Large Language Models for Trauma Note Quality: NTDS Complication Abstraction

Kaijie Zhang
kaz029@ucsd.edu

Viv Somani
visomani@ucsd.edu

Aaron Boussina
aboussina@health.ucsd.edu

Abstract

The American College of Surgeons Committee on Trauma requires trauma centers to maintain surgical quality scores, which are critical for reporting trauma quality metrics. However, traditional registries are labor-intensive and costly. We use the Llama 8B Instruction model on AWS to streamline this process and tested out a variety of test-time inference techniques, including best-of-N with Chain of Thought prompting, beam search, and Diverse Verifier Tree Search (DVTS). Out of these, we obtained the best results with best-of-N with tree search.

Code:
https://github.com/KaijieZhang0831/TQIP-Trauma-Note-LLM-Extraction

1 Introduction

1.1 Overview

An effective surgical quality program relies heavily on maintaining a comprehensive patient registry to track outcomes and benchmark against national standards. The American College of Surgeons Committee on Trauma requires trauma centers to maintain these registries for accreditation, which are critical for reporting trauma quality metrics. However, traditional registries are labor-intensive and costly. Artificial intelligence, particularly Large Language Models (LLMs), offer us a potential solution to streamline this process. We hypothesized that a LLM could be applied to review patient charts and identify complications as defined by the Trauma Quality Improvement Program, offering an effective adjunct to manual chart reviews.

1.2 Prior Work

Over the last two decades trauma quality improvement programs (TQIPs), such as the American College of Surgeons (ACS) TQIP, have proven to be essential in ensuring that participating trauma centers maintain the highest standards of care. Through participation in the ACS Committee on Trauma's (COT) Verification, Review and Consultation (VRC) program, trauma centers undergo a rigorous formal assessment to verify that they are in line with all or most criteria to earn the classification of a Level I, II, or III trauma center [Resources for Optimal Care of the Injured Patient]. At the national level, this standardization improves the care of trauma patients by ensuring that institutions are well-equipped to care for patients with complex and multi-faceted injuries. Trauma centers subsequently benefit from verification and participation in this TQIP via the program's regular benchmark reports that allow them to identify deficiencies in their care processes or quality metrics relative to national rates and similarly verified institution's, allowing program leadership to implement process improvement initiatives (Hemmila et al. 2010). As a result, the ACS TQIP has left an indelible mark in the advancement of care for trauma patients at ACS verified trauma centers, with a recent study demonstrating that high performing centers (those in the lowest decile of overall risk-adjusted mortality) were more likely to be adherent to several VRC quality metrics (Cho et al. 2025).

Accurate benchmarking, quality-measure adherence, and self-assessment is dependent on the integrity of data reported from participating trauma centers (Nathens, Cryer and Fildes 2012). The ACS VRC program requires all centers, regardless of level, to maintain a trauma registry and have a written data quality plan to validate that the data being reported are high-quality. Meeting these data reporting standards can be costly, with the VRC program requiring 0.5 full time equivalents (FTEs), or registrars, per 200-300 annual patient entries that meet National Trauma Data Standard (NTDS) inclusion criteria. For most level I centers, maintenance this means employing upwards of 5.0 FTEs to keep up with patient volume, with level II centers requiring about 3.0 FTEs (Elkbuli, McKenney et al. 2020). In addition to the considerable upfront financial costs (Moore and Clark 2008), it also takes considerable time and effort to train trauma registrars to meet the requirements dictated by the ACS COT (Nathens, Cryer and Fildes 2012). ACS requires registrars to attend and pass several mandated courses. Additionally, the VRC program requires that 80 percent of patient records be completed within 60 days after patient discharge. In addition to the requirements of the ACS trauma registry, many trauma centers also participate in regional or state registry, each with their own data requirements (Hemmila et al. 2017) (Hemmila et al. 2018). These certification requirements in combination with wages and a persistently growing patient data backlogs can lead to a high rate of trauma registrar turnover (Day 2012). Although well-funded trauma centers may be able to shoulder the burden of high turnover rates among registry personnel, smaller trauma centers may find it difficult to keep up with the demands of high-quality patient data entry. These challenges are often cited as a barrier to the implementation of trauma registries in middle- and low-income countries (Bommakanti et al. 2018) (Purcell et al. 2020) (Klappenbach et al. 2024).

In an effort to facilitate data digestibility and registry-related work, many institutions have made efforts to manipulate or configure provider documentation to improve the ease of chart review and data-entry. However, this often results in additional documentation burden being placed in a field of surgery already rife with documentation issues (Ludley et al. 2023). Artificial intelligence (AI), and specifically large language models (LLMs), is one way trauma centers can reduce the time, effort, and cost it takes to maintain a trauma registry. LLMs have gained recent acclaim in medicine due to their ability to be trained on and pass board exams for several specialties and even show promise in diagnosis (Mahajan et al. 2025), (Alessandri-Bonetti et al. 2024). More recently, they have demonstrated their promise in the abstraction of complex hospital quality measures (Boussina et al. 2024). Based on this work, we hypothesized that LLMs could be used to streamline the identification of complications as defined by the NTDS, offering a faster and more cost-effective alternative to manual chart reviews performed by trauma registrars.

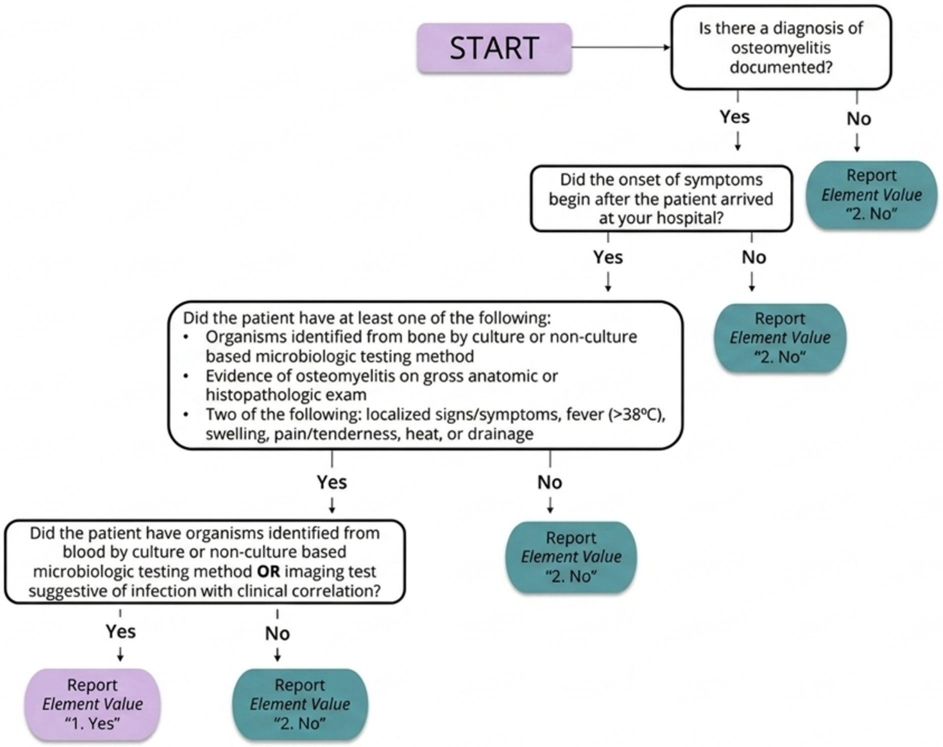
START
Is there a diagnosis of osteomyelitis documented?
Yes
No
Report Element Value "2. No"
Did the onset of symptoms begin after the patient arrived at your hospital?
Yes
No
Report Element Value "2. No"
Did the patient have at least one of the following:
• Organisms identified from bone by culture or non-culture based microbiologic testing method
• Evidence of osteomyelitis on gross anatomic or histopathologic exam
• Two of the following: localized signs/symptoms, fever (>38ºC), swelling, pain/tenderness, heat, or drainage
Yes
No
Report Element Value "2. No"
Did the patient have organisms identified from blood by culture or non-culture based microbiologic testing method OR imaging test suggestive of infection with clinical correlation?
Yes
No
Report Element Value "1. Yes"
Report Element Value "2. No"

Figure 1: This is the labeling procedure for osteomyelitis given in TQIP, one of the eighteen conditions we tasked our LLM with labeling. Several factors need to be present for a patient for a true positive label.

1.3 Relevant Data

Clinical data, especially trauma notes and medication orders, are protected from UCSD Health and sensitive; all of our data, the patient features, were stored in a monitored and protected environment. The Prompt with Chain of Thought Reasoning was created based on the National Trauma Data Standard Data Dictionary 2025 Admission as the instruction. The LLM we used is ”us.deepseek.r1-v1:0” and the embedding we used is ”amazon.titan-embed-text-v2:0” via Amazon Bedrock.

2 Methods

2.1 Data and LLMs

We included all patient encounters from a large academic level 1 trauma center with a complication in the registry from January 1, 2023 through October 31, 2024. This was a convenience sample selected based on availability of data exported from the trauma registry. Institutional review board (IRB) approval was obtained with waiver of informed con-

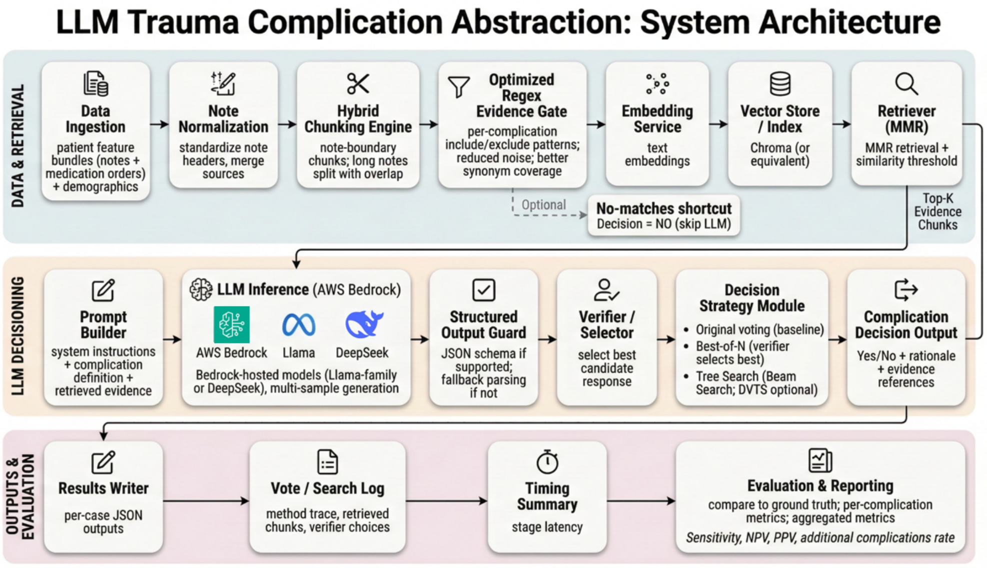
LLM Trauma Complication Abstraction: System Architecture
DATA & RETRIEVAL
Data Ingestion
patient feature bundles (notes + medication orders) + demographics
Note Normalization
standardize note headers, merge sources
Hybrid Chunking Engine
note-boundary chunks; long notes split with overlap
Optimized Regex Evidence Gate
per-complication include/exclude patterns; reduced noise; better synonym coverage
Embedding Service
text embeddings
Vector Store / Index
Chroma (or equivalent)
Retriever (MMR)
MMR retrieval + similarity threshold
Optional
No-matches shortcut
Decision = NO (skip LLM)
Top-K Evidence Chunks
LLM DECISIONING
Prompt Builder
system instructions + complication definition + retrieved evidence
LLM Inference (AWS Bedrock)
AWS Bedrock
Llama
DeepSeek
Bedrock-hosted models (Llama-family or DeepSeek), multi-sample generation
Structured Output Guard
JSON schema if supported; fallback parsing if not
Verifier / Selector
select best candidate response
Decision Strategy Module
• Original voting (baseline)
• Best-of-N (verifier selects best)
• Tree Search (Beam Search; DVTS optional)
Complication Decision Output
Yes/No + rationale + evidence references
OUTPUTS & EVALUATION
Results Writer
per-case JSON outputs
Vote / Search Log
method trace, retrieved chunks, verifier choices
Timing Summary
stage latency
Evaluation & Reporting
compare to ground truth; per-complication metrics; aggregated metrics
Sensitivity, NPV, PPV, additional complications rate

Figure 2: Full Pipeline. This is the process our architecture takes, from retrieving data to the LLM making decisions to having outputs checked.

sent (#808297). We followed STROBE guidelines to ensure appropriate reporting of this research (Von Elm et al. 2007). Clinical notes were retrieved from the institution's electronic health record using the Fast Healthcare Interoperability Resources (FHIR) standard version R4 (Figure 1). We utilized the Llama 3.1 8B Instruct model (meta.llama3-1-8b-instruct-v1:0) deployed via Amazon Bedrock, coupled with Retrieval Augmented Generation (RAG) to handle the extensive length of clinical notes. Notes were first filtered using regular-expressions based fuzzy matching against an augmented term list for 18 specific complications, including unplanned admission to ICU, unplanned intubation, severe sepsis, delirium, pressure ulcer, stroke, alcohol withdrawal, cardiac arrest with cardiopulmonary resuscitation (CPR), deep venous thromboembolisms (DVTs), acute kidney injury (AKI), unplanned visits to the operating room (OR), pulmonary embolisms (PE), catheter-associated urinary tract infections (CAUTI), myocardial infarctions (MI), ventilator-associated pneumonia (VAP), acute respiratory distress syndrome (ARDS), osteomyelitis, and superficial surgical site infection. The keyword list was initially constructed from clinical terminology and subsequently expanded using an LLM prompted with the relevant NTDS complication definitions to generate candidate synonyms and clinical variants, which were then reviewed and curated by a human. The resulting notes were then chunked into segments using a hybrid strategy: each note was treated as a single chunk by default to preserve clinical context and note boundaries, with notes exceeding 1200 characters further subdivided using fixed-size chunking with an overlap of [PLACEHOLDER: chunk size / overlap]. Chunks were converted into vectors using the Amazon Titan Embed Text v2 model (Amazon Web Services 2024). RAG was performed using maximal marginal relevance (MMR) similarity search between the embeddings and the prompt, selecting the top 12 most similar chunks into the final prompt (Carbonell and Goldstein 1998). MMR was chosen to capture diverse segments of text from highly redundant clinical notes.

The abstraction guidelines from the 2025 NTDS Data Dictionary informed the design of prompts for the LLM to identify each of the 18 complications. Each complication was represented as a structured decision tree, wherein each node encodes a discrete binary criterion aligned with the NTDS abstraction logic. This design standardizes LLM reasoning to produce traceable, stepwise outputs consistent with clinical abstraction guidelines.

2.2 Test-Time Inference Strategies

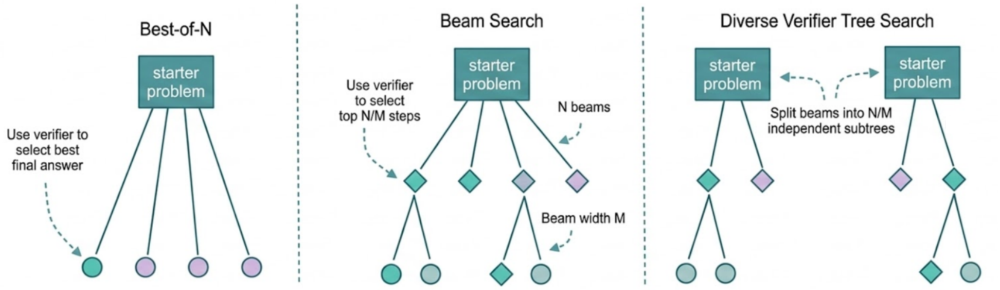
Best-of-N
starter problem
Use verifier to select best final answer
Beam Search
starter problem
Use verifier to select top N/M steps
N beams
Beam width M
Diverse Verifier Tree Search
starter problem
starter problem
Split beams into N/M independent subtrees

Figure 3: Differences between beam search and DVTS. DVTS enables larger diversitiy of nodes to be considered

We evaluated four inference strategies applied at test time to produce binary complication labels.

The first strategy, *majority voting*, served as the baseline. The model was prompted with a single holistic instruction per complication and queried k times with a temperature of 0.3; a complication was considered present if the LLM identified it in at least two of five responses (Snell et al. 2024). The LLM additionally provided rationale for each determination, citing specific text from the EHR.

The second strategy, *best-of-N*, generated N candidate responses to the same holistic prompt and subsequently employed a separate verifier prompt—supplied to the same Llama model—to evaluate and select the single most internally consistent and clinically coherent candidate. This approach reduces the variance introduced by sampling while avoiding the aggregation assumptions of majority voting.

The third strategy, *beam search*, operated on the decision tree in a node-by-node fashion. At each binary decision node, multiple candidate continuations were sampled and scored using a proxy confidence derived from the proportion of affirmative responses across repeated node-level queries. Since Amazon Bedrock does not expose token-level log probabilities, this proportion-based score served as a local self-consistency estimate; the log of this proportion was accumulated across nodes so that paths with consistently high local agreement received higher cumulative scores. The top-scoring partial paths were retained at each step, with the final label determined by the best surviving path at termination.

The fourth strategy, *Diverse Verifier Tree Search* (DVTS), extended beam search by partitioning the search budget across independent subtrees to preserve path diversity. Node-level

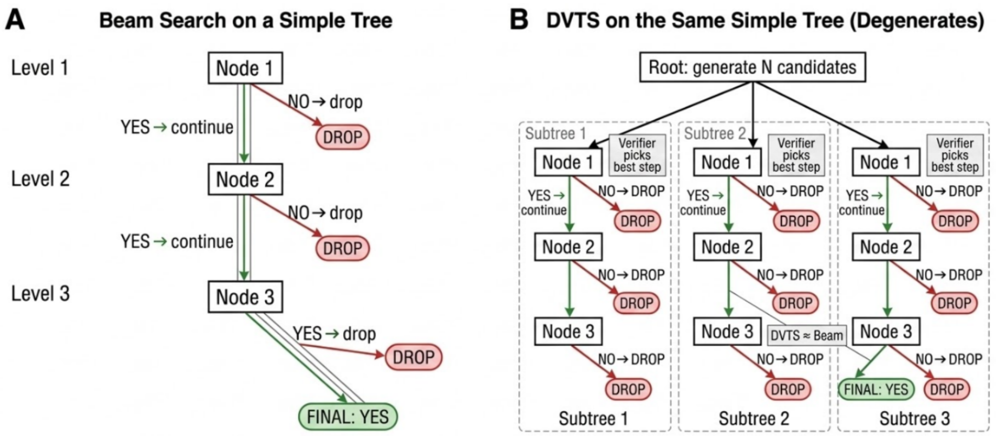

Figure 4: TQIP Labeling. Four techniques were used in total, including our baseline of majority voting and the three depicted techniques.

scoring used the same proxy confidence as beam search; however, rather than pruning to the single highest-scoring frontier, DVTS maintained distinct subtree explorations to mitigate premature convergence. We note that DVTS can degenerate in complications with highly constrained decision logic—such as VAP, which exhibits a single dominant criterion sequence at most nodes—reducing its effective diversity benefit.

All four strategies operated under identical retrieval and data conditions to enable direct comparison.

2.3 Statistical Analysis

Our primary outcome was agreement between the LLM and manual reviews performed by the institution's trauma registry. We assessed sensitivity, negative predictive value (NPV), positive predictive value (PPV), and frequency of complications identified by the LLM but not the registrar. Additionally, a subset of cases and output from the LLM was reviewed and validated by clinical subject matter experts (SMEs) via manual chart review. Cases in which the LLM missed complications that the human registrars identified were specifically included in this subset. Rationale provided by the LLM facilitated targeted reviews and verification of output as true or false. Statistical analysis was performed in Python version 3.9.11.

3 Results

3.1 Performance

Table 1: Complication-level performance summary

Complication	Sensitivity	PPV	NPV
Alcohol Withdrawal Syndrome	0.429	0.750	0.840
Delirium	0.895	0.370	0.951
DVT/Thrombophlebitis	0.625	0.625	0.940
Stroke/CVA	0.733	0.314	0.971
Unplanned Intubation	0.889	0.267	0.982
Unplanned Admission to ICU	0.750	0.340	0.950
Severe Sepsis	1.000	0.262	1.000
Pressure Ulcer	0.714	0.128	0.985
Cardiac Arrest with CPR	1.000	0.500	1.000
Acute Kidney Injury	0.778	0.206	0.986
Unplanned Visit to OR	1.000	0.224	1.000
Pulmonary Embolism	0.500	0.040	0.993
Myocardial Infarction	0.667	0.111	0.994
VAP	0.333	0.167	0.947
ARDS	0.800	0.471	0.987
CAUTI	0.500	0.154	0.988
Osteomyelitis	1.000	0.400	1.000
Superficial Incisional SSI	0.667	0.111	0.994
Overall Sensitivity	**0.736 (190/258)**		
Total TP / FP / FN / TN	**190 / 455 / 68 / 2419**		
Average Additional Complications	**261.49%**		

As a simple smoke test for AWS Bedrock with new embedding and LLMs, here is the pre-Embedding and pre-LLM optimization (raw) of experiment on 20 patient features (During extended runs we encountered several technical issues that required additional time to diagnose and stabilize, therefore we used a subset of 20 samples as a preliminary peek at the system behavior):

Overall sensitivity is 73.6% with 190 true positives and 68 false negatives. This indicates that the pipeline retains a meaningful portion of true complications, and recall is moderate rather than collapsed. As a screening system this is not disastrously low, but it is still not strong enough to support a high confidence registry workflow, especially given the remaining missed complications across multiple categories.

The average percentage of additional complications is 261.49%, which shows that the model is generating far more positive complication labels than the ground truth supports. With 190 true positives and 455 false positives, the overall positive predictive value is only about

29.5%. In practical terms, most predicted positives are still incorrect, so the system remains heavily over triggering. This would translate into substantial manual review burden, high trust cost, and limited usability in a real abstraction setting.

Several complications show a strong recall oriented but low precision pattern. For example, unplanned visit to OR reaches sensitivity 1.000 but has PPV only 0.224, meaning it captures all true cases while greatly over predicting. Severe sepsis also reaches sensitivity 1.000 with PPV 0.262, and pressure ulcer reaches sensitivity 0.714 with PPV 0.128, again suggesting broad triggering with weak definition control. At the same time, other complications remain weak on both sides: VAP has sensitivity 0.333 and PPV 0.167, while pulmonary embolism has sensitivity 0.500 and PPV only 0.040. These patterns suggest that the system is not uniformly biased in one direction, but instead suffers from complication specific miscalibration, where some labels are over activated and others remain under captured.

Although this run is much larger and more stable than the earlier small scale tables in the mid-quarter, the main bottleneck is still precision rather than recall. The system shows that it can detect many true complications, but it does so with excessive positive spillover and weak label discipline. The dominant risk is therefore not a total failure to find complications, but rather systematic over prediction caused by loose thresholding and imperfect definition alignment. In its current form the pipeline is still not suitable for deployment, and the next priority should be reducing false positive inflation while preserving the current recall structure as much as possible.

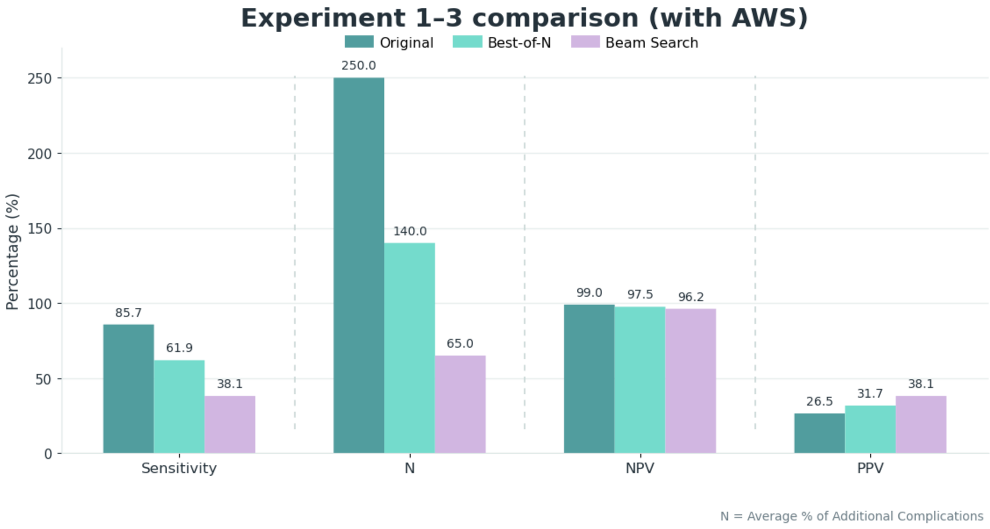

Figure 5: Comparison between scores for three techniques including all 18 symptoms. As we move from the original to the best of n to the beam search, we see a consistent finding where there are fewer false positives but also a lower sensitivity.

Table 2: Results by task, including Alcohol Withdrawal Syndrome

Experiment	Sens. (%)	N (%)	NPV (%)	PPV (%)	Time (s)
Original (No CoT, 5-vote majority)	85.71	250.00	98.97	26.47	1425.93
Best-of-N using CoT (3 candidates)	61.90	140.00	97.49	31.71	1476.67
Beam Search using atomic decision tree (3 candidates, beam width = 3)	38.10	65.00	96.17	38.10	1292.40

Table 3: Results by task, not including Alcohol Withdrawal Syndrome

Experiment	Sens. (%)	N (%)	NPV (%)	PPV (%)	Time (s)
Original (No CoT, 5-vote majority)	85.71	314.29	99.30	21.43	1425.93
Best-of-N using CoT (3 candidates)	78.57	200.00	99.00	28.21	1476.67
Beam Search using atomic decision tree (3 candidates, beam width = 3)	57.14	85.71	98.13	40.00	1292.40

Best-of-N using Chain of Thought substantially improves negative-side performance, reducing false positives by (44.0%) and additional complications by 44.0% relative to original. If AWS is excluded, Best-of-N still improves True Negatives by about (5.7%) while simultaneously reducing FP by about 36.4%, making it the most balanced configuration overall.

Beam Search pushes this trend even further, achieving the strongest negative filtering, with a 74.0% reduction in FP and a 74.0% reduction in additional complications relative to Original, while also slightly improving runtime. However, this comes with a much larger recall penalty, so Beam Search is better interpreted as an extreme high true negative operating point rather than the main screening configuration.

3.2 Time Analysis

Table 4: Runtime Breakdown of the Pipeline of the Baseline with Simple Voting (5 cases)

Module	**Total Time (sec)**	**Mean Time**	**% of Total Runtime**
LLM Engine	690.08	12.32 / call	79.7%
Vectorstore Build (Embedding)	166.67	3.09 / build	19.3%
Retriever Invoke	7.56	0.14 / query	0.9%
Chunk Filtering	0.52	0.006 / call	0.06%
Text Splitting	0.015	0.003 / case	0.002%
Other Overhead	0.40	–	0.05%
Total Runtime (5 cases)	865.25 sec (100%)		

Table 5: Runtime Breakdown of the Pipeline of Best-of-N with CoT (124 cases)

Module	**Total Time (sec)**	**Mean Time**	**% of Total Runtime**
LLM Engine	4145.88	2.528 / call	27.73%
LLM Verifier	681.35	0.452 / call	4.56%
Vectorstore Build	9642.38	5.880 / build	64.48%
Retriever Invoke	238.82	0.146 / query	1.60%
Chunk Filtering	220.26	0.099 / question	1.47%
Text Splitting	1.34	0.011 / case	0.01%
Note Preparation	0.21	0.002 / case	0.00%
Vectorstore Delete	21.67	0.013 / delete	0.14%
Write Case JSON	0.17	0.001 / case	0.00%
Other Overhead	1.46	0.012 / case	0.01%
Total Runtime (124 cases)	**14953.56 sec**	**120.59 / case**	**100%**

The time for the LLM Engine to perform initial inference and to build the vector store was the longest. For simple voting, initial LLM Inference was the largest bottleneck. For the best-of-n with chain of thought, the LLM produced results with a substantially lower mean time, but building the vector store took longer compared to in simple voting. Therefore, building the vector store took the longest time, which was unexpected. All other tasks, such as invoking the retriever, verifying LLM outputs, filtering chunks, etc. took substantially less time.

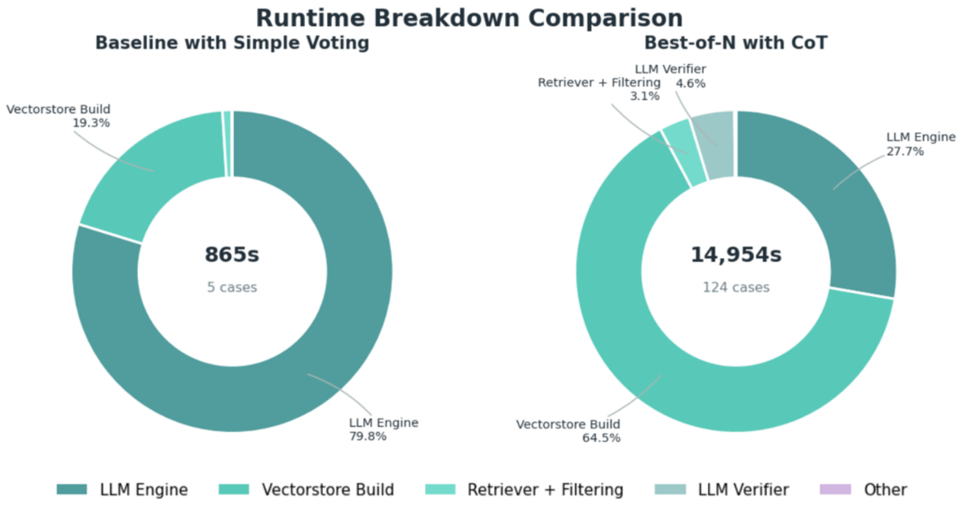

Figure 6: Alternative View of Time Analysis. Seconds in the middle represent total time taken.

4 Discussion

Our experiments show that Best-of-N achieves a substantially higher PPV with a relatively small decrease in NPV and sensitivity. However, when moving to Beam Search, the sensitivity falls off significantly further. This is a kind of limitation. We suspet the atomic design makes the abstraction process too strict. As a result, some features that actually contain complications fail to identify even a single complication. This becomes the main reason why Beam Search, and later DVTS, produce more false negatives than Best-of-N with Chain of Thought (CoT). This type of strictness is difficult to relax through prompt modification, which we tried, and attempts to loosen it often remove the reduced true negatives. A possible direction is that if no complication is identified under CoT, the system could fall back to running Best-of-N with CoT once and adopt that result. If still no complication is found, then the feature likely does not contain a complication. In the future we could combine these scaling methods more strategically to achieve a better trade-off.

For DVTS, we ended up getting very simple decision trees for most of the eighteen complications. Typically a Yes continues to the next node while a No immediately exits and outputs No. Because of this structure, the surviving subtrees are almost always all Yes paths. This limits the advantage of DVTS and causes it to behave very similarly to Beam Search. Only in more complex trees such as Ventilator Associated Pneumonia (VAP) does DVTS show a noticeably different effect. We believe that improving the decision structures for specific complications would be an important direction for future work.

5 Conclusion

We have tested several test-time inference techniques to perform condition labelling for TQIP. To our knowledge, these techniques had not been tried previously. Overall, we found that best-of-N with chain of thought performed the best.

5.1 Contributions

Kaijie Zhang: I was responsible for the model inference pipeline, implementation, and experimental evaluation of the different test-time methods, excluding the filter optimization component. I also led all visualization work and created the poster, including the figures, result presentation, and overall visual layout.

Viv Somani: I created an improved scheme to perform initial regular expressions chunk filtering. In addition, I wrote the introduction and methods section – and assisted in writing several other sections of this report.

References

Alessandri-Bonetti, Mario, Hilary Y Liu, James M Donovan, Jenny A Ziembicki, and Francesco M Egro. 2024. "A comparative analysis of ChatGPT, ChatGPT-4, and Google Bard performances at the Advanced Burn Life Support exam." *Journal of Burn Care & Research* 45 (4): 945–948

Amazon Web Services. 2024. "Amazon Titan Models: Titan Embed Text v2." https://aws.amazon.com/bedrock/titan/

Bommakanti, Krishna, Isabelle Feldhaus, Girish Motwani, Rochelle A Dicker, and Catherine Juillard. 2018. "Trauma registry implementation in low-and middle-income countries: challenges and opportunities." *Journal of surgical research* 223 : 72–86

Boussina, Aaron, Rishivardhan Krishnamoorthy, Kimberly Quintero, Shreyansh Joshi, Gabriel Wardi, Hayden Pour, Nicholas Hilbert, Atul Malhotra, Michael Hogarth, Amy M Sitapati et al. 2024. "Large language models for more efficient reporting of hospital quality measures." *Nejm ai* 1 (11), p. AIcs2400420

Carbonell, Jaime, and Jade Goldstein. 1998. "The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries." In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Cho, Nam Yong, Jeff Choi, Saad Mallick, Galinos Barmparas, David Machado-Aranda, Areti Tillou, Daniel Margulies, Peyman Benharash, Academic Trauma Research Consortium et al. 2025. "Beyond American college of surgeons verification: quality metrics associated with high performance at level I and II trauma centers." *Journal of the American College of Surgeons* 240 (2): 190–200

Elkbuli, Adel, Mark McKenney et al. 2020. "Trauma Registry Staffing and Costs: A National Survey of Level I and Level II Trauma Centers." *Annals of Surgery*

Hemmila, Mark R et al. 2017. "Association of Hospital Participation in a Regional Trauma Quality Improvement Collaborative With Patient Outcomes." *JAMA Surgery* 152 (8): 743–753

Hemmila, Mark R et al. 2018. "Collaborative Quality Improvement for Trauma: The Michigan Experience." *Journal of Trauma and Acute Care Surgery* 85 (1): 200–207

Hemmila, Mark R, Avery B Nathens, Shahid Shafi, J Forrest Calland, David E Clark, H Gill Cryer, Sandra Goble, Christopher J Hoeft, J Wayne Meredith, Melanie L Neal et al. 2010. "The Trauma Quality Improvement Program: pilot study and initial demonstration of feasibility." *Journal of Trauma and Acute Care Surgery* 68 (2): 253–262

Klappenbach, H. et al. 2024. "Trauma Registries in Low- and Middle-Income Countries: A Systematic Review." *Injury*

Ludley, Alistair, Andrew Ting, Dean Malik, and Naveethan Sivanadarajah. 2023. "Observational analysis of documentation burden and data duplication in trauma patient pathways at a major trauma centre." *BMJ Open Quality* 12 (2)

Mahajan, Arnav, Andrew Tran, Esther S Tseng, John J Como, Kevin M El-Hayek, Prerna Ladha, and Vanessa P Ho. 2025. "Performance of trauma-trained large language models on surgical assessment questions: a new approach in resource identification." *Surgery* 179 , p. 108793

Moore, Lynne, and David E Clark. 2008. "The value of trauma registries." *Injury* 39 (6): 686–695

Nathens, Avery B, H Gill Cryer, and John Fildes. 2012. "The American College of Surgeons trauma quality improvement program." *Surgical Clinics* 92 (2): 441–454

Purcell, L. N. et al. 2020. "Barriers to and Facilitators of the Development of a Trauma Registry in a Low-Income Setting." *World Journal of Surgery* 44 : 4112–4119

Snell, Charlie et al. 2024. "Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters." *arXiv preprint arXiv:2408.03314*

Von Elm, Erik, Douglas G Altman, Matthias Egger et al. 2007. "The Strengthening the Reporting of Observational Studies in Epidemiology (STROBE) Statement: Guidelines for Reporting Observational Studies." *The Lancet* 370 (9596): 1453–1457