Give Me The One-liner: Extracting Short Patient Summaries from Radiation Oncology Notes through Intermediates

Thibault Heintz, MD, MSc* TGHEINTZ@BWH.HARVARD.EDU
Mass General Brigham, Dana-Farber Cancer Center, Harvard Medical School, USA, Maastricht University, The Netherlands

Suraj Pai, MSc, PhD* BSPAI@BWH.HARVARD.EDU
Mass General Brigham, Dana-Farber Cancer Center, Harvard Medical School, USA, Maastricht University, The Netherlands

Marion Tonneau, MD MTONNEAU@BWH.HARVARD.EDU
Mass General Brigham, Dana-Farber Cancer Center, Harvard Medical School, USA

Cosmin Ciausu MSc CCIAUSU@BWH.HARVARD.EDU
Mass General Brigham, Dana-Farber Cancer Center, Harvard Medical School, USA, Maastricht University, The Netherlands

Shan Chen MSc, PhD SCHEN73@BWH.HARVARD.EDU
Mass General Brigham, Dana-Farber Cancer Center, Harvard Medical School, USA, Maastricht University, The Netherlands

Danielle Bitterman, MD, Prof. DBITTERMAN@BWH.HARVARD.EDU
Mass General Brigham, Dana-Farber Cancer Center, Harvard Medical School, USA

Hugo Aerts, MSc., Prof., HAERTS@BWH.HARVARD.EDU
Mass General Brigham, Dana-Farber Cancer Center, Harvard Medical School, USA

Raymond Mak, MD, Prof., RMAK@MGB.ORG
Mass General Brigham, Dana-Farber Cancer Center, Harvard Medical School, USA

Abstract

A patient one-liner is a very concise summary used in Radiation Oncology to streamline communication. In this study, we assess the ability of LLMs to provide apt one-liners through summarization of long-form consultation, imaging and pathology notes written by physicians for 101 patients encountered in Radiation Oncology practice. LLMs are known to struggle with long context lengths for summarization, often providing irrelevant output that does not align with the summarization intent. To tackle this, we extract one-liners via means of a two-step pipeline with an intermediate summary. We compare different methods of intermediate summarization, namely, 1) bulk summarization through structured fields (Generate once), 2) incremental summaries through structured fields and add/update operations (Chain of Key/CoK), and 3) bulk summarization through automatically optimized prompt (DSPy) using 2 open-source (deepseek-r1-8b, gemma3-27b) and one closed-source LLM (o3-mini). The intermediate summaries were passed to another automatic prompt optimization program to produce the final patient one-liner. Aggregating our observations across LLMs, we observe that CoK significantly outperformed Generate Once, demonstrating incremental summarization is more effective compared to bulk summarization when looking at structured intermediates. Secondly, Automatic prompt optimization, via DSPy, without structured fields or incremental operation outperforms Generate Once. Lastly, no significant differences were found between DSPy and CoK. Our blinded user-study found LLM generated one-liners more complete and preferred by the Radiation Oncologist compared to the human baseline. But, it was also found that they tended to produce more non-important information. Overall, our work shows the potential of automatic prompt optimization as well as

* These authors contributed equally

structured incremental summarization to provide one-liner patient summaries that may find routine application in radiation oncology and highlights future work focused on end-to-end optimization of structured intermediates.

Keywords: Clinical Note Summarization, LLM, JSON, DSPy

Data and Code Availability Data was collected under IRB 11286 at Brigham and Women's Hospital, Boston, USA. The data is not publicly available. Code is available at github.com/AgentRadOnc/give_me_the_oneliner.

Institutional Review Board (IRB) IRB approval, IRB number 11286 at Brigham and Women's Hospital, was obtained for this study.

1. Introduction

Medical Doctors refer to the "one-liner" as a very brief summary of a patient containing only the most essential information needed to start a discussion on the patient plan. In Radiation Oncology, it is used as a way to communicate the various facets of the Radiation Oncology planning process. To the doctor unfamiliar with the patient, writing the one-liner requires extensive chart-review. Automatically generated one-liners, confirmed by a clinician, could serve as a fast introduction to the patient for a first visit, help steer radiation therapy planning for a radiation oncologist, or even an AI algorithm. (Oh et al., 2024; Rajendran et al.).

Several studies have shown the potential of Large Language Models (LLMs) to extract and summarize clinical information from medical records as well as medical literature(Agrawal et al., 2022; Tang et al., 2023). Interestingly, Van Veen et al. found that physicians often prefer LLM produced summaries over human produced summaries. Van Veen et al. consider the medical summarization problem as summarizing a single note. In clinical reality, however, a doctor may want to summarize over many different notes, testing the limits of model context lengths. Therefore, direct prompting for a one-liner, one prompt from notes to summary, is either infeasible or prohibitively expensive. Furthermore, LLM performance is known to degrade for in context learning with longer context lengths. (Hwang et al., 2024)

To address this issue, Hwang et al. (2024) introduce the use of structured intermediates to keep a running summary for incremental summarization. They demonstrate that using JSON as the running summary for iterative document summarization outperforms using unstructured text. Furthermore, they develop Chain of Key, which we describe in Section 2. Another field of research is using automatic prompt optimization, via DSPy (Khattab et al., 2024, 2022), where optimizers operate on declarative input and output, few-shot examples, etc. are used to select optimal prompts.

In our work, we evaluate these paradigm in a clinical setting, build a two-step pipeline that generates a detailed intermediate summary, which is subsequently used to generate a 'one-liner' for a patient. We provide the following contributions:

1) We investigate a diverse set of choices in long-form clinical summarization, namely, using *structured vs unstructured* intermediates, *incremental vs bulk* summarization, smaller *open-source vs proprietary* models.

2) We develop a framework for multi-step one-liner generation, first, generating a longer detailed summary and then distilling information into a one-liner through self-configuring prompts (Khattab et al., 2024, 2022).

3) We conduct a user-study with a radiation oncologist to determine the clinical acceptability of the LLM generated one-liners.

2. Methods

2.1. Dataset

We leverage data from 101 patients treated for thoracic cancers at the department of Radiation Oncology at Brigham and Women's Hospital between 2001-2022, consisting of 53.5% females and 46.5% males, with an mean age of 67.5 years (SD: 10.5). The data contains consultation, imaging and pathology notes written by physicians, along with a one-line summary created during radiation therapy planning. We selected notes written 6 months preceding the start of radiation therapy, kept only the 10 latest consultation notes, 5 latest imaging and 5 latest pathology notes. Notes that originate from departments unrelated to radiation oncology were filtered out. Cases that did not contain basic patient demographics and tumor staging were removed. For downstream optimization of one-liner generation, explained in Section 2.4, we split this data in a train and test set, respectively 41 and 60 samples.

2.2. Generation of Structured Intermediates

Following the work of Hwang et al. (2024), we implement our incremental summarization through a running structured summary (JSON), that is iteratively updated/added to by each new input note (Figure 1). This Chain of Key (CoK) method uses 2 LLM calls for each note: 1) To generate the note's summary JSON; 2) To modify the running summary by providing the current running summary and new note's summary along with JSON keys to add and update. For comparison, we also implement Generate Once, which aggregates all notes in bulk and produces the structured summary in a single LLM call. For the generation of these summaries, three models are compared: o3-mini, Gemma 3 27B and DeepSeek R1-8B.

2.3. Automatic Prompt Optimization for Intermediate Summary

We additionally developed a summary generation program based on a self optimized prompt built with DSPy, that takes the concatenation of all notes and outputs a plain text summary as determined by the optimizer. The optimized prompt was built on a notes selected from a subset of 10 patients, using the MIPROv2 optimizer (Opsahl-Ong et al., 2024). o3-mini was used as the summary generating LLM and gpt-4o as the evaluating LLM with the same LLM-as-a-judge metric as defined in Section 2.5. Appendix Figure 6 shows the optimized prompt.

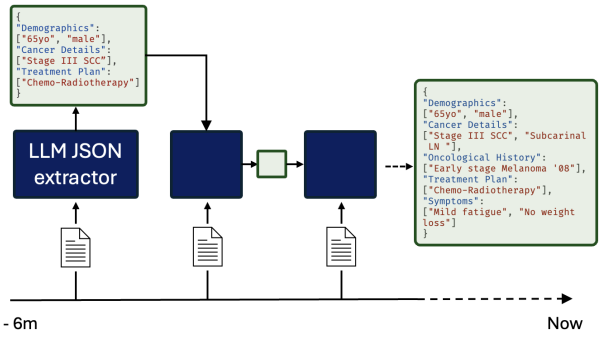

Figure 1: Iterative summarization workflow used by CoK. Based on Hwang et al. (2024).

2.4. One-liner Generation

The structured or unstructured summary is converted to a one-liner using a module implemented in DSPy, that automatically optimizes the prompt using their Bootstrap with Random Search optimizer on the training set described in Section 2.1. (Opsahl-Ong et al., 2024)

2.5. Evaluation

Three commonly used evaluation metrics for summaries, were used to compare the predicted one-liner with the corresponding ground truth, namely, ROUGE-L F1 score (Lin, 2004) and BERTScore (Zhang* et al., 2020). We also construct our own LLM-as-a-Judge using gpt-4o, grading the one-liner on 7 questions with a 'Yes / No / Not mentioned in reference' answer. The score is calculated as the fraction of "Yes" answers, while predictions attributed to the latter choice are ignored. The prompt is provided in Appendix Figure 7. The Wilcoxon Signed Rank test is used to test for significance when comparing methods and models.

2.6. Radiation Oncologist Validation Study

A blinded validation study was set up to evaluate the generated one-liners where a radiation oncologist was presented with the patient notes and the 2 one-liners: the ground truth and a predicted one, originating from either CoK and DSPy methods (27 samples each). The radiation oncologist was asked 4 questions on a 5 point likert scale: 'Which summary more completely captures important information?', 'Which summary includes less false information?', 'Which summary contains less non-important information?', 'Which summary would you prefer given the patient presentation?'. The first 3 questions were directly adapted from Van Veen et al.. Correlation with our quantitative metrics was computed using spearman correlation.

Lastly to evaluate our LLM-as-a-Judge metric, our annotator answered the same questions as those embedded in its prompt. 60 samples from the o3-mini with CoK experiment were compared using spearman rank correlation.

3. Results

3.1. Models and Techniques

We present the performance on ROUGE-L, BERT-Score and LLM-Score across methods and LLM used in Table 1. Comparing different summarization techniques, we notice DSPy and Chain of Key boast the highest scores for all metrics. Comparing Generate Once with Chain-of Key, we observe that Chain of Key performs significantly better ($p<0.05/3$, Bonferroni correction) for all language models on all metrics except for the o3-mini language model. Comparing Generate Once with DSPy, we observe outperformance of DSPy, statistically significant ($p<0.05/3$, Bonferroni correction) for all of the metrics across all language models combined, for DeepSeek R1 8B and o3-mini on the ROUGE-L and BERTScore metrics. Comparing Chain of Key with DSPy, we only notice a significant improvement of CoK over DSPy for the Gemma 3 model when evaluated on ROUGE-L or BERTScore.

Table 4 shows the comparisons between models. O3-mini based summaries outperform Deepseek based summaries significantly ($p<0.05$) for all methods combined, however, no statistical significance was reached for the subgroup Chain of Key on the ROUGE-L and BERTScore metrics and for DSPy on the LLM-Judge metric. O3-mini significantly outperforms Gemma 3 on all metrics when combining the 3 technique groups. This advantage can also be observed for the DSPy subgroup, but not for the CoK group or Generate Once for ROUGE-L and BERTScore subgroups. No significant differences were found between DeepSeek R1 8B and Gemma 3 27B. Two samples failed to produce a one-liner for the Gemma 3 model due to out of context errors in one-liner generation. Only the matching 58 samples were used to make comparisons with Gemma 3.

3.2. User Study

The results of the user study are displayed in Figure 2. We observe that for both of the best performing techniques, the LLM based one-liner performs better on the completeness question. No significant differences are observed for the question on false information. The ground truth is significantly preferred on the non-important information question, except for the DSPy subgroup. General preference is given to the AI based one-liners. This statistic is significant, except for the samples produced by CoK. There is no significant difference between AI and human for question 2, evaluating false positive clinical facts. The results of correlation analysis can be found in Appendix Table 2. No significant correlations were found between our metrics and the Doctor's assessment, except for the BERT-Score, which was negatively correlated to the Doctor for the question 'Which summary includes less false information?'.

Our correlation analysis comparing the LLM-as-a-Judge questions answered by a human and an LLM showed a spearman rank correlation of 0.1986, which was not significant with a p-value of 0.1282.

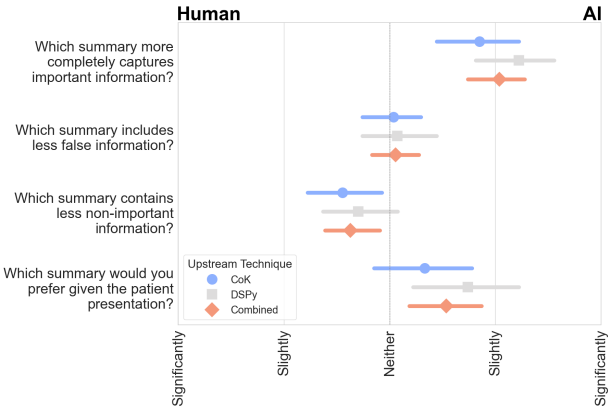

Figure 2: A pointplot comparing Generated one-liners with Ground truth, for CoK and DSPy, displaying the mean and 95% CI.

4. Discussion

In this work, we explored the use of intermediate representations for the generation of a patient one-liner. As expected, proprietary LLMs outperformed smaller language models, as scale offers several benefits, including better incorporation of large context lengths. We observe that, for structured intermediates, using incremental summarization improves results when compared to bulk summarization. We attribute this to the fact that in structured intermediates, having incremental summaries via CoK, removes redundancy through providing reasonable updates. Surprisingly, when bulk summarization is performed using an optimized prompt (DSPy), written by an LLM based feedback loop, similar results are achieved. This is especially interesting given the simplicity of the

Table 1: Comparison of all methods across LLM Models on ROUGE-L, BERTScore, and LLM-Judge score. Bolded corresponds to the best performing Method for a model for a metric. For statistical testing, we refer to Table 3 and Table 4

Method	Metric	DeepSeek R1-8b	Gemma3-27b	GPTo3-mini
Generate once	ROUGE-L	0.1332 ± 0.0594	0.1466 ± 0.0754	0.1746 ± 0.0853
	BERTScore	0.5844 ± 0.0530	0.5950 ± 0.0510	0.6115 ± 0.0529
	LLM-Judge	0.5354 ± 0.3380	0.6112 ± 0.2859	0.7542 ± 0.2732
Chain of Key	ROUGE-L	0.1710 ± 0.0830	0.1859 ± 0.0753	0.1844 ± 0.0783
	BERTScore	0.6096 ± 0.0560	0.6182 ± 0.0546	0.6204 ± 0.0571
	LLM-Judge	0.6725 ± 0.2186	0.7178 ± 0.2567	0.8044 ± 0.2276
DSPy	ROUGE-L	0.1591 ± 0.0758	0.1622 ± 0.0747	0.1997 ± 0.0728
	BERTScore	0.5970 ± 0.0603	0.5996 ± 0.0582	0.6235 ± 0.0502
	LLM-Judge	0.6660 ± 0.3043	0.6643 ± 0.2885	0.7887 ± 0.1950

DSPy setup requiring us to only define a task and an evaluating metric or language model. Here,we suggest future work implements automated end-to-end prompt optimization from DSPy in a method producing structured intermediates.

Interestingly, our results show that CoK improves the results for the Gemma 3 model up to parity on all metrics with the closed source o3-mini model. We suggest that CoK used in combination with a midsize language model like Gemma 3 27B could be ideal for resource constrained settings, like a clinic without access to private LLM providers. As the CoK method nearly doubles (2n - 1) the amount of LLM calls compared to our other methods, we suggest its use is better suited for local hardware setups as using a pay-per-token LLM service could result in high costs.

Our user study revealed that LLM generated one-liners are preferred by humans and considered more complete, confirming LLMs' ability as strong data extractors, as well as their natural tendency to output well formatted, easy to read text. LLM based one-liners, however, do contain more irrelevant information, which is a known inductive bias for LLMs.

We acknowledge our work has limitations. During our user study, unblinding may have occurred, stemming from the grammatically correct and well formatted text produced by LLM contrasting with human written one-liners. Moreover, content provided in one-liners is not consistent between doctors, leading to another source of heterogeneity, which may have influenced our evaluations. Our sample size was only 60 cases, limiting the conclusions we can draw from our work. Lastly, we found no meaningful correlations between our evaluation metrics and the questions answered by our annotator during our user study, nor between the LLM-as-a-Judge score assessed by a human vs an LLM. This suggests further development and evaluation of a fitting scoring metric is needed.

5. Citations and Bibliography

References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. Large language models are few-shot clinical information extractors. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1998–2022, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.130. URL https://aclanthology.org/2022.emnlp-main.130/.

EunJeong Hwang, Yichao Zhou, James Bradley Wendt, Beliz Gunel, Nguyen Vo, Jing Xie, and Sandeep Tata. Enhancing incremental summarization with structured representations. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3830–3842, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.220. URL https://aclanthology.org/2024.findings-emnlp.220/.

Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *arXiv preprint arXiv:2212.14024*, 2022.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. Dspy: Compiling declarative language model calls into self-improving pipelines. 2024.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

Yujin Oh, Sangjoon Park, Hwa Kyung Byun, Yeona Cho, Ik Jae Lee, Jin Sung Kim, and Jong Chul Ye. LLM-driven multimodal target volume contouring in radiation oncology. 15(1): 9186, 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-53387-y. URL https://www.nature.com/articles/s41467-024-53387-y.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing instructions and demonstrations for multi-stage language model programs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9340–9366, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.525. URL https://aclanthology.org/2024.emnlp-main.525/.

Praveenbalaji Rajendran, Yizheng Chen, Liang Qiu, Thomas Niedermayr, Wu Liu, Mark Buyyounouski, Hilary Bagshaw, Bin Han, Yong Yang, Nataliya Kovalchuk, Xuejun Gu, Steven Hancock, Lei Xing, and Xianjin Dai. Autodelineation of treatment target volume for radiation therapy using large language model-aided multimodal learning. 121(1):230–240. ISSN 03603016. doi: 10.1016/j.ijrobp.2024.07.2149. URL https://linkinghub.elsevier.com/retrieve/pii/S0360301624029717.

Liyan Tang, Zhaoyi Sun, Betina Idnay, Jordan G. Nestor, Ali Soroush, Pierre A. Elias, Ziyang Xu, Ying Ding, Greg Durrett, Justin F. Rousseau, Chunhua Weng, and Yifan Peng. Evaluating large language models on medical evidence summarization. *npj Digital Medicine*, 6(1):158, August 2023. ISSN 2398-6352. doi: 10.1038/s41746-023-00896-7. URL https://doi.org/10.1038/s41746-023-00896-7.

Dave Van Veen, Cara Van Uden, Louis Blankemeier, Jean-Benoit Delbrouck, Asad Aali, Christian Bluethgen, Anuj Pareek, Malgorzata Polacin, Eduardo Pontes Reis, Anna Seehofnerová, Nidhi Rohatgi, Poonam Hosamani, William Collins, Neera Ahuja, Curtis P. Langlotz, Jason Hom, Sergios Gatidis, John Pauly, and Akshay S. Chaudhari. Adapted large language models can outperform medical experts in clinical text summarization. 30 (4):1134–1142. ISSN 1078-8956, 1546-170X. doi: 10.1038/s41591-024-02855-5. URL https://www.nature.com/articles/s41591-024-02855-5.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeHuCVFDr.

Appendix A. Methods: summarization and one liner generation specifics

For summarization using our local LLM endpoint, the temperature was set to 0 and number of tokens to 16384. For the Azure endpoint, we used the default settings, as o3-mini does not support temperature.

For DSPy optimized one-liner generation from summaries, we use the BootstrapFewShotWithRandomSearch optimizer. We set max_bootstrapped_demos to 3, max_labeled_demos to 6 and num_candidate_programs to 6. For each model - method combination, the prompt was optimized again.

Appendix B. Prompts and Statistics

Task Overview:
Your task involves synthesizing information from detailed clinical documentation about a specific patient case into a structured summary table. This JSON will highlight key medical attributes and findings along with their detailed descriptions derived from the provided clinical texts.

Instructions:
- Extract Clinical Values: Focus on extracting specific, detailed medical information rather than general or vague descriptors.
- Ensure that descriptions are clinically precise and informative.
- Present a Comprehensive View: The table should reflect a comprehensive clinical perspective, including normal, abnormal, and significant findings.
- For attributes with multiple assessments, indicate the sources supporting each finding.

Attribute Selection:
- Commonly Assessed Attributes: Include attributes that are generally of clinical interest for medical case documentation and patient care.
- Unique Attributes: Also identify and include unique clinical attributes that are specifically mentioned in the provided documentation.
- Do not include irrelevant information about different patients in the summary. Irrelevant information includes patient identifiers (PATIENT1, SUBJECT) that are different from the target patient case.

Structure of the Summary Table:
- The JSON should contain a dictionary format data, where keys are clinical attributes and values are detailed descriptions of their corresponding findings.
- List attributes with their corresponding values, indicating the source documentation and relevant clinical excerpts for substantiation.
- If an attribute has multiple values, include all values as a list of the attribute.
- Each value should contain sufficient clinical evidence extracted from the documentation related to the patient.
- The JSON should be only a single level.

Example:
Patient: PATIENT_001
Clinical Documentation:
- P1. Patient presents with chest pain and shortness of breath. Vital signs stable with BP 140/90. ECG shows ST elevation in leads II, III, aVF. Troponin levels elevated at 15.2 ng/mL.
- P2. Patient has history of hypertension and diabetes mellitus type 2. Current medications include metformin 500mg BID and lisinopril 10mg daily. Blood glucose well controlled.
- P3. Echocardiogram reveals regional wall motion abnormalities in inferior wall. Ejection fraction estimated at 45%. Patient reports 8/10 chest pain severity.

Summary JSON:
{
"attributes": {
"Presenting Symptoms": ["Chest pain", "Shortness of breath"],
"Vital Signs": ["Blood pressure 140/90"],
"Diagnostic Tests": ["ECG shows ST elevation in leads II, III, aVF", "Elevated troponin levels at 15.2 ng/mL"],
"Medical History": ["Hypertension", "Diabetes mellitus type 2"],
"Current Medications": ["Metformin 500mg BID", "Lisinopril 10mg daily"],
"Imaging Results": ["Regional wall motion abnormalities in inferior wall", "Ejection fraction 45%"],
"Pain Assessment": ["8/10 chest pain severity"]
}
}

Your Task: Generate a similar table based on the following clinical documentation of the specified patient case.
Patient: {entity_name}
Clinical Documentation: {paragraph}

Proceed to generate the clinical summary JSON

Figure 3: The Generate Once prompt used for summarization. Edited for readability.

I will provide a JSON format summary in a section called [**NEW SUMMARY**], and a class definition [**CLASS**], which define some fields that need to be generated, and an instantiation of that class under [**PARTIAL SUMMARY**] that is a response to the question in [**QUESTION**]. Your task is to propose updates to [**PARTIAL SUMMARY**] gathered from the information in [**NEW SUMMARY**].
There are two types of revisions that you can suggest: **ADD** and **UPDATE**.
For **UPDATE**, follow these instructions:

1. Your proposed updates must be for valid JSONPaths that already exist in [PARTIAL SUMMARY]. If the JSONPath does not exist, you should not propose an update for that JSONPath.
2. Updates can be made by modifying an existing value using content from [NEW SUMMARY].
3. Updates should never reduce the amount of information in [PARTIAL SUMMARY]
4. Never remove existing information from the [PARTIAL SUMMARY].
5. Proposed update must be a 'Dict[str, ProposedUpdate]' where the key is a valid JSONPath in [CLASS] and 'ProposedUpdate' is defined as follows:

   class ProposedUpdate(BaseModel):
   update: List[str]

For ADD, follow these instructions:

- Proposed additions must be for valid JSONPaths that adhere to the definition in [CLASS]. They are allowed to increase the size of lists in the definition, but they must not define new fields which are not defined in the class definition.
- It is OK to add partial objects. Leave fields unset if [NEW SUMMARY] does not contain a value for one of the fields in [PARTIAL SUMMARY].
- Proposed additions must be a 'Dict[str, ProposedAdd]' where the key is a valid JSONPath in [CLASS] and 'ProposedAdd' is defined as follows:

  class ProposedAdd(BaseModel):
  add: List[str]

For both operations, follow these instructions:

1. Values have sufficient context: the values of the [PARTIAL SUMMARY] should have enough context so a reader can understand what it means.
2. No redundant keys: If information from [NEW SUMMARY] can be incorporated by updating an existing key in [PARTIAL SUMMARY], then do not introduce a new redundant key.
3. No redundant values under the same key: If one value encompasses most of the details in another value, merge them together.

Example:

[QUESTION]
Merge the new clinical summary and existing clinical summary of PATIENT.
[NEW SUMMARY]
{
"attributes": {
"Vital Signs": ["Heart rate 88 bpm"],
"Medical History": ["Previous MI in 2019"],
"Laboratory Results": ["HbA1c 6.8%", "Troponin elevated at 15.2 ng/mL"],
"Pain Assessment": ["Chest pain 3/10 severity"]
}
}

[CLASS]
class Summary(BaseModel):
attributes: Dict[str, List[str]] # Keyed by clinical attribute, with a list of sufficient medical details about the attribute.
[PARTIAL SUMMARY]
{
"attributes": {
"Vital Signs": ["Blood pressure 140/90"],
"Medical History": ["Hypertension", "Diabetes mellitus type 2"]
}
}

Figure 4: The Chain of Key prompt used for summarization part 1. Edited for readability.

[THOUGHTS FOR UPDATE]

1. I need to figure out which fields and values to update.
2. [PARTIAL SUMMARY] contains information about the following: ["Vital Signs", "Medical History"]
3. [NEW SUMMARY] contains new content relevant to the following existing content: ["Vital Signs", "Medical History"]
4. The content should be updated at the following JSONPaths: [".*attributes.VitalSigns*","attributes.Medical History"]

[THOUGHTS FOR ADD]

1. I need to figure out which fields and values to add.
2. [NEW SUMMARY] mentions information about the following: ["Laboratory Results", "Pain Assessment"]
3. [PARTIAL SUMMARY] does not yet have information about: ["Laboratory Results", "Pain Assessment"]
4. The content should be added at the following JSONPaths: ["$.attributes.Laboratory Results", "$.attributes.Pain Assessment"]

1.
Output:
{
"UPDATE":
{
"*.attributes.VitalSigns*" : {
"*update*" : ["*Heartrate*88*bpm*"]
},
".attributes.Medical History": {
"update": ["Previous MI in 2019"]
} },
"ADD":
{
"*.attributes.LaboratoryResults*" : {
"*add*" : ["*HbA*1*c*6.8%","*Troponinelevatedat*15.2*ng/mL*"]
},
".attributes.Pain Assessment": {
"add": ["Chest pain 3/10 severity"]
} } }
Your Task:
[QUESTION]
Merge the new clinical summary and existing clinical summary of {entity_name}.

[NEW SUMMARY]
{new_summary}

[CLASS]
class Summary(BaseModel):
attributes: Dict[str, List[str]] # Keyed by attribute, with a list of sufficient details about the attribute.

[PARTIAL SUMMARY]
{existing$_{s}ummary$}

[THOUGHTS FOR UPDATE]

1. I need to figure out which fields and values to update.
2. [PARTIAL SUMMARY] contains information about the following: {existing_keys}
3. [NEW SUMMARY] contains new content relevant to the following existing content: {relevant_keys}
4. The content should be updated at the following JSONPaths: {update_paths}

[THOUGHTS FOR ADD]

1. I need to figure out which fields and values to add.
2. [NEW SUMMARY] mentions information about the following: {new_keys}
3. [PARTIAL SUMMARY] does not yet have information about: {missing_keys}
4. The content should be added at the following JSONPaths: {add_paths}

Output: { "UPDATE": {}, "ADD": {} }
Return a json with the added and updated objects

Figure 5: The Chain of Key prompt used for summarization part 2. Edited for readability.

Your input fields are:

1. `note` (str):

Your output fields are:

1. `reasoning` (str):
2. `summary` (str):

All interactions will be structured in the following way, with the appropriate values filled in.

```
[[ ## note ## ]]
{note}
[[ ## reasoning ## ]]
{reasoning}
[[ ## summary ## ]]
{summary}
[[ ## completed ## ]]
```

In adhering to this structure, your objective is:
Please analyze the provided medical record or clinical documentation about a patient's case, and generate a concise yet comprehensive summary. Your summary should capture essential information such as patient demographics, relevant medical history, significant diagnostic findings (including laboratory results, imaging studies, and diagnostic procedures), clinical presentations, and treatment interventions. Ensure that key clinical details are clearly stated using appropriate medical terminology and standard abbreviations, maintaining clinical accuracy and completeness.

Figure 6: The optimized DSPy prompt used for summarization

Table 2: Spearman Rank Correlation Between Human Scores and LLM bases scores. Reported as spearman correlation (p value).

Metric Question	BERTScore	LLM-Judge	ROUGE-L
Total Human Score	-0.151 (0.276)	0.011 (0.939)	-0.004 (0.98)
Which summary contains less non-important information?	-0.096 (0.492)	-0.127 (0.36)	0.143 (0.302)
Which summary includes less false information?	-0.316 (**0.02**)	-0.064 (0.647)	-0.135 (0.329)
Which summary more completely captures important information?	0.067 (0.632)	0.16 (0.248)	0.028 (0.843)
Which summary would you prefer given the patient presentation?	-0.111 (0.426)	0.118 (0.394)	-0.086 (0.536)

Objective: Evaluate the accuracy of a candidate clinical blurb in comparison to a reference clinical blurb written by an expert. Process Overview: You will be presented with:

1. The reference clinical blurb.
2. The candidate clinical blurb.
3. A set of questions to guide your evaluation.

Key Considerations:

- Focus on the accuracy of the clinical content, not the writing style.
- If the reference blurb does not include certain information, mark the question as "Not mentioned in reference." Such questions will not count toward the final score.
- Missing critical information from the reference blurb in the candidate blurb should be flagged as "No."
- The LLM should not calculate or output the score. The score will be calculated manually based on the evaluation results.

Reference Blurb:
{ground_truth}
Candidate Blurb:
{prediction}

Evaluation Questions:

1. **Patient Demographics**: Does the candidate blurb correctly state the patient's age and gender?
   Yes / No / Not mentioned in reference
2. **Performance Status (e.g., ECOG)**: Does the candidate blurb correctly state the patient's performance status (if mentioned in the reference)?
   Yes / No / Not mentioned in reference
3. **Diagnosis**: Does the candidate blurb correctly state the diagnosis (e.g., adenocarcinoma, tumor type)?
   Yes / No / Not mentioned in reference
4. **Tumor Staging**: Does the candidate blurb correctly state the tumor staging (e.g., T1aN1M0)?
   Yes / No / Not mentioned in reference
5. **Lymph Node Involvement**: Does the candidate blurb correctly identify lymph node involvement (e.g., 4R)?
   Yes / No / Not mentioned in reference
6. **Prior Radiation/Oncological Treatment** Does the candidate blurb correctly document any prior radiation therapy or oncological treatments (e.g., chemotherapy, immunotherapy, surgery, or previous radiation fields and doses)?
   Yes / No / Not mentioned in reference
7. **Other Relevant Clinical Details**: Does the candidate blurb correctly state any other relevant clinical details (if mentioned in the reference)?
   Yes / No / Not mentioned in reference

Output Format: Provide your evaluation in the following JSON format:

```
{
    "Patient_Demographics": "<Yes/No/Not mentioned in reference>",
    "Performance_Status": "<Yes/No/Not mentioned in reference >",
    "Diagnosis": "<Yes/No/Not mentioned in reference >",
    "Tumor_Staging": "<Yes/No/Not mentioned in reference >",
    "Lymph_Node_Involvement": "<Yes/No/Not mentioned in reference >",
    "Other_Clinical_Details": "<Yes/No/Not mentioned in reference >"
}
```

Figure 7: The prompt use by our LLM-as-a-Judge. Minor adjustments made for formatting.

Table 3: Comparison of Techniques Across Models and Metrics. Each cell shows the mean difference between the two techniques, with the p-value in parentheses. Significant p-values ($p < 0.05/3$) (Bonferroni correction) are bolded. Positive differences indicate that the first technique outperformed the second technique.

Techniques	Model	ROUGE-L	BERTScore	LLM-Judge
CoK vs Generate Once	Combined	0.029 (**0.000**)	0.019 (**0.000**)	0.098 (**0.000**)
	o3-mini	0.010 (0.141)	0.009 (0.074)	0.050 (0.213)
	Gemma 3 27B	0.039 (**0.000**)	0.023 (**0.001**)	0.107 (**0.009**)
	DeepSeek R1 8B	0.038 (**0.000**)	0.025 (**0.000**)	0.137 (**0.007**)
CoK vs DSPY	Combined	0.007 (0.477)	0.009 (0.066)	0.025 (0.339)
	o3-mini	-0.015 (0.052)	-0.003 (0.200)	0.016 (0.595)
	Gemma 3 27B	0.024 (**0.011**)	0.019 (**0.008**)	0.053 (0.155)
	DeepSeek R1 8B	0.012 (0.425)	0.013 (0.173)	0.007 (0.830)
DSPY vs Generate Once	Combined	0.022 (**0.000**)	0.010 (**0.000**)	0.073 (**0.007**)
	o3-mini	0.025 (**0.005**)	0.012 (**0.001**)	0.035 (0.490)
	Gemma 3 27B	0.016 (0.070)	0.005 (0.371)	0.053 (0.312)
	DeepSeek R1 8B	0.026 (**0.004**)	0.013 (**0.013**)	0.131 (**0.011**)

Table 4: Comparison of Models Across Techniques and Metrics. Each row shows the mean difference and p-value (in parentheses) for ROUGE-L, BERTScore, and LLM-Judge. Positive differences indicate that the first model in the comparison performs better, while negative differences indicate that the second model performs better. Comparisons with Gemma 3 were made on 58 samples instead of 60. Significant values ($p < 0.05/3$) (Bonferroni correction) are bolded.

Comparison	Technique	ROUGE-L	BERTScore	LLM-Judge
DeepSeek R1 8B vs O3-mini	All Techniques	-0.032 (**0.000**)	-0.021 (**0.000**)	-0.151 (**0.000**)
	CoK	-0.015 (0.113)	-0.011 (0.026)	-0.128 (**0.000**)
	DSPy	-0.039 (**0.000**)	-0.026 (**0.000**)	-0.113 (0.033)
	Generate Once	-0.043 (**0.000**)	-0.026 (**0.000**)	-0.211 (**0.000**)
o3-mini vs Gemma 3 27B	All Techniques	0.021 (**0.002**)	0.014 (**0.000**)	0.116 (**0.000**)
	DSPy	0.036 (**0.001**)	0.024 (**0.000**)	0.121 (**0.009**)
	CoK	-0.001 (0.760)	0.002 (0.401)	0.088 (0.043)
	Generate Once	0.028 (0.038)	0.016 (0.020)	0.138 (**0.007**)
Deepseek R1 8B vs Gemma 3 27B	All Techniques	-0.011 (0.152)	-0.007 (0.163)	-0.035 (0.194)
	DSPy	-0.002 (0.730)	-0.002 (0.460)	0.007 (0.632)
	CoK	-0.016 (0.066)	-0.009 (0.187)	-0.040 (0.284)
	Generate Once	-0.014 (0.342)	-0.010 (0.082)	-0.073 (0.070)

Metric	Mean ± SD
Mean number of words	6522.50 ± 3777.95
Mean number of tokens	12682.38 ± 7379.79

Table 5: Lengths of the concatenated patient notes as number of words and number of tokens. Number of tokens calculated as BPE tokens, using OpenAI's `o200k_base`)