output rather than the model's internal process. Thus, even when finding sources for all information in the summary, we can not definitively prove, that the text was actually based on the source text instead of the world knowledge of the model. Ultimately, however, the distinction between attribution and proving real usage is largely theoretical. As long as the generated claims can be reliably verified against the source document, the primary objective of information tracing is successfully achieved.

References

Reinald Kim Amplayo, Peter J Liu, Yao Zhao, and Shashi Narayan. 2023. SMART: Sentences as basic units for text evaluation. In *The Eleventh International Conference on Learning Representations*.

Yen-Chun Chen and Mohit Bansal. 2018. Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Anshuman Chhabra, Hadi Askari, and Prasant Mohapatra. 2024. Revisiting Zero-Shot Abstractive Summarization in the Era of Large Language Models from the Perspective of Position Bias. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 1–11, Mexico City, Mexico. Association for Computational Linguistics.

Ori Ernst, Ori Shapira, Ramakanth Pasunuru, Michael Lepioshkin, Jacob Goldberger, Mohit Bansal, and Ido Dagan. 2021. Summary-Source Proposition-level Alignment: Task, Datasets and Supervised Baseline. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 310–322, Online. Association for Computational Linguistics.

Ori Ernst, Ori Shapira, Aviv Slobodkin, Sharon Adar, Mohit Bansal, Jacob Goldberger, Ran Levy, and Ido Dagan. 2024. The Power of Summary-Source Alignments. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6527–6548, Bangkok, Thailand. Association for Computational Linguistics.

Alexander Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2022. QAFactEval: Improved QA-based factual consistency evaluation for summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2587–2601, Seattle, United States. Association for Computational Linguistics.

Joseph L. Fleiss. 1981. *Statistical methods for rates and proportions*, 2nd edition. John Wiley, New York.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488, Singapore. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High Quality Word Alignments Without Parallel Training Data Using Static and Contextualized Embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the Factual Consistency of Abstractive Text Summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Paul Landes and Barbara Di Eugenio. 2024. CALAMR: Component ALignment for Abstract Meaning Representation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2622–2637, Torino, Italia. ELRA and ICCL.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. Scoring Sentence Singletons and Pairs for Abstractive Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2175–2189, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 3075–3081, San Francisco, California, USA. AAAI Press.

Marcel Nawrath, Agnieszka Nowak, Tristan Ratz, Danilo C. Walenta, Juri Opitz, Leonardo F. R. Ribeiro, João Sedoc, Daniel Deutsch, Simon Mille, Yixin Liu, Lining Zhang, Sebastian Gehrmann, Saad Mahamood, Miruna Clinciu, Khyathi Chandu, and Yufang Hou. 2024. On the role of summary content units in text summarization evaluation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 272–281, Mexico City, Mexico. Association for Computational Linguistics.

Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 145–152, Boston, Massachusetts, USA. Association for Computational Linguistics.

Mathieu Ravaut, Aixin Sun, Nancy Chen, and Shafiq Joty. 2024. On context utilization in summarization with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2764–2781, Bangkok, Thailand. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA. Association for Computing Machinery.

Yash Saxena, Raviteja Bommireddy, Ankur Padia, and Manas Gaur. 2025. Generation-time vs. post-hoc citation: A holistic evaluation of LLM attribution. In *NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling*.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. One Embedder, Any Task: Instruction-Finetuned Text Embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121, Toronto, Canada. Association for Computational Linguistics.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and Answering Questions to Evaluate the Factual Consistency of Summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. AlignScore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348, Toronto, Canada. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*.

A Appendix

The appendix offers supplementary material.

A.1 Dataset Generation

As part of our research, we publish a manually annotated dataset of LLM-generated multi-document summaries. In the following, we describe how this dataset was created.

A.1.1 Source Data

As a data basis for the multi-document summaries, Wikipedia articles were used. We selected two topics (dog breeds and cities) and four Wikipedia articles per topic to serve as the source texts for the summaries. For the summaries based on dog breeds, the input consists of *Cockapoo*, *Dachshund*, *Golden Retriever*, and *Pug*. The summarized cities include *Bremen*, *Hamburg*, *Hanover*, and *Kiel*.

A.1.2 Summary Generation

Summaries based on the previously described inputs were generated using multiple LLMs. These models were accessed through various channels: the OpenRouter API[4], official APIs, or official chat interfaces. To increase output variance, several summaries were processed using a hierarchical-merge architecture instead of a standard single-pass call. An overview of the included summaries is provided in Table 3.

A.1.3 Manual Annotation

In order to manually annotate the dataset for evaluation and verify the results of automatic methods, we developed a custom interface to edit and view annotations, displayed in Figure 4. When viewing alignments calculated by an automatic system, as shown in Figure 4b, the color indicates the value resulting from f_{SCORE}. Furthermore, all sources per summary unit are grouped together.

A.2 Reproducibility Setup

To enable full reproducibility, we outline our serving infrastructure, hyperparameters, and the exact prompt templates utilized for segmentation and scoring. The complete source code, dataset and full prompt templates (including few-shot examples) are publicly available in our repository[5].

A.2.1 Hardware

All models used for scoring were executed using a RTX 4090 GPU with 24 GB VRAM.

A.2.2 Model Serving

For local model serving of LLMs and embeddings, Llama.cpp[6] provisioned through llama-swap[7] was utilized. During inference, we enabled Flash Attention to optimize processing speed. We further explicitly disabled context shift to prevent silent prompt truncation when exceeding the maximum context window. Finally, to fit within our 24 GB VRAM constraints, we used quantized GGUF formats provided by Unsloth on Hugging Face[8]: `Q4_K_XL` for Gemma 4 26B, `Q8_K_XL` for Qwen3.5 9B, and `Q8_0` for Qwen3 8B Embedding.

A.2.3 Inference Hyperparameters

In order to speed up processing time we disabled the native reasoning token generation for all LLMs used. For further hyperparameters, we followed the optimal settings for the models employed as recommended by Unsloth for non-thinking general tasks. Specifically, for Gemma 4 26B A4B, we set the temperature to 1.0, top-p to 0.95, and top-k to 64. For Qwen3.5 9B, we utilized a temperature of 0.7, top-p of 0.80, top-k of 20, and min-p of 0.0.

A.2.4 Prompts

The exact system prompts utilized for proposition extraction (Prompt 1) and scoring unit-pairs using an Instruct-embedding (Prompt 2) as well as an LLM (Prompt 3) are detailed below. It should be noted that in Prompt 3 the few-shot examples have been omitted for brevity.

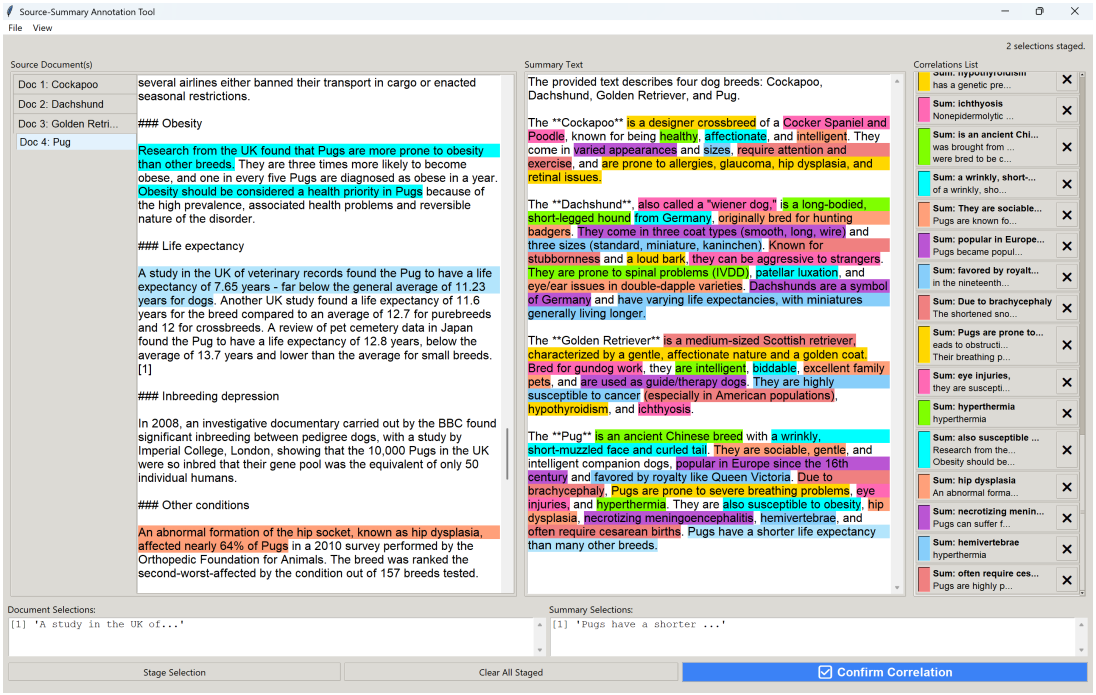
[4] https://openrouter.ai/

[5] https://github.com/apfelphine/decoupling-summary-source-alignments

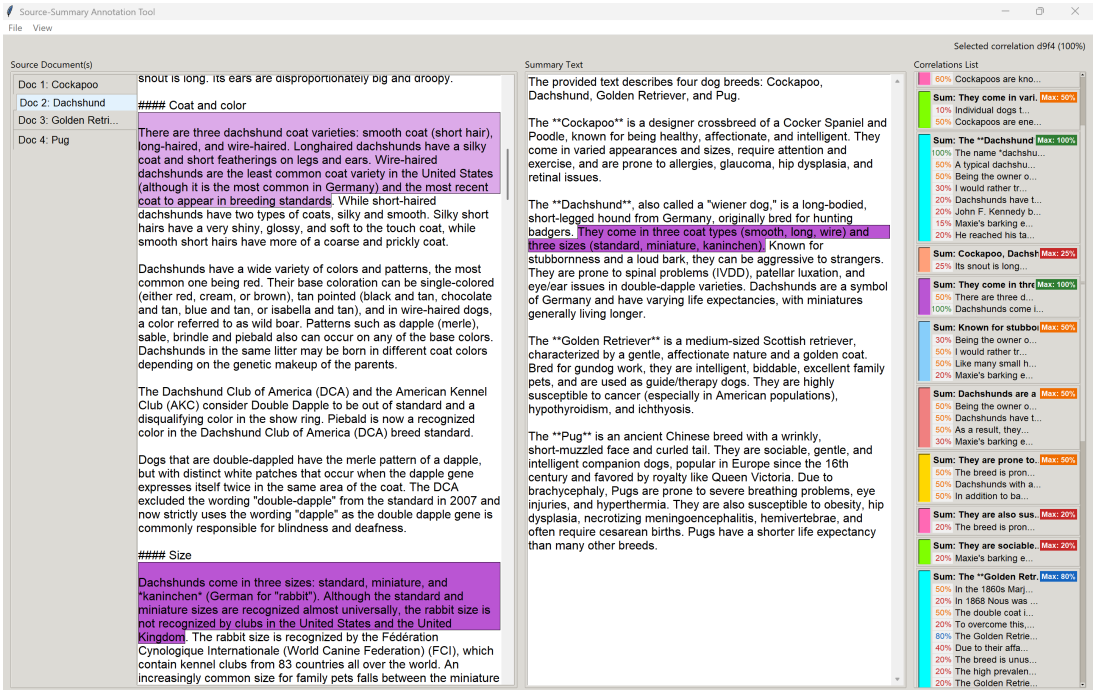
[6] https://github.com/ggml-org/llama.cpp

[7] https://github.com/mostlygeek/llama-swap

[8] https://huggingface.co/unsloth



(a) Editing alignments



(b) Viewing alignments

Figure 4: Annotation interface.

Topic	ID	Architecture	Model	Access Method
Dog Breeds	0	Hierarchical Merge	Gemini 2.5 Flash	Official API
Dog Breeds	1	Hierarchical Merge	GPT OSS 20B	OpenRouter
Dog Breeds	2	Single Pass	Gemini 2.5 Flash	Official API
Dog Breeds	3	Single Pass	GPT OSS 20B	OpenRouter
Dog Breeds	4	Single Pass	Llama 70B	OpenRouter
Cities	0	Single Pass	Gemini 3.1 Pro	Chat Interface
Cities	1	Single Pass	Gemini 3.1 Flash-Lite	Chat Interface
Cities	2	Single Pass	GPT OSS 20B	OpenRouter

Table 3: Overview of generated summaries based on Wikipedia inputs.

```
**TASK:**
Your task is to extract all factual, verifiable claims from a given text.
A 'claim' is a single, unambiguous statement of fact.

Only extract claims from the **Text to Analyse**. If available, you are given
    additional information as context.

**CLAIM EXTRACTION STEPS:**
When extracting claims, always follow the following steps:
1. **Identify:** Read the text and identify all sentences or phrases that contain
    factual information.
2. **Contextualize:** Rewrite each claim so it is fully understandable on its own,
    resolving any pronouns (like 'it', 'they', 'that') by replacing them with the
    specific nouns they refer to.
3. **Decompose:** If a single chunk contains multiple distinct claims, split it into
     separate, simple claims.
4. **Attribute:** For each claim, provide the verbatim text part that supports it.

**ESSENTIAL ATTRIBUTION RULES:**
When attributing the source of the claim, imaging using a highlighter.
You can use the highlighter once to highlight one part of the text (for example a
    word, a sub-sentence, etc.).
You cannot lift the highlighter.
You must only highlight the part of the chunk including the **essential information
    ** used to make this claim without unnecessary context.
Return the highlighted information verbatim, including any markdown formatting,
    whitespaces, quotation marks etc. present in the original text.

**OUTPUT FORMAT:**
Strictly respond in the following json format:
```json
{
    "extracted_claims": [
        {
            "claim": String, # The extracted claim
            "verbatim_source": String, # The **verbatim** source of the claim
        }
    ]
}
```
```

Prompt 1: Template for proposition extraction.

```
Instruct: Retrieve the source text that supports or implies this statement
Query: {{ summary_chunk }}
```

Prompt 2: Embedding instruction for summary units.

```
<identity>
    You are an expert at verifying factual consistencies between a document and a
        summary created based on it.
    Your goal is to identify, if information from a given part of the source
        document was used in the summary and in what part of the summary.
    You insist on information in summaries being present in the original text to
        classify as factual support. You are pedantic about inconsistencies.
</identity>
<task>
    <input_format>
        You will be given:
        - A summary, split into several parts ("<summary_chunks>"). Every summary
            chunk is specified as ("<summary_chunk id=idx>Text</<summary_chunk>").
        - A part from the original document ("<source_chunk>") the summary was
            created based on.

        The chunks are formatted as markdown. Headers ("#") at the beginning of each
             chunk signify the section(s) of the chunk in the original document.
            They may serve as context, however they do NOT count as support.
    </input_format>
    <instruction>
        Judge how much each summary chunk is factually **supported by** the
            information in the source chunk.
        For each of the summary chunks, identify supported and unsupported
            information, then score what percentage of its information is **
            supported by** the given source chunk on a scale of 0-100.
    </instruction>
    <scoring_guidelines>
        **Base the score only on the alignment of specific facts and claims. A
            shared general topic is not sufficient for support. If summary chunk or
            source chunk do not contain any factual information, the score should be
             0.**
        A score of 100 means every claim in the summary chunk is supported by the
            source. A score of 0 means no information in the summary chunk is
            supported by the source.
    </scoring_guidelines>
    <output_format>
        Strictly respond in the following json format:
        ```json
        {
            "results":
                [
                    "chunk_idx": Integer, // The ID provided in the input tag. There
                         must be exactly one entry per summary chunk.
                    "supported_information": String, // Brief description of
                        information in this **summary chunk** that is also **
                        included in the source chunk**
                    "unsupported_information": String, //  Brief description of
                        information in this **summary chunk** that is **NOT included
                         in the source chunk**
                    "score": Integer // Final score from 0 to 100 of this summary
                        chunk
                ]
        }
        ```
    </output_format>
</task>
<examples>
    ...
</examples>
```

Prompt 3: Template for scoring all summary units for a document unit (excluding few shot examples).