Rapport TER

Arame Diop
Valentino Cocks
Oscar Jimenez Flores
Yazmín Mendoza

May 30, 2025

Contents

Acknowledgments

We would like to express our heartfelt gratitude to Professor M. Huchard for graciously accepting to supervise this study, proposing the research topic and providing generous guidance and encouragement throughout the project. Our sincere thanks also go to Professor Y. Zhang, whose active participation in our meetings and insightful feedback helped us sharpen the focus and direction of our work. And last but not least, we express our sincere thanks to the team of researchers who helped us compute the concepts of the 150×150 example.

Summary

This report evaluates large language models' ability to generate code and directly compute formal concepts from binary contexts using various prompting strategies, highlighting their effectiveness, accuracy limitations, and computational efficiency compared to symbolic methods.

Introduction

Formal Concept Analysis (FCA, [GW99])) is a mathematical framework for extracting and structuring knowledge from binary data tables that record which objects possess which attributes. By treating every maximal set of objects (called the extent) sharing exactly the same attributes—and, dually, every maximal set of attributes (called the intent) shared by exactly those objects—as a formal concept, FCA builds a concept lattice whose nodes are concepts and whose ordering captures general-to-specific relationships. Computing these concepts is a high-intensity combinatorial task. In the context of our Travail Encadré de Recherche (TER) within the Master Intégration de Compétences (MICo) at Montpellier University, we explore the question of how effectively large language model (LLM)-generated programs can identify complete and correct intent–extent pairs and compute them directly (i.e., given a formal context file, generate the associated concepts). Beyond the theoretical structure of FCA, many practical applications and usage domains have been documented, particularly in the surveys by Poelmans et al. [P+13a, P+13b]. These works highlight how FCA has been applied in fields such as text and document analysis, software engineering and web usage mining. For instance, FCA has been used to explore user navigation patterns on websites to inform the design of recommendation systems. Common usage types include data exploration, rule extraction (e.g., implication rules between attributes) and clustering. These applications showcase FCA's capacity to transform raw binary relations into interpretable conceptual structures that support analysis, decision-making, and knowledge discovery.

These lattices can be used in several ways:

1. **Rule and implication mining** — Logical dependencies such as "if X and Y, then Z" correspond to lattice relations. FCA provides two complementary ways to obtain them: (a) automatic computation of a complete implicational basis by specialised algorithms [JKK21] (e.g. NextClosure, LinCbO, and other dependency–cover algorithms surveyed in Baixeries et al., 2023 [GO12]); (b) interactive acquisition via attribute exploration, where the algorithm queries an expert until enough implications (and counter-examples) have been gathered [GO12].

2. **Feature engineering & clustering for machine learning** — Concept intents provide high-level, interpretable features, while extents yield coherent clusters; FCA has been combined with classifiers, recommender systems and association-rule learners. [BV10] — shows how optimal Boolean matrix factors coincide with formal concepts, enabling FCA-based dimensionality reduction and interpretable feature construction used in downstream clustering/ML pipelines.

3. **Text and information retrieval** — Treating terms as attributes of documents lets concept lattices drive faceted search, query refinement and document recommendation. [CR20] — describes the GALOIS system, which builds concept lattices over

document–term data and exploits them for interactive browsing, query refinement and IR evaluation.

Our work aims to determine how well Large Language Models (LLMs) can identify concepts given a context in tabular form.

Finding all formal concepts in a data table is hard because the number of possible object-attribute combinations explodes—often into the millions or more—even for modest-sized datasets (our synthetic example of dimension 150x150 has more than 30 million concepts). To keep the task practical, researchers rely on smart, depth-first search routines such as Next-Closure [Gan10], Close-by-One [Kuz93] and In-Close [And11]. These algorithms walk through the context in a fixed order, use quick "is this really new?" checks to skip duplicates, and lean on bit-level tricks or parallel hardware to trim unnecessary work.

Formal Concept Analysis (FCA) belongs to the symbolic-AI tradition: its algorithms operate on discrete sets and deterministic inference rules, guaranteeing that every formal concept in a context is enumerated exactly once. Large Language Models (LLMs), by contrast, are statistical systems whose code-generation ability rests on probabilistic token prediction; the programs they emit therefore have no built-in correctness guarantees and may omit concepts, duplicate work, or fail altogether.

Our study proceeds in four stages. First, we select benchmark FCA contexts of varying size and run a state-of-the-art symbolic miner (fca4j [GL21]) to obtain the ground-truth concept sets and their execution costs. Second, we craft a spectrum of prompting techniques for several contemporary LLMs, each prompt requesting executable Java code that enumerates all formal concepts for a given binary context. Alongside the code-generation prompts, we introduce a second, "direct-concept" prompt family, that does not ask for a Java code to output the full list of ⟨extent, intent⟩ pairs for the given binary context, but gives the LLM the possibility to implement their own way in their response; they used then a variety of methods, from direct computing to generating and running Python code as the contexts became more complex. By comparing these raw concept lists to the ground-truth lattice, we can gauge the models' reasoning ability in isolation and distinguish conceptual errors from those that arise during code synthesis or execution. Third, we execute every generated program and measure its performance. We assess the quality of each LLM's concept set with recall, accuracy, and F1-score, three metrics long established in Information Retrieval. Recall measures the share of ground-truth concepts that the model recovers, accuracy adds the notion of correctly identifying concepts, and the F1-score is the harmonic mean of precision and recall, offering a single figure that balances completeness with correctness.

Across 50 benchmark contexts, the symbolic baseline (Fca4j) delivered perfect recall and F1 = 1.0 in every case while finishing in milliseconds to a few seconds; by contrast, LLM-generated miners showed sharp and inconsistent degradation as problem size grew. Chain-of-thought and "step-back" prompts never exceeded 30% F1 even on 9×9 toy tables and fell below 1% beyond 20×20, though they remained relatively quick (0.7–1 s). Few-shot and one-shot prompting occasionally reached 100% F1 on small synthetic contexts, but their runtime ballooned to tens of minutes or hours and collapsed to 0% F1 on most medium and real-world datasets. Role-based and self-consistency variants achieved perfect scores on a broader slice of synthetic inputs, yet still failed or exhausted memory on larger or denser cases, and in extreme instances needed days of wall-clock

time. Overall, LLM code generation can match symbolic accuracy only in narrow, low-complexity settings and at a much higher inference-plus-execution cost; for realistic FCA workloads, optimized symbolic algorithms remain decisively superior in both reliability and efficiency.

The "direct-concept" prompt family, without the constraint of generating Java code, proved to be suited only for very small contexts, composed by less than 15 objects and 15 attributes (15 x 15). Within that range, the results of the general models tested (DeepSeek-V3, Mistral Large) and one of the reasoning models (ChatGPT-o4) where heterogeneous, from F1 20% up to F1 100%, with no direct correlation to context complexity. For example, Mistral Large achieved an F1 score of 38.33% with contextual prompting on a 2×9 context, while reaching 96,67% with the same prompt on a larger size, 5 x 5. None of these general models consistently achieved 100% F1, not even with the smallest context size (2×3). In contrast, the other two reasoning models (Claude 4 Sonnet and Gemini 2.5 Pro) consistently performed better within this same size range, with two-thirds of their results exceeding 80%.

Beyond the 15 x 15 context size frontier, the performance of most of the LLM, no matter the type of prompt, dropped below 20% F1. For their part, Claude 4 Sonnet and Gemini 2.5 showed remarkable exceptions, achieving near-perfect performance with 15×15 contexts using the few-shot (99.84%) and self-consistency (100%) prompts for Gemini, and few-shot (99.84%), role prompting (100%), step-back prompting (99.79%), chain-of-thought (99.84%), and self-consistency (100%) for Claude 4 Sonnet. Nevertheless, we would like to highlight two points. First, in most of these high-performing cases, the LLM generated a coded tool in Python (never suggested in the prompt), but none of these code-based solutions achieved 100% accuracy in every application during testing. And second, at this point, with the size 15 x 15, the low levels of accuracy show that the LLM generated too many wrong concepts, or even created their own objects and attributes. For example, with the prompt Chain of thought, Gemini switched the objects from O1, O2, O3 and so on (the originals), to Patient 1, Patient 2 and the subsequent ones, and changed the original attributes Attr1, Attr2..., to Cough, Fatigue, Fever, Headache and Sore Throat.

Chapter 1 details our methodology. Finally, we present a summary of our results in Chapter 2. In chapter 3 we comment on our results and finally in Chapter 4 we give an overview of possible future work to enhance the precision of our results. Appendix A contains our prompts and the raw data of our results.

1. Methodology

Our goal was to evaluate how effectively Large Language Models (LLMs) can generate programs that compute formal concepts from a given context, and how effectively they can find in themselves by providing them with a csv file representing a context.

In addition to probing the models' coding ability, we run a second track that lets the LLM to choose the way to find the concepts: after supplying the same CSV-encoded context, we simply ask each large language model to list every ⟨extent, intent⟩ pair in its reply. Since these "direct-concept" don't ask for external computation, the methods used by the LLM vary from combinatorial search within its own reasoning window (with very small contexts, like 2 x 3) to the generation and application of some small, not fully efficient, coded Python tools.

We thus adopted two complementary evaluation tracks—(i) asking the LLM to write a program that mines concepts, and (ii) asking the same model to enumerate the concepts directly after reading a CSV-encoded context—because they probe different cognitive capacities of statistical models. The code-generation task tests algorithmic synthesis: can an LLM translate its latent knowledge of depth-first search, canonicity checks and pruning heuristics into runnable code that achieves symbolic-level completeness once executed? The direct-enumeration task, by contrast, measures the model's in-context reasoning and working-memory limits: can it internally carry out the combinatorial search within its token window without external computation? Our prompts are available in Annex A.

1.1. Tested Large Language Models

We tested the following LLMs, all in their respective April 2025 versions.

- OpenAI's ChatGPT o4-mini-high;
- Google's Gemini 2.5 Pro;
- Anthropic's Claude 3.7 Sonnet for prompts requesting executable Java code;
- Anthropic's Claude 4 Sonnet for prompts not requesting for code;
- Mistral AI's Mistral Large;
- Deepseek Inc.'s DeepSeek-V3.

OpenAI's ChatGPT o4-mini-high, Anthropic's Claude and Gemini 2.5 Pro are reas oning models. A reasoning model is one that deliberately spends extra computation steps on an internal chain-of-thought before writing the final answer. The o-series models (o1, o3, o4-mini, . . .) fall in this category: they break problems into sub-steps, simulate plans, and iterate over possible solutions, which makes them stronger at multi-hop tasks such

as coding, math and scientific reasoning, albeit a bit slower and costlier than "searching" or purely retrieval-oriented models [1].

1.2. Prompt engineering

LLMs do not expose an API specialized for Formal Concept Analysis, hence we relied on prompt engineering as specified in the document Prompting Engineering Techniques by Lee Bonstra[2]. Each prompt was crafted to conform to one of the following techniques (see Annex A for full prompt texts):

- Zero-shot;
- One-shot and few-shot;
- Role prompting;
- Contextual prompting;
- Step-back;
- Chain of Thought (CoT);
- Self-consistency;
- Tree of Thoughts.

All our prompts can been seen in our annex, these include Zero-shot that simply gives the model an instruction ("list all formal concepts in this table") and relies on its latent knowledge to respond. One-shot and few-shot prepend one or a handful of worked examples so the LLM can infer the desired format and reasoning pattern by analogy. Role prompting involves assigning a specific role to the LLM, which lets it generate more relevant and informative output according to that role. Contextual prompting refers to a technique where the model is given rich, structured background information before being asked to perform a task. Step-back prompts ask the model first to restate the task or identify sub-problems ("what do I need to compute before listing the concepts"), encouraging a brief plan before execution. Chain-of-Thought (CoT) makes that planning explicit and continuous: the model is instructed to think "step by step", generating a visible sequence of intermediate deductions that can improve accuracy on multi-step problems. Self-consistency takes CoT further by sampling many independent chains and then selecting the most common or best-scoring answer. Tree of Thoughts generalises this idea into an explicit search tree: multiple partial reasoning branches are expanded in parallel, heuristically scored, and pruned, much like beam search—so the LLM explores diverse lines of thought while keeping the combinatorial explosion in check. The first family of prompts required an implementation in Java, and after running each LLM's outputs we tested them on our test set made of synthetic and real-life examples. For the second family of prompts, we directly evaluated the LLM's output.

[1] See OpenAI's announcement "Introducing OpenAI o1-preview" https://openai.com/index/introducing-openai-o1-preview/, which describes these models as "designed to spend more time thinking before they respond.

[2] https://www.kaggle.com/whitepaper-prompt-engineering

1.3. Generation environment

In the case of generated code, it was immediately copied into the **Eclipse IDE 2024-12** workspace, which made it easy to pinpoint compilation errors. We did not feed correction prompts back to the model in case of errors since these were memory-related only. All produced code compiled on the first try and was syntax-error free.

1.4. Test data

Section 2.4.1 presents the synthetic data we used for our test, while Section 2.4.2 deals with the real-life dataset.

1.4.1. Synthetic contexts

We produced a suite of random binary context matrices (CSV files) of increasing size (from 2×3 to 50×50) using a Java program. Each CSV file entry was independently set to 1 with probability $p = 0.5$. For every generated context we pre–computed the full concept lattice using the `fca4j` CLI; these lattices serve as ground-truth.

The set can be download with the following link: https://github.com/OskrJF/TER-M1—Code-and-Test-sets.git

1.4.2. Real-world contexts

To assess practical usefulness, we selected the following standard FCA benchmarks:

- **fca4j-benchmark-master-datasets**.

This FCA benchmark was assembled by researcher Alain Gutierrez by translating into CSV a selection of datasets from the UCI Machine Learning Repository [DG19], and binarizing them accordingly.

The UCI Machine Learning Repository is a well-known source of real-world datasets used extensively for empirical evaluation in machine learning and data analysis.

1.5. Evaluation

All the datasets and code can be found in the same repository as given in 2.4.1.

Our aim was to store fca4j's output result into a data structure that made it easier to compare it to the LLM's program's output. We remained flexible in this approach since we did not give any constraints to the LLMs as to what data structures to use, allowing them freedom in how they compute the concepts.

First, we wrote a Java program that parses fca4j's output (its standard output is a DOT file) and copies its concepts into a data structure compatible with each LLM's output.

Second, we ran each LLM's program and modified their code to store their final result in a structure of the same type as fca4j's.

We did not modify the LLM's code regarding how they computed the concepts; their final result was simply converted to the easiest data structure we could work with. For example, some LLMs chose to represent the concepts as a predefined class, while others simply as `ArrayList<String>`. In the case of the "direct-concept" prompt family, we also first applied the prompt to ask the LLM for the concepts, and after that we asked them to give to their output the format we needed to compare it against the fca4j's output. We decided to leave this request of format outside of the original prompts because we observed that by including it from the beginning the performance dropped too much.

We compared:

- the set of concepts returned (exact match required for full success),
- runtime (wall-clock time in Eclipse IDE on an Apple M1 2021 8GB RAM and Apple M3 16 GB RAM).

We also calculated their F1, accuracy, precision, and recall measures. These calculations were performed by the comparator itself.

We have defined these metrics as follows:

1. precision (correctness): Measures how many of the predicted concepts are correct. Defined as:
$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$
2. recall (completeness): Measures how many of the true concepts were found. Defined as:
$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$
3. F1 measure: Harmonic mean of precision and recall. Defined as:
$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where True Positives are concepts found by the LLM, that are also part of the ground truth, False Positives are subset pairs that the LLM claims to be concepts but that are not concepts (i.e, they are not present in the ground truth) and False Negatives are concepts in the ground truth but not in the LLM's output.

1.6. Results interpretation

A generated program was deemed correct if, for every context, the multiset of (extent, intent) pairs matched those produced by the specialised software. Performance was considered acceptable when runtime was within a factor of 10 of `fca4j` on the same datasets; no hard threshold was imposed on peak memory. In the case of the direct request, since it showed to perform with small contexts only, execution time was always under one minute, therefore, we did not consider this factor to evaluate the performance of the LLM.

2. Results

In this section we present the results we gathered during our experiments while testing every prompting technique, whether asking for generated code or asking the direct computation of concepts.

2.1. Code

This section presents the results of our first approach, that is, asking the LLMs to generate concept-mining code.

A total of 22 synthetic examples and a total of 19 real examples (as described in 2.4.2) were used for the tests.

2.1.1. ChatGPT o3-mini-high

Chain Of Thought - Oscar

The Chain Of Thought prompting technique yielded code that was unable to find any of the examples provided, whether real or synthetic. That is, it achieved a 0% mark on precision, recall and accuracy.

Figure B.1 in annex B shows the details of our measurements. Examples winequality-red_binarized, soybean_binarized and cmc_binarized failed with an "index out of bounds error".

The average running times for the synthetic examples are shown in table 3.1, with the more precise details given in annex B1.

Average Time - Synthetic - CoT

	LLM average time	Fca4j average time
Average	7s 474ms	5s 254ms

Figure 2.1.

The average running times for the real examples are shown in table 3.2, with the more precise details given in annex B2.

Average Time - Real - CoT

	LLM average time	Fca4j average time
Average	13s 311ms	10s 735ms

Figure 2.2.

One-Shot - Oscar

The One-Shot prompting technique yielded more varied results, achieving a 100% score across F1, recall and accuracy for 9 tests out of 22 synthetic tests, and 5 out of 19 real tests, figure 3.3 summarizes this. The success ratio was computed by dividing the number of passed tests by the number of total tests.

Test results - One Shot

Test type	Passed	Not passed	Success Ratio
Synthetic	9	13	40,91 %
Real	5	14	26,32 %

Figure 2.3.

The average running times for exhibit a much longer duration than that of specialized software fca4j. Across all prompting techniques, when ChatGPT-o3-mini-high achieved 100% scores across all metrics, it necessarily took more time to achieve this. table 3.4 summarizes the running times for the synthetic examples and table 3.5 for the real examples. Details for running times are given in figures B.3 and B.4.

Average Time - Synthetic - One Shot

	LLM average time	Fca4j average time
Average	13m 20s 23ms	5s 254ms

Figure 2.4.: Synthetic example runtimes

Average Time - Real - One Shot

	LLM average time	Fca4j average time
Average	14m 45s 624ms	10s 619ms

Figure 2.5.: Real example runtimes

Few-Shot - Oscar

The Few Shot prompting technique yielded a success ratio of 40.91% for synthetic results and success ratio of 21.05% for the real examples. As depicted in Figure 3.6. Details are given in annex B.11 and B.12.

Test results - Few Shot

Test type	Passed	Not passed	Success Ratio
Synthetic	9	13	40,91 %
Real	4	15	21,05 %

Figure 2.6.

The Few Shot prompting technique exhibited 53.94% slower times for the synthetic examples and 21.05% slower times for the real examples. Details are given in annex B.13.

Average Time - Synthetic - Few Shot

	LLM average time	Fca4j average time
Average	47m 16s 717ms	5s 254ms

Average Time - Real - Few Shot

	LLM average time	Fca4j average time
Average	38m 13s 679ms	10s 619ms

Figure 2.7.

Role - Oscar

The Role prompting technique yielded near perfect results, scoring 100% on all metrics across nearly all examples besides: bank_binarized where after 2 days of computation we stopped the program and soybean_binarized where after 4 hours of running time a Java heap space error abruptly stopped the execution of the program. Details are given in table B.8 and B.9.

The Role prompt exhibited the second best running time performance for the synthetic examples while maintaining accuracy and recall.

As for the average running times for the real examples, we observed a gigantic discrepancy, with fca4j being 99.9%.

Details are given in annex B.10.

Average Time - Synthetic - Role

	LLM average time	Fca4j average time
Average	1m 29s 895ms	5s 254ms

Average Time - Real - Role

	LLM average time	Fca4j average time
Average	2h 55m 48s 684ms	10s 619ms

Figure 2.8.

Self-Consistency - Oscar

The Self-Consistency prompting technique had a success ratio of 100% for the synthetic examples 78.95% success ratio for the real ones. Details are given in annex B.14 and B.15.

Test results - Self Consistency

Test type	Passed	Not passed	Success Ratio
Synthetic	22	0	100 %
Real	15	4	78,95 %

Figure 2.9.

The average running times are as follows: 211.2% slower for the synthetic tests and 37% slower for the real tests. Details are given in annex B.16.

Average Time - Synthetic - SF

	LLM average time	Fca4j average time
Average	16s 348ms	5s 254ms

Average Time - Real - SF

	LLM average time	Fca4j average time
Average	1h 5m 49s 805ms	10s 619ms

Figure 2.10.

Step-Back - Oscar

The step-Back prompting technique yielded 0% scores across all datasets.

Test results - Step Back

Test type	Passed	Not passed	Success Ratio
Synthetic	0	22	0,00 %
Real	0	19	0,00 %

Figure 2.11.

The average running times were closer to those of fca4j, but we found this not to be meaningful because of its poor results across all metrics. Details are given in annex.

Average Time - Synthetic - Step Back

	LLM average time	Fca4j average time
Average	7s 219ms	5s 254ms

Average Time - Real - Step Back

	LLM average time	Fca4j average time
Average	22s 468ms	10s 619ms

Figure 2.12.

2.1.2. Claude3.7

Zero-Shot - Valentino

The Zero-Shot prompting technique yielded a success ratio of 21.74% on the synthetic examples and 33.33% on the real examples (Figure B.19); The details are given in the annex at B.2.1.

Test results – Zero-Shot

Test type	Passed	Not passed	Success Ratio
Synthetic	5	18	21.74 %
Real	5	10	33.33 %

Figure 2.13.

One-Shot - Valentino

The One-Shot prompting technique yielded a success ratio of 21.74% on the synthetic examples and 35.00% on the real examples (Figure B.20); The details are given in the annex at B.2.2.

Test results – One-Shot

Test type	Passed	Not passed	Success Ratio
Synthetic	5	18	21.74 %
Real	7	13	35.00 %

Figure 2.14.

Few-Shot - Valentino

The Few-Shot prompting technique yielded a success ratio of 95.65% on the synthetic examples and 75.00% on the real examples (Figure B.21); The details are given in the annex at B.2.3.

Test results – Few-Shot

Test type	Passed	Not passed	Success Ratio
Synthetic	22	1	95.65 %
Real	12	4	75.00 %

Figure 2.15.

Chain-of-Thought - Valentino

The Chain-of-Thought prompting technique yielded a success ratio of 100.00% on the synthetic examples and 20.00% on the real examples (Figure B.22); The details are given in the annex at B.2.4.

Test results – Chain-of-Thought

Test type	Passed	Not passed	Success Ratio
Synthetic	23	0	100.00 %
Real	4	16	20.00 %

Figure 2.16.

Tree-of-Thoughts - Valentino

The Tree-of-Thoughts prompting technique yielded a success ratio of 100.00% on the synthetic examples and 70.59% on the real examples (Figure 2.17); The details are given in the annex at B.2.4.

Test results – Tree-of-Thoughts

Test type	Passed	Not passed	Success Ratio
Synthetic	23	0	100.00 %
Real	12	5	70.59 %

Figure 2.17.

Self-Consistency - Valentino

The Self-Consistency prompting technique yielded a success ratio of 0.00% on the synthetic examples and 65.00% on the real examples (Figure B.24); The details are given in the annex at B.2.6.

Test results – Self-Consistency

Test type	Passed	Not passed	Success Ratio
Synthetic	0	13	0.00 %
Real	13	7	65.00 %

Figure 2.18.

Role Prompting - Valentino

The Role prompting technique yielded a success ratio of 100.00% on the synthetic examples and 20.00% on the real examples (Figure **??**); The details are given in the annex.

Test results – Role Prompting

Test type	Passed	Not passed	Success Ratio
Synthetic	23	0	100.00 %
Real	4	16	20.00 %

Figure 2.19.

Step-back Prompting - Valentino

The Step-back prompting technique yielded a success ratio of 0.00% on both the synthetic and real examples (Figure **??**; The details are given in annex at B.2.7.

Test results – Step-back Prompting

Test type	Passed	Not passed	Success Ratio
Synthetic	0	18	0.00 %
Real	1	18	5.26 %

Figure 2.20.

2.1.3. Deepseek V3

Zero-Shot - Valentino

The Zero-Shot prompting technique yielded a success ratio of 0.00% on the synthetic examples and 15.79% on the real examples (Figure B.26); The details are given in annex

at B.3.1.

Test_results___Zero-Shot

	Passed	Not passed	Success Ratio
Synthetic	0	23	0.00 %
Real	3	16	15.79 %

Figure 2.21.

One-Shot - Valentino

The One-Shot prompting technique yielded a success ratio of 13.04% on the synthetic examples and 20.00% on the real examples (Figure B.27); The details are given in annex at B.3.2.

Test_results___One-Shot

	Passed	Not passed	Success Ratio
Synthetic	3	20	13.04 %
Real	4	16	20.00 %

Figure 2.22.

Few-Shot - Valentino

The Few-Shot prompting technique yielded a success ratio of 11.11% on the synthetic examples and 5.26% on the real examples (Figure B.28); The details are given in annex at B.3.3.

Test_results___Few-Shot

	Passed	Not passed	Success Ratio
Synthetic	2	16	11.11 %
Real	1	18	5.26 %

Figure 2.23.

Chain-of-Thought - Valentino

The Chain-of-Thought prompting technique yielded a success ratio of 86.96% on the synthetic examples and 0.00% on the real examples (Figure B.29); The details are given in annex at B.3.4.

Test_results___ChainOfThought

	Passed	Not passed	Success Ratio
Synthetic	20	3	86.96 %
Real	0	5	0.00 %

Figure 2.24.

Tree-of-Thoughts - Valentino

The Tree-of-Thoughts prompting technique yielded a success ratio of 0.00% on the synthetic examples and 21.05% on the real examples (Figure B.30); The details are given in annex at B.3.5.

Test_results___TreeOfThought

	Passed	Not passed	Success Ratio
Synthetic	0	23	0.00 %
Real	4	15	21.05 %

Figure 2.25.

Self-Consistency - Valentino

The Self-Consistency prompting technique yielded a success ratio of 0.00% on the synthetic examples and 15.79% on the real examples (Figure B.32); The details are given in annex at B.3.7.

Test_results___Self-Consistency

	Passed	Not passed	Success Ratio
Synthetic	0	23	0.00 %
Real	3	16	15.79 %

Figure 2.26.

Role Prompting - Valentino

The Role Prompting technique yielded a success ratio of 86.96% on the synthetic examples and 36.84% on the real examples (Figure B.31); The details are given in annex at B.3.6.

Test_results___Role_Prompting

	Passed	Not passed	Success Ratio
Synthetic	20	3	86.96 %
Real	7	12	36.84 %

Figure 2.27.

Step-back Prompting - Valentino

The Step-back Prompting technique yielded a success ratio of 0.00% on the synthetic examples and 5.26% on the real examples (Figure B.32); The details are given in annex at B.3.8.

stepback_prompting_summary

Test type	Passed	Not passed	Success Ratio
Synthetic	0	18	0.00 %
Real	1	18	5.26 %

Figure 2.28.

2.2. Direct concept query

This section presents the results of our second approach, that is, that of asking the LLMs to directly enumerate the concepts.

Zero-shot - Arame and Yazmín's

ChatGPT-4o: The model performs well on some files but shows significant inconsistency, especially with complex examples like `eg15_15.csv`. DeepSeek-V3: DeepSeek shows good precision and recall on most tasks, with only slight drops in harder examples. Claude 4 Sonnet: Claude demonstrates solid and steady performance overall, although accuracy varies depending on the complexity of the input. Mistral Large: Results are generally weak, with the model struggling especially on the more challenging files. Gemini 2.5 Pro: Gemini handles simple tasks effectively but struggles to generalize as difficulty increases.

Zero shot

Zero shot - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	64,20	66,67	47,98	63,83
eg2_9.csv	19,45	50,00	16,19	27,78
eg5_5.csv	67,50	93,75	66,67	75,00
eg8_8.csv	100,00	100,00	100,00	100,00
eg15_15.csv	5,79	2,89	1,92	3,77

Zero shot - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	93,33	100,00	93,33	96,30
eg2_9.csv	50,95	83,33	45,83	62,63
eg5_5.csv	100,00	93,75	93,75	96,67
eg8_8.csv	82,86	66,67	57,56	72,40
eg15_15.csv	27,48	9,62	7,20	13,42

Zero shot - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	86,67	91,67	78,33	87,83
eg2_9.csv	93,33	100,00	93,33	96,30
eg5_5.csv	60,99	87,50	56,09	71,84
eg8_8.csv	63,89	81,25	61,91	69,23
eg15_15.csv	55,78	61,22	54,13	57,63

Zero shot - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	80,00	100,00	80,00	88,89
eg2_9.csv	57,14	100,00	57,14	72,73
eg5_5.csv	54,15	81,25	44,28	57,95
eg8_8.csv	60,94	93,75	58,77	73,43
eg15_15.csv	0,16	0,64	0,13	0,25

Zero shot - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	66,67	66,67	51,11	66,67
eg2_9.csv	70,00	75,00	56,67	72,22
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	100,00	91,67	91,67	95,46
eg15_15.csv	44,00	7,05	6,47	12,15

Figure 2.29.

Few-shot - Arame and Yazmín's

ChatGPT-4o: The model benefits clearly from few-shot prompting, with strong improvements on most examples. DeepSeek-V3: Few-shot learning helps maintain high performance, though results vary more on challenging cases. Claude 4 Sonnet: Claude handles this setup quite well, especially on simpler examples, but has issues with consistency on the harder files. Mistral Large: Performance improves slightly, but results remain relatively low and unstable. Gemini 2.5 Pro: The model shows near-perfect performance on almost all examples, including the hardest one.

Few shot

Few shot - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	91,67	91,67	86,67	91,67
eg5_5.csv	75,00	93,75	71,82	83,34
eg8_8.csv	73,69	68,75	63,24	70,93
eg15_15.csv	50,00	5,45	5,45	9,83

Few shot - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	96,33	100,00	96,00	96,00
eg5_5.csv	63,64	68,75	59,38	65,79
eg8_8.csv	59,48	58,33	40,91	57,48
eg15_15.csv	11,81	1,60	1,43	2,82

Few shot - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	57,14	100,00	57,14	72,73
eg2_9.csv	90,00	100,00	90,00	94,45
eg5_5.csv	86,37	93,75	80,12	88,77
eg8_8.csv	60,00	58,34	55,00	59,09
eg15_15.csv	100,00	99,68	99,68	99,84

Few shot - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	80,00	100,00	80,00	88,89
eg2_9.csv	62,50	62,50	57,15	62,50
eg5_5.csv	54,62	62,50	42,86	56,00
eg8_8.csv	46,67	29,17	22,00	35,90
eg15_15.csv	40,82	14,10	11,72	20,97

Few shot - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	100,00	100,00	100,00	100,00
eg15_15.csv	100,00	99,68	99,68	400,00

Figure 2.30.

Chain of thought - Arame et Yazmín's

ChatGPT-4o: The model achieves perfect results on simpler files but loses effectiveness on more complex ones. DeepSeek-V3: DeepSeek maintains high scores across the board, although performance slightly dips on `eg8_8.csv` and `eg15_15.csv`. Claude 4 Sonnet: Results are strong overall, with clear gains in reasoning, though `eg5_5.csv` and `eg15_15.csv` remain weak points. Mistral Large: The model shows clear improvement on several files, although its performance on the most complex examples remains low. Gemini 2.5 Pro: Once again, Gemini shows consistently high results, with almost no degradation across difficulty levels.

Chain of thought

Chain of thought - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	75,00	75,00	60,00	75,00
eg8_8.csv	77,78	29,17	26,92	42,42
eg15_15.csv	3,92	2,14	1,41	2,77

Chain of thought - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	93,33	100,00	100,00	96,33
eg5_5.csv	96,63	100,00	100,00	99,00
eg8_8.csv	78,34	41,67	37,97	53,93
eg15_15.csv	3,76	2,94	1,91	3,03

Chain of thought - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	87,50	87,50	80,00	87,50
eg5_5.csv	59,52	50,00	41,67	54,29
eg8_8.csv	86,84	79,17	74,14	82,56
eg15_15.csv	100,00	99,68	99,68	99,84

Chain of thought - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	75,00	75,00	60,00	75,00
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	91,67	91,67	84,62	91,67
eg15_15.csv	7,79	13,79	5,19	9,45

Chain of thought - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	57,36	43,75	33,04	49,56
eg15_15.csv	50,00	50,00	50,00	50,00

Figure 2.31.

Contextual prompting - Arame et Yazmín's

ChatGPT-4o: The model improves slightly in context-rich examples but continues to struggle with generalization on complex files. DeepSeek-V3: DeepSeek delivers reliable performance with this prompt, particularly on the easier and medium examples. Claude 4 Sonnet: Claude shows consistent precision and recall, although results fluctuate depending on file complexity. Mistral Large: There is slight progress compared to previous prompts, but the model still underperforms on harder cases. Gemini 2.5 Pro: Performance is surprisingly low here for Gemini, especially on `eg2_3.csv` and `eg15_15.csv`.

Contextual prompting

Contextual prompting - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	80,56	91,67	75,56	85,00
eg2_9.csv	93,33	100,00	93,33	96,30
eg5_5.csv	57,39	100,00	57,39	70,25
eg8_8.csv	21,23	22,92	13,54	22,04
eg15_15.csv	27,67	8,55	6,55	12,29

Contextual prompting - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	100,00	93,75	93,75	96,67
eg8_8.csv	82,50	54,17	48,06	64,68
eg15_15.csv	50,22	11,54	10,08	18,28

Contextual prompting - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	83,33	83,33	77,78	83,33
eg5_5.csv	58,18	75,00	47,73	64,59
eg8_8.csv	70,00	66,67	61,11	68,18
eg15_15.csv	28,81	15,60	9,99	17,77

Contextual prompting - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	60,71	91,67	55,71	71,47
eg2_9.csv	25,00	83,33	23,72	38,33
eg5_5.csv	100,00	93,75	93,75	96,67
eg8_8.csv	52,36	58,33	49,92	52,78
eg15_15.csv	34,60	4,91	3,42	6,49

Contextual prompting - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	50,00	41,67	40,00	44,44
eg2_9.csv	50,00	25,00	20,00	33,33
eg5_5.csv	85,00	93,75	81,82	88,89
eg8_8.csv	47,50	39,59	35,78	43,19
eg15_15.csv	100,00	67,09	67,09	67,51

Figure 2.32.

Role prompting - Arame and Yazmín's

ChatGPT-4o: The model shows some gains on early examples but falls short on harder files like `eg8_8.csv` and `eg15_15.csv`. DeepSeek-V3: DeepSeek is highly consistent and performs strongly across all levels of difficulty. Claude 4 Sonnet: The model maintains strong results throughout, including very solid handling of difficult files. Mistral Large: Despite some improvement on a few files, the model remains inconsistent and struggles with complexity. Gemini 2.5 Pro: Gemini continues to shine, delivering perfect scores in most examples and staying robust even under challenging conditions.

Role prompting

Role prompting - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	57,14	100,00	57,14	72,73
eg2_9.csv	93,63	100,00	93,63	93,33
eg5_5.csv	62,50	93,75	60,26	75,00
eg8_8.csv	46,67	29,17	26,32	35,90
eg15_15.csv	26,11	6,80	9,83	17,90

Role prompting - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100
eg2_9.csv	96,33	100,00	96,33	100
eg5_5.csv	96,63	100,00	96,63	96,33
eg8_8.csv	63,78	79,00	63,78	80,88
eg15_15.csv	5,88	3,21	2,16	4,89

Role prompting - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	75,00	75,00	60,00	75,00
eg5_5.csv	93,75	93,75	88,89	93,75
eg8_8.csv	59,33	33,34	27,03	42,00
eg15_15.csv	100,00	100,00	100,00	100,00

Role prompting - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	75,00	75,00	85,71
eg2_9.csv	33,33	25,00	16,67	28,57
eg5_5.csv	70,72	75,00	56,82	72,23
eg8_8.csv	12,30	47,92	10,85	19,57
eg15_15.csv	0,87	3,21	0,70	1,37

Role prompting - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	97,06	75,00	73,67	84,48
eg15_15.csv	80,31	7,05	6,56	11,85

Figure 2.33.

Self consistency - Arame and Yazmín's

ChatGPT-4o: The model performs very well on most examples, though it still fails to handle `eg15_15.csv` reliably. DeepSeek-V3: This prompt stabilizes DeepSeek's performance, keeping metrics high even on moderately hard tasks. Claude 4 Sonnet: Claude delivers consistent results, especially on complex reasoning cases. Mistral Large: The model achieves top scores on basic and mid-level files, but performance drops again on the last one. Gemini 2.5 Pro: Gemini once again shows excellent results, with near-perfect consistency across the dataset.

Self consistency

Self consistency - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	66,67	100,00	66,67	80,00
eg8_8.csv	100,00	95,83	95,83	97,87
eg15_15.csv	20,00	1,28	1,22	2,41

Self consistency - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	94,74	75,00	72,00	83,72
eg15_15.csv	60,00	1,92	1,90	3,73

Self consistency - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	80,00	100,00	80,00	88,89
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	100,00	100,00	100,00	100,00
eg15_15.csv	100,00	100,00	100,00	100,00

Self consistency - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	80,00	100,00	80,00	88,89
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	96,00	100,00	96,00	97,96
eg15_15.csv	34,15	8,97	7,65	14,21

Self consistency - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	100,00	100,00	100,00	100,00
eg15_15.csv	100,00	100,00	100,00	100,00

Figure 2.34.

Step-back prompting - Arame and Yazmín's

ChatGPT-4o: The prompt helps moderately on easier cases, but fails to boost performance on complex examples. DeepSeek-V3: DeepSeek delivers solid performance on early files, though the prompt doesn't fully compensate for difficult inputs. Claude 4 Sonnet: Claude demonstrates excellent handling of both easy and complex files, with steady improvement. Mistral Large: Overall performance is weak, though a few files show slight progress over previous prompts. Gemini 2.5 Pro: Gemini continues to dominate, showing excellent results on all examples.

Step-back prompting

Step-back prompting - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	83,33	83,33	73,33	83,33
eg2_9.csv	91,67	91,67	86,67	91,67
eg5_5.csv	69,40	87,50	62,22	73,61
eg8_8.csv	63,75	50,00	38,46	55,41
eg15_15.csv	45,01	11,97	8,77	16,07

Step-back prompting - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	93,33	75,00	68,33	76,30
eg5_5.csv	67,33	91,67	61,05	73,20
eg8_8.csv	86,44	58,34	54,17	68,66
eg15_15.csv	24,80	8,97	6,68	12,29

Step-back prompting - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	88,89	100,00	88,89	93,33
eg2_9.csv	41,67	50,00	40,00	44,44
eg5_5.csv	95,83	91,67	88,43	93,61
eg8_8.csv	100,00	98,61	98,61	99,29
eg15_15.csv	100,00	99,57	99,57	99,79

Step-back prompting - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	66,67	83,33	56,67	72,22
eg2_9.csv	38,96	41,67	24,52	35,86
eg5_5.csv	42,70	33,33	23,45	34,21
eg8_8.csv	90,00	47,22	46,14	55,87
eg15_15.csv	58,94	1,49	1,32	2,59

Step-back prompting - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	70,37	70,83	68,75	70,59
eg8_8.csv	100,00	79,17	79,17	79,17
eg15_15.csv	86,22	10,26	8,70	14,39

Figure 2.35.

Tree of thoughts - Arame and Yazmín's

ChatGPT-4o: The model improves marginally on some files, but its performance on complex data remains limited. DeepSeek-V3: DeepSeek shows good reasoning with this strategy, although accuracy varies across the dataset. Claude 4 Sonnet: Claude performs very well on some inputs but lacks consistency on more ambiguous examples. Mistral Large: There is a lack of robustness in handling complexity, even with this advanced prompting technique. Gemini 2.5 Pro: Gemini stands out once again, with perfect or near-perfect scores, showcasing strong multi-step reasoning.

Tree of thoughts

Tree of thoughts - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	83,33	75,00	65,00	78,57
eg2_9.csv	90,00	100,00	90,00	94,45
eg5_5.csv	58,33	87,50	53,85	70,00
eg8_8.csv	48,28	58,33	35,90	52,83
eg15_15.csv	11,11	1,28	1,16	2,30

Tree of thoughts - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	86,67	100,00	86,67	92,59
eg2_9.csv	90,00	100,00	90,00	94,45
eg5_5.csv	72,73	100,00	72,73	84,21
eg8_8.csv	82,35	58,33	51,85	68,29
eg15_15.csv	65,52	12,18	11,45	20,54

Tree of thoughts - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	75,00	75,00	60,00	75,00
eg2_9.csv	100,00	87,50	87,50	92,86
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	29,79	58,33	24,56	39,44
eg15_15.csv	100,00	0,64	0,64	1,27

Tree of thoughts - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	77,50	87,50	70,00	81,95
eg2_9.csv	40,00	50,00	40,00	44,45
eg5_5.csv	42,86	37,50	25,00	40,00
eg8_8.csv	15,79	12,50	7,50	13,95
eg15_15.csv	0,93	0,64	0,38	0,76

Tree of thoughts - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100,00	100,00	100,00	100,00
eg2_9.csv	100,00	100,00	100,00	100,00
eg5_5.csv	100,00	100,00	100,00	100,00
eg8_8.csv	100,00	100,00	100,00	100,00
eg15_15.csv	11,11	1,28	1,16	2,30

Figure 2.36.

3. Discussion

The empirical results in Chapter **??** reveal marked performance disparities among models, prompting strategies, and task types. In this chapter, we interpret these outcomes in light of model design, prompt formulation, and the nature of Formal Concept Analysis (FCA) as a task. Rather than exhaustively reiterating numeric scores, we aim to distill broader lessons from our findings and situate them within the broader capabilities and limitations of current Large Language Models (LLMs).

3.1. Prompting as a cognitive scaffold

Prompting strategies played a decisive role in shaping LLM behavior. Techniques like Chain-of-Thought (CoT) and Self-Consistency consistently outperformed Zero-shot or Step-back variants, especially in the code generation track. This shows that structured prompting encourages models to simulate planning and decompositional reasoning, which is critical in generating correct algorithms for symbolic problems like FCA.

However, the effect of prompting was muted in the *direct enumeration* task. Even when prompted with explicit reasoning scaffolds (e.g., CoT or Tree of Thoughts), models struggled to faithfully enumerate formal concepts. This suggests that prompting can scaffold explicit computation only up to the model's working-memory and attention limits. Beyond this window, accuracy degrades, regardless of prompt quality.

3.2. Sampling temperature and correctness

We did not systematically vary sampling temperature, but it likely influenced results in subtle ways. Higher temperatures increase diversity in model outputs, which is useful for creative tasks or stochastic search (e.g., Tree of Thoughts), but may harm determinism and correctness in formal tasks like concept enumeration. Conversely, low temperature settings reduce variance but can trap the model in local optima—especially when a poor chain of reasoning dominates. In future work, a comparative study of temperature's effect on stability and correctness would be valuable, particularly for direct reasoning prompts.

3.3. Why code outperforms direct reasoning

Across all models, code generation proved more successful than direct concept enumeration. This outcome stems from fundamental architectural constraints. In the code track,

the LLM must only generate an abstract algorithm; the actual combinatorial work is offloaded to the CPU. In contrast, direct concept listing forces the model to carry out the search internally within its finite token and attention budget. This overloads its context window, especially for larger contexts, and leads to degraded performance. Additionally, the discrete nature of formal concepts—requiring exact extent/intent matches—leaves little room for partial credit or plausible approximation.

3.4. Model specialization and performance

Our benchmark spanned general-purpose reasoning models (ChatGPT, Claude, Gemini) and one code-specialized model (Mistral). The latter category, while optimized for syntactic correctness and pattern completion in code, did not necessarily excel at symbolic correctness or logic-heavy reasoning. This suggests that code models are not inherently better at formal tasks—they are better at *formatting* valid code. For FCA, success depends not just on well-formed syntax but on deep understanding of concept closure, which is rarely encountered in training corpora.

More generally, multimodal models (e.g., those trained on both text and images) were not evaluated in this round, but their relevance to FCA is likely limited unless future prompts involve diagrammatic lattices or graphical contexts.

3.5. On the efficacy of prompting techniques

Certain prompting strategies, such as Self-Consistency and Role prompting, emerged as particularly effective. Self-Consistency leverages model variance by sampling multiple reasoning chains and voting among them—a technique well-suited to noisy problems. Role prompting, which frames the model as a domain expert or software engineer, seems to activate latent problem-solving behavior and domain-specific heuristics. By contrast, Step-back prompting underperformed, likely because it failed to provide enough scaffolding for multi-step synthesis, particularly when the model lacked domain-specific priors.

These results underscore that prompting is not a neutral interface—it actively steers the model's cognition. Selecting prompts that fit both the model's inductive biases and the task's logical structure is key to successful deployment.

3.6. Discussion on the direct querying of LLMs

In this evaluation, we explored a wide range of prompting strategies to guide large language models (LLMs) in extracting ⟨extent, intent⟩ pairs from binary formal contexts. These included zero-shot, few-shot, role prompting, contextual prompting, step-back prompting, chain of thought (CoT), tree of thought (ToT) and self-consistency techniques. Each was systematically tested across five LLMs: ChatGPT-4o, Claude 4 Sonnet, Gemini 2.5 Pro, DeepSeek-V3, and Mistral Large, on a variety of context sizes.

The experiments revealed several key findings:

– No prompt was universally effective. Across all models and context sizes, no single prompt outperformed all others consistently. A prompt that worked well with one LLM might fail entirely with another.

– Performance was highly model-dependent. Even with the same prompt and the same context, the five LLMs produced drastically different results. This highlights how prompt effectiveness is closely tied to each model's internal architecture and reasoning capabilities.

– Some advanced prompts stood out. Among all techniques tested, step-back prompting, chain-of-thought, and especially tree-of-thought delivered the most robust and high-quality results. These were particularly effective with reasoning-oriented models such as Claude 4 Sonnet and Gemini 2.5 Pro.

– Prompt structure mattered more than length. Prompts that encouraged the model to "think before answering" — for example by listing options before choosing one (step-back), or following a structured procedure (CoT, ToT) — helped reduce hallucinations and improved logical consistency.

– Self-consistency increased stability, but not always accuracy. Repeating zero-shot prompts and selecting the best answer slightly improved reliability, but did not guarantee correct outputs, especially on larger contexts.

– Larger contexts drastically reduced accuracy. No matter what prompt was used, all LLMs showed a marked decline in output quality in contexts exceeding 15 objects $\times$ 15 attributes. Even strong performers like Claude and Gemini began hallucinating objects, renaming them (e.g., "O1" 'Patient 1') or inventing attributes.

Prompt engineering plays a crucial role in determining the success of LLMs in FCA tasks. Although basic prompts suffice for simple cases, solving more realistic or complex problems requires thoughtfully designed prompts that engage the model's reasoning. Among the techniques tested, step-back, chain-of-thought, and tree-of-thought were the most effective across models — though none of them succeeded consistently on all LLMs.

In the end, symbolic algorithms remain essential for exhaustive and reliable extraction of formal concepts, especially on large contexts. But for small to medium contexts, the right combination of prompt strategy and LLM — particularly with Claude 4 Sonnet or Gemini 2.5 Pro — can yield surprisingly strong results.

4. Perspectives

While our current benchmarking study offers a first look at how prompting strategies and model types affect LLM performance on Formal Concept Analysis (FCA) tasks, it also highlights key limitations that suggest rich avenues for further exploration. With access to a one-year research budget, several directions emerge that could substantially improve both empirical performance and theoretical understanding.

First, our evaluation pipeline could be expanded with a **systematic hyperparameter sweep**, particularly around sampling temperature, max token length, and decoding strategies (e.g., nucleus sampling vs. greedy decoding). These variables were held constant for tractability, but tuning them per model and per task could reveal significant latent capacity, especially in direct enumeration settings where performance was weakest.

Second, the **development of FCA-specific prompt engineering techniques** offers promising leverage. This could include creating synthetic training prompts that expose the model to hundreds or thousands of toy FCA problems during fine-tuning, or the use of meta-prompts that teach the model to self-check for concept closure violations. Role prompting could also be refined to inject more domain-specific heuristics, such as enforcing anti-chain constraints or duality symmetry in enumerated concepts.

Third, we would propose training a **small domain-adapted language model**, either through instruction tuning or continual pretraining, using corpora derived from FCA literature, source code repositories, and logic-heavy symbolic reasoning datasets. Unlike general-purpose LLMs, such a model could acquire inductive biases more aligned with formal structure manipulation. Coupled with contrastive evaluation datasets (e.g., nearly-correct but invalid concept sets), this would enable precise assessment of both syntactic fluency and semantic correctness.

Finally, it would be valuable to extend our benchmark to **multimodal and multilingual settings**, including tasks that involve interpreting concept lattices as diagrams or generating concepts from textual contexts in non-English languages. Such an extension would test the generalization capacity of models across input modalities and linguistic settings, broadening the practical scope of FCA automation.

Taken together, these perspectives point toward a multi-pronged research agenda that encompasses prompt design, model adaptation and hybrid reasoning.

A. Annex: Prompts used for LLM evaluation

A.1. Valentino's Prompts

A.1.1. Pre-Prompt (applied before each prompt)

Pre-Prompt

In the field of Formal Concept Analysis (FCA), explain the theoretical framework necessary to implement a concept derivation algorithm. The objective is to find all of the correct concepts. Your explanation should include:

1. What constitutes a formal context within the FCA framework, and how can it be represented in code?

2. What is a formal concept in FCA, and how is it defined mathematically?

3. What is the role of extents and intents in defining formal concepts, and how do they emerge through the Galois connection?

4. What is the process by which formal concepts are derived?

5. How should formal concepts be represented programmatically?

6. What are algorithms such as `NextClosure` or other enumeration strategies commonly used to compute all formal concepts from a given context?

A.1.2. Zero-shot Prompt

Zero-shot

Now using what you've just explained, write Java code that reads a semicolon-separated CSV representing a formal context and computes all formal concepts (intent and extent).

A.1.3. One-shot Prompt

One-shot

The CSV file looks like this: ;a1;a2
o1;0;1
o2;1;1

This means that the first row contains attribute names (columns), the first column contains object names (rows), and a cell with 1 means the object has that attribute.

For this input, the formal concepts are:
Extent: [o1, o2] Intent: [a2]
Extent: [o2] Intent: [a1, a2]

Now using the same logic, write Java code that reads the following semicolon-separated CSV file and computes all formal concepts (intent and extent). Print each concept like:
Extent: [object list] Intent: [attribute list]

New input:
;a1;a2;a3;a4;a5
o1;1;0;0;1;0
o2;0;0;0;1;0
o3;1;0;0;0;1
o4;1;1;1;0;1
o5;0;0;1;0;1

A.1.4. Few-Shot Prompt

Few-Shot

Example 1 Input:
;a1;a2
o1;0;1
o2;1;1
Explanation: The first row contains attribute names, and the first column contains object names. A 1 means the object has that attribute.

Output:
Extent: [o1, o2] Intent: [a2]
Extent: [o2] Intent: [a1, a2]

Example 2 Input:
;a1;a2;a3
o1;1;1;0
o2;1;1;0
o3;0;1;1

Output:
Extent: [o1, o2, o3] Intent: []
Extent: [o1, o2] Intent: [a1]
Extent: [o3] Intent: [a2, a3]
Extent: [] Intent: [a1, a2, a3]

Now write Java code that reads the following semicolon-separated CSV file and computes all formal concepts (intent and extent). Print each concept like:
Extent: [object list] Intent: [attribute list]

New Input:
;a1;a2;a3;a4;a5
o1;1;0;0;1;0
o2;0;0;0;1;0
o3;1;0;0;0;1
o4;1;1;1;0;1
o5;0;0;1;0;1

A.1.5. Role Prompting

Role Prompting

You are a computational theory researcher specializing in Formal Concept Analysis (FCA) and Java-based algorithm prototyping. As part of your role, your task is to write well-structured and documented Java code that reads a semicolon-separated CSV representing a formal context and computes all formal concepts (intent and extent). Use your expertise to do this.

A.1.6. Chain of Thought Prompt

Chain of Thought

Prompt 1: Begin by explaining how to parse the CSV file into a formal context (objects, attributes, and their relations).

Prompt 2: Describe the data structures you would use to represent the context and to track extents and intents.

Prompt 3: Explain how you would systematically compute formal concepts using closure operators.

Prompt 4: Describe how you would avoid duplicates and infinite loops during the computation.

Prompt 5: Now using what you've just explained, write Java code that reads a semicolon-separated CSV representing a formal context and computes all formal concepts (intent and extent).

A.1.7. Step-Back Prompt

Step-Back Prompting

Prompt 1: I want to compute all formal concepts (intent and extent) from a semicolon-separated CSV representing a formal context using Java. The CSV file looks like this:
;a1;a2
o1;0;1
o2;1;1
What kind of computational task is this?

Prompt 2: What kind of input is involved?

Prompt 3: What is the goal of this kind of task?

Prompt 4: What general strategy or reasoning process applies to solving it?

Prompt 5: What common challenges should be considered when implementing this in Java (e.g., infinite loops, duplicate concepts)?

Prompt 6: Now, using what you've just explained, write Java code that reads a semicolon-separated CSV representing a formal context and computes all formal concepts (intent and extent).

A.1.8. Self-Consistency Prompt

Self-Consistency

Please solve the following Java coding task five times independently—for each attempt, show your full chain of thought and then your final Java implementation. After you've given all five full reasoning traces and code snippets, compare the five solutions and state which implementation is the best by majority vote, explaining why. For each attempt, you may use your own reasoning and implementation strategy.

Task: Write Java code that reads a semicolon-separated CSV file representing a formal context and computes all formal concepts (intent and extent).

The CSV will look like this:
;a1;a2
o1;0;1
o2;1;1

This means that the first row contains attribute names (columns), the first column contains object names (rows), and a cell with 1 means the object has that attribute.

Each concept should be printed in this format:
Extent: [object list] Intent: [attribute list]

Generate at least three independent implementations. Then compare them and determine which solution is the most correct and consistent with the task requirements.

Instructions:
Chain-of-Thought #1 → Code #1
Chain-of-Thought #2 → Code #2
Chain-of-Thought #3 → Code #3
Chain-of-Thought #4 → Code #4
Chain-of-Thought #5 → Code #5

Comparison & Vote: After all five, pick the implementation that is most correct and complete, and explain your choice.

A.1.9. Tree of Thought

Tree of Thought

Step 0: Initial Setup
Thought A0: “I will parse CSV with BufferedReader + String.split.”
Thought B0: “I will use Scanner + delimiter regex.”
Thought C0: “I will use java.nio.file + Streams API.”
Evaluate A0 vs B0 vs C0 (compare simplicity, performance, ease of use)
Prune all but the best parsing strategy

Step 1: Data Structures for Context
Thought A1: “Store incidence in boolean[][].”
Thought B1: “Store in List<BitSet>.”
Thought C1: “Store in Map<String, Set<String».”
Evaluate memory and access patterns
Prune to one

Step 2: Concept Generation Algorithm
Thought A2: “Use NextClosure (Ganter’s) algorithm.”
Thought B2: “Use Close-by-One (CbO) algorithm.”
Thought C2: “Use brute-force double closure (all subsets).”
Evaluate time complexity and clarity
Prune to one

Step 3: Code Structure and API
Thought A3: “One monolithic main() method.”
Thought B3: “Split into FormalContext, FormalConcept, Algorithm classes.”
Evaluate maintainability and readability
Prune to one

Step 4: Final Implementation
Based on the surviving branch choices, write the complete Java code. Ensure it:
- Reads the CSV correctly
- Computes all formal concepts
- Prints each in the required format

A.2. Oscar's prompts

A.2.1. Zero-shot Prompt

Zero-shot

Generate a Java program that computes all formal concepts from a given context in Formal Concept Analysis (FCA). The input is a CSV file where rows represent objects, columns represent attributes, and cell values indicate whether an object has an attribute (e.g., 'X' for presence, empty for absence). The CSV file path should be hardcoded for use in Eclipse IDE.
The program should:
- Read the CSV file from a fixed path (e.g., "C:/input.csv").
- Parse the context into a binary matrix (objects × attributes).
- Implement the NextClosure algorithm (or Ganter's algorithm) to compute all formal concepts.
- Print each formal concept as a pair (extent, intent), where extent is the set of objects and intent is the set of attributes.
- Use only standard Java libraries (no external dependencies). Ensure the code compiles and runs in Eclipse IDE.

A.2.2. One-shot Prompt

One-shot

Goal: Write a Java program to be run on Eclipse that prints all concepts of a context (as in Formal Concept Analysis).
EXAMPLE 1
input:
;a1;a2;a3
o1;1;1;0
o2;0;1;1
o3;1;0;0
o4;0;0;0

output:
— Found Formal Concepts (6) —
1. (Extent: {o1, o2, o3, o4}, Intent: {})
2. (Extent: {o1, o3}, Intent: {a1})
3. (Extent: {o1, o2}, Intent: {a2})
4. (Extent: {o1}, Intent: {a1, a2})
5. (Extent: {o2}, Intent: {a2, a3})
6. (Extent: {}, Intent: {a1, a2, a3})

A.2.3. Few-shot Prompt

Few-Shot

Goal: Write a Java program to be run on Eclipse that prints all concepts of a context (as in Formal Concept Analysis). The context is a CSV file given as a hard-coded path.

EXAMPLE 1

input:
;a1;a2;a3
o1;1;1;0
o2;0;1;1
o3;1;0;0
o4;0;0;0

output:
— Found Formal Concepts (6) —
1. (Extent: {o1, o2, o3, o4}, Intent: {})
2. (Extent: {o1, o3}, Intent: {a1})
3. (Extent: {o1, o2}, Intent: {a2})
4. (Extent: {o1}, Intent: {a1, a2})
5. (Extent: {o2}, Intent: {a2, a3})
6. (Extent: {}, Intent: {a1, a2, a3})

EXAMPLE 2

input:
;a1;a2;a3;a4
o1;0;0;0;1
o2;0;1;1;1
o3;1;0;0;1
o4;1;1;1;1
o5;1;0;1;1
o6;0;1;0;0
o7;0;1;1;1

output:
— Found Formal Concepts (8) —
1. (Extent: {o1, o2, o3, o4, o5, o6, o7}, Intent: {})
2. (Extent: {o3, o4, o5}, Intent: {a1, a4})
3. (Extent: {o2, o4, o6, o7}, Intent: {a2})
4. (Extent: {o4}, Intent: {a1, a2, a3, a4})
5. (Extent: {o2, o4, o5, o7}, Intent: {a3, a4})
6. (Extent: {o4, o5}, Intent: {a1, a3, a4})
7. (Extent: {o2, o4, o7}, Intent: {a2, a3, a4})
8. (Extent: {o1, o2, o3, o4, o5, o7}, Intent: {a4})

A.2.4. Role Prompting

Role Prompting

Goal: I want you to act as a computer scientist. Your research necessitates that you write a Java program to be run on Eclipse that prints all concepts of a context (as in Formal Concept Analysis). The context is a CSV file given as a hard-coded path.

A.2.5. Chain of Thought Prompt

Chain of Thought

Prompt 1:
Explain the core ideas of Formal Concept Analysis (FCA) as if I were a programmer new to the topic. Cover:
- What is a "formal context" (objects × attributes matrix)?
- Define "formal concept" (extent + intent) with an example.
- What properties must a valid concept satisfy?

Use this binary context as an example (rows = objects, cols = attributes):
Object 0: [1, 0, 1]
Object 1: [0, 1, 1]
Object 2: [1, 1, 0]

Prompt 2:
For the context above, I want to compute all formal concepts programmatically.
1. Compare NextClosure vs. Lindig's algorithm for this task. Which is simpler to implement in Java?
2. Provide pseudocode for the chosen algorithm, highlighting:
- How to compute the closure of an attribute set.
- How to generate concepts without duplicates.

Prompt 3:
Write a Java class `FormalConceptAnalyzer` to be run on Eclipse IDE that:
1. Takes a CSV-coded context as a hardcoded path.
2. Uses the algorithm from Prompt 2 to find all concepts.
3. Represents each concept as `List<Concept>` with your own class design.
Write your answer in one single Java block.

A.2.6. Step-Back Prompt

Step-Back Prompting

Prompt 1:
Based on published computer science papers and well-known practical algorithms, what are the best ways to compute formal concepts given a formal context? List the best methods and explain them without writing any code yet.

Prompt 2:
Context: Best methods for computing formal concepts:
- NextClosure (Ganter 1984)
- AddIntent (Godin et al. 1995)

Take one of these methods and write a Java program to be run on Eclipse IDE that reads a CSV file from a hardcoded path and prints all of the formal concepts.

A.2.7. Self-Consistency Prompt

Self-Consistency

We chose `result4.java` because it has every feature that the other programs have. `result4.java` is the one that most closely represents the whole 5 results together.

Prompt:
I have a CSV representing a formal context (as in formal context analysis) and I want to compute all of the formal concepts associated with it. Let's think step by step and write a Java program (to be run on Eclipse IDE) that reads a CSV file from a hardcoded path and prints the associated concepts. Remember to think step by step.

A.3. Arame et Yazmín's prompts

A.3.1. Zero-shot Prompt

Zero-shot

PROMPT 1:
In the field of formal concept analysis (FCA), given this .csv file with a context, I want you to extract and enumerate all lattice concepts

PROMPT 2:
Put the list of concepts in a single line with this format:
`[[[], [o1, o2, o3, o4, o5]], [[a1], [o1, o3, o4]], [[a1, a2, a3, a5], [o4]]]` without any quotation marks.

A.3.2. Few-shot Prompt

Few-shot

PROMPT 1:
In the field of formal concept analysis (FCA), given this .csv file with a context, I want you to extract and enumerate all lattice concepts.

EXAMPLE:
For the following context:
```
;a1;a2;a3;a4;a5;a6;a7;a8;a9;a10;a11;a12;a13;a14;a15;a16
o1;no;no;no;no;no;no;no;no;yes;no;yes;no;no;yes;yes;no
o2;yes;no;yes;no;yes;no;no;yes;yes;no;no;yes;yes;yes;no;no
o3;no;yes;no;no;yes;no;yes;no;yes;yes;no;yes;no;no;no;no
```
The list of lattice concepts is:
`[[[a9], [o1, o2, o3]], [[a1, a3, a5, a8, a9, a12, a13, a14], [o2]], [[a1, a2, a3, a4, a5, a6, a7, a8, a9, a10, a11, a12, a13, a14, a15, a16], []], [[a2, a5, a7, a9, a10, a12], [o3]], [[a5, a9, a12], [o2, o3]], [[a9, a11, a14, a15], [o1]], [[a9, a14], [o1, o2]]]`

PROMPT 2:
Put the list of concepts in a single line with this format:
`[[[], [o1, o2, o3, o4, o5]], [[a1], [o1, o3, o4]], [[a1, a2, a3, a5], [o4]]]` without any quotation marks.

A.3.3. Role Prompting

Role Prompting

PROMPT 1:
I want you to act as a computer science teacher who's explaining to their students.

Within the field of formal concept analysis (FCA), given this .csv file with a context, I want you to extract and enumerate all lattice concepts.

PROMPT 2:
Put the list of concepts in a single line with this format:
`[[[], [o1, o2, o3, o4, o5]], [[a1], [o1, o3, o4]], [[a1, a2, a3, a5], [o4]]]` without any quotation marks.

A.3.4. Contextual Prompting

Contextual Prompting

PROMPT 1:
In the field of formal concept analysis (FCA), we can find the lattice concepts from small contexts (less than 20 objects and less than 20 attributes) by simple methods rather than powerful algorithms.

Given this .csv file with a context, I want you to extract and enumerate all lattice concepts.

PROMPT 2:
Put the list of concepts in a single line with this format:
`[[[], [o1, o2, o3, o4, o5]], [[a1], [o1, o3, o4]], [[a1, a2, a3, a5], [o4]]]` without any quotation marks.

A.3.5. Step-Back Prompting

Step-Back Prompting

PROMPT 1:
In the field of formal concept analysis (FCA), give me three ways to find the lattice concepts in a context with less than 20 objects and less than 20 attributes.

PROMPT 2:
Pick one of the three ways that you mentioned to extract all the lattice concepts from the context in this .csv file.

PROMPT 3:
Put the list of concepts in a single line with this format:
`[[[], [o1, o2, o3, o4, o5]], [[a1], [o1, o3, o4]], [[a1, a2, a3, a5], [o4]]]` without any quotation marks.

A.3.6. Chain of thought

Chain of thought

PROMPT 1:
In the field of formal concepts analysis (FCA), find all the lattice concepts in this .csv file with a context by following these steps
1. Build a binary relationship matrix from the file, indicating which objects possess which attributes
2. Make a list of every possible subset of the objects found in the CSV file, consider too the empty subsets. Each such subset is going to be considered a candidate for an "extent" of a formal concept
3. For the first candidate object subset A, determine the set of attributes B that are shared by all objects in the subset A. This set B is the "intent" corresponding to A.
4. From this set of attributes B, find all objects in the entire context that possess every attribute in B. Let's call this new set of objects A_prime.
5. If the candidate object subset A is identical to the derived set of objects A_prime (A = A_prime), then the pair (A, B) is a formal concept.
6. Repeat steps 3, 4 and 5 for every subset of objects from step 2 4. Collect all unique concepts found through this process

PROMPT 2:
Put the list of concepts in a single line with this format:
`[[[], [o1, o2, o3, o4, o5]], [[a1], [o1, o3, o4]], [[a1, a2, a3, a5], [o4]]]` without any quotation marks.

A.3.7. Self consistency

Self consistency

PROMPT 1 (repeat three times, choose the one with the best results):
In the field of formal concept analysis (FCA), given this .csv file with a context, I want you to extract and enumerate all lattice concepts

PROMPT 2:
Put the list of concepts in a single line with this format:
`[[[], [o1, o2, o3, o4, o5]], [[a1], [o1, o3, o4]], [[a1, a2, a3, a5], [o4]]]` without any quotation marks.

A.3.8. Tree of thoughts

Tree of thoughts

PROMPT 1A (repeat three times, take one of the solutions from each try for PROMPT 1B):
In the field of formal concept analysis (FCA), give me three ways to find the lattice concepts in a context with less than 20 objects and less than 20 attributes. Write a single paragraph for each way

PROMPT 1B:
Pick one of the next three ways to extract all the lattice concepts from the context in this .csv file and apply it to give me the concepts:

(insert three of the ways suggested with prompt 1A)

PROMPT 2:
Put the list of concepts in a single line with this format:
`[[[], [o1, o2, o3, o4, o5]], [[a1], [o1, o3, o4]], [[a1, a2, a3, a5], [o4]]]` without any quotation marks.

B. Annex B: Performance data tables and graphs

B.1. ChatGPT-o3-mini-high

B.1.1. Chain of Thought - Oscar

The follwoing table depicts the gathered results of our tests.

Example	F1-Score	Recall	Accuracy	LLM Running Time	Fcs4j Running Time	False Positives	True Positives	False Negatives	Notes
eg9_9.csv	0 %	0 %	0 %	1s 316ms	12ms	6	0	34	
eg20_20.csv	0 %	0 %	0 %	4s 242ms	30ms	7	0	573	
eg29_30.csv	0 %	0 %	0 %	5s 215ms	93ms	10	0	3653	
eg29_31.csv	0 %	0 %	0 %	3s 563ms	90ms	9	0	3398	
eg30_29.csv	0 %	0 %	0 %	4s 898ms	97ms	8	0	3857	
eg30_30_2.csv	0 %	0 %	0 %	5s 503ms	85ms	7	0	3132	
eg30_30.csv	0 %	0 %	0 %	6s 208ms	101ms	9	0	4130	
eg30_31.csv	0 %	0 %	0 %	8s 507ms	111ms	9	0	4337	
eg30_40.csv	0 %	0 %	0 %	6s 271ms	265ms	9	0	10355	
eg30_80.csv	0 %	0 %	0 %	9s 292ms	1s 500ms	10	0	51877	
eg30_300.csv	0 %	0 %	0 %	12s 695ms	1m 19s 265ms	10	0	655244	
eg31_10.csv	0 %	0 %	0 %	9s 228ms	22ms	8	0	219	
eg31_29.csv	0 %	0 %	0 %	8s 693ms	110ms	9	0	4227	
eg31_31_2.csv	0 %	0 %	0 %	4s 643ms	138ms	8	0	5400	
eg31_31.csv	0 %	0 %	0 %	7s 443ms	111ms	8	0	4489	
eg31_32.csv	0 %	0 %	0 %	9s 504ms	141ms	9	0	5484	
eg32_32.csv	0 %	0 %	0 %	4s 457ms	131ms	9	0	4981	
eg35_35.csv	0 %	0 %	0 %	6s 931ms	209ms	10	0	8295	
eg40_40.csv	0 %	0 %	0 %	4s 740ms	500ms	10	0	19627	
eg50_50_2.csv	0 %	0 %	0 %	4s 469ms	2s 611ms	10	0	86581	
eg50_50.csv	0 %	0 %	0 %	6s 822ms	2s 672ms	10	0	85492	
eg1726_20.csv	0 %	0 %	0 %	29s 784ms	27s 299ms				
balance-scale_binarized.csv	0 %	0 %	0 %	9s 420ms	9ms	7	0	2106	
bank_binarized.csv	0 %	0 %	0 %	2m 20s 663ms	3m 2s 551ms	8	0	427587	
breast-cancer_binarized.csv	0 %	0 %	0 %	13s 213ms	9ms	9	0	9944	
car_binarized.csv	0 %	0 %	0 %	7s 599ms	10ms	8	0	8001	
cmc_binarized.csv	0 %	0 %	0 %	2s 624ms	10ms				Error: Index out of bounds
CombinedRoles.csv	0 %	0 %	0 %	10s 354ms	127ms	9	0	2780	
flare_binarized.csv	0 %	0 %	0 %	4s 785ms	10ms	12	0	10300	
lymphography_binarized.csv	0 %	0 %	0 %	6s 73ms	9ms	7	0	51939	
monks-1_binarized.csv	0 %	0 %	0 %	4s 17ms	9ms	9	0	4465	
nom5bikesharing_day_cut.csv	0 %	0 %	0 %	1s 245ms	1s 834ms	11	0	61853	
nursery_binarized.csv	0 %	0 %	0 %	4s	9ms	10	0	115201	
ord5bikesharing_day_cut.csv	0 %	0 %	0 %	2s	3s 360ms	23	0	81277	
primary-tumor_binarized.csv	0 %	0 %	100 %	4s	8ms	18	0	279384	
soybean_binarized.csv	0 %	0 %	0 %	10s 306ms	9ms				Error: Index out of bounds
spect_binarized.csv	0 %	0 %	0 %	7s 706ms	9ms	15	0	14534	
tic-tac-toe_binarized.csv	0 %	0 %	0 %	1s 715ms	1s 631ms	9	0	59505	
trains_binarized.csv	0 %	0 %	0 %	8s 323ms	16ms	8	0	91	
voting-records_binarized.csv	0 %	0 %	0 %	9s	14s 342ms	9	0	264035	
winequality-red_binarized.csv	0 %	0 %	0 %	5s 867ms	9ms				Error: Index out of bounds

Figure B.1.

Running times for Chain Of Thought - ChatGPT-o3-mini-high. Figure B.1 depicts the data for the synthetic dataset and B.2 for the real dataset.

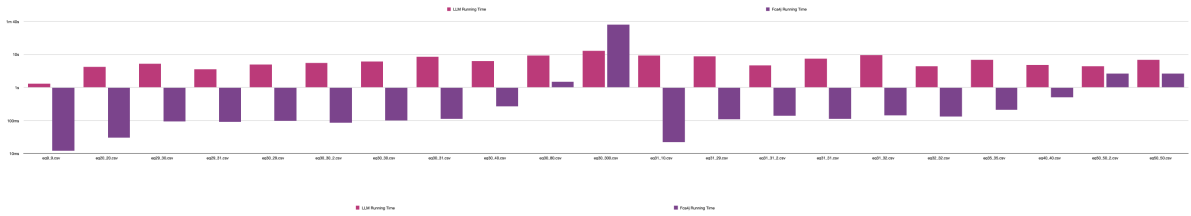

Figure B.2.

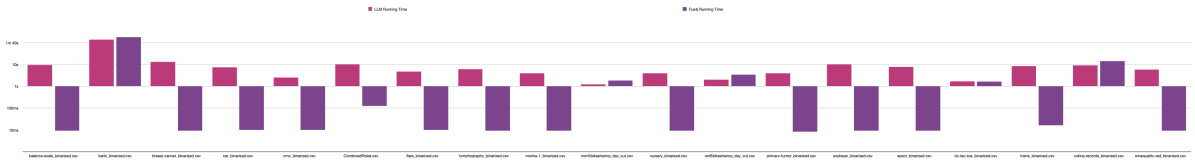

Figure B.3.

B.1.2. One Shot - Oscar

The measurements for One Shot are shown in Figure B.5 and Figure B.4 depicts the gathered results.

Example	F1-Score	Recall	Precision	LLM Running Time	Fca4j Running Time	False Positives	True Positives	False Negatives	Notes
eg9_9.csv	100 %	100 %	100 %	1s 71ms	12ms	0	34	0	
eg20_20.csv	100 %	100 %	100 %	7s 854ms	30ms	0	573	0	
eg29_30.csv	100 %	100 %	100 %	1h 8m 20s	93ms		3653		
eg29_31.csv	0 %	0 %	0 %	944ms	90ms	0	0	3398	
eg30_29.csv	100 %	100 %	100 %	49m 13s 693ms	97ms	0	3857	0	
eg30_30_2.csv	100 %	100 %	100 %	41m 41s 280ms	85ms	0	3132	0	
eg30_30.csv	100 %	100 %	100 %	1h 10m 51s	101ms	0	4130	0	
eg30_31.csv	0 %	0 %	0 %	1s 989ms	111ms	0	0	4337	
eg30_40.csv	0,14 %	0,07 %	100 %	2s 614ms	265ms	0	7	10348	
eg30_80.csv	0,01 %	0 %	100 %	4s 648ms	1s 500ms	0	2	51875	
eg30_300.csv	0 %	0 %	100 %	30s 81ms	1m 19s 265ms	0	1	655243	
eg31_10.csv	100 %	100 %	100 %	1s 492ms	22ms	0	219	0	
eg31_29.csv	100 %	100 %	100 %	47m 38s 21ms	110ms	0	4227	0	
eg31_31_2.csv	0 %	0 %	0 %	3s 58ms	138ms	0	0	5400	
eg31_31.csv	0 %	0 %	0 %	2s 976ms	111ms	0	0	4489	
eg31_32.csv	0,04 %	0,02 %	100 %	2s 364ms	141ms	0	1	5483	
eg32_32.csv	0,04 %	0,02 %	100 %	2s 393ms	131ms	0	1	4980	
eg35_35.csv	0,1 %	0,05 %	100 %	2s 678ms	209ms	0	4	8291	
eg40_40.csv	0,1 %	0,05 %	100 %		500ms	0	10	19617	
eg50_50_2.csv	0,08 %	0,04 %	100 %	6s 38ms	2s 611ms	0	33	86548	
eg50_50.csv	0,07 %	0,04 %	100 %	6s 397ms	2s 672ms	0	30	85462	
eg1726_20.csv	100 %	100 %	100 %	59s 899ms	27s 299ms	0	341613	0	
balance-scale_binarized.csv	100 %	100 %	100 %	45s 728ms	9ms	0	2106	0	
bank_binarized.csv	0 %	0 %	100 %	44s 472ms	3m 2s 551ms	0	1	427586	
breast-cancer_binarized.csv	0 %	0 %	0 %	978ms	9ms	9	0	1	
car_binarized.csv	100 %	100 %	100 %	898ms	10ms	0	8001	0	
cmc_binarized.csv	0 %	0 %	0 %	9s 917ms	10ms	1	0	1	
CombinedRoles.csv					127ms				
flare_binarized.csv	0 %	0 %	0 %	2s 733ms	10ms	0	0	10300	
lymphography_binarized.csv	0 %	0 %	0 %	2s 352ms	9ms	0	0	51939	
monks-1_binarized.csv	100 %	100 %	100 %	2s 929ms	9ms	0	4465	0	
nom5bikesharing_day_cut.csv	0 %	0 %	100 %	2s 262ms	1s 834ms	0	1	61852	
nursery_binarized.csv	100 %	100 %	100 %	2h 26m 52s	9ms		115200		
ord5bikesharing_day_cut.csv	0 %	0 %	100 %	10m 28s 530ms	3s 360ms	0	2	81275	
primary-tumor_binarized.csv	0 %	0 %	0 %	13s 756ms	8ms	0	0	279384	
soybean_binarized.csv	0 %	0 %	0 %	3m 36s	9ms				Java Error heap space
spect_binarized.csv	100 %	100 %	100 %	50s 676ms	9ms	0	14534	0	
tic-tac-toe_binarized.csv	100 %	100 %	100 %	1h 40m 29s 985ms	1s 631ms	0	59505	0	
trains_binarized.csv	0 %	0 %	0 %	21s 422ms	16ms	0	0	91	
voting-records_binarized.csv	0,01 %	0,01 %	100 %	18s 103ms	12s 133ms	0	16	264019	
winequality-red_binarized.csv	0,01 %	0,01 %	100 %	38s 485ms	9ms	0	1	18249	

Figure B.4.

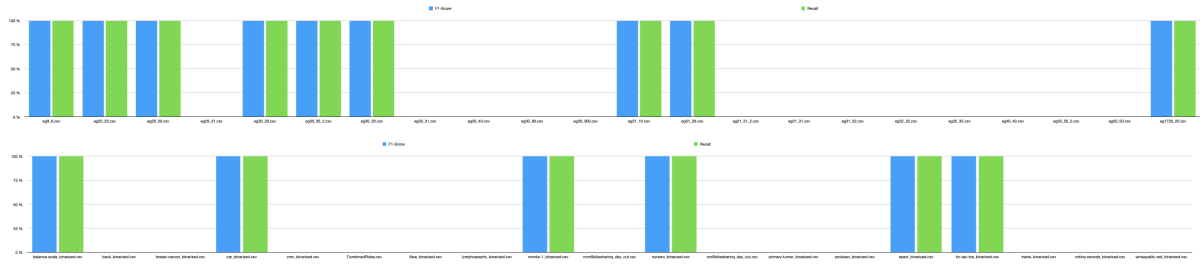

Figure B.5.

Running times for One Shot - - ChatGPT-o3-mini-high. Figure B.6 depicts the data for the synthetic dataset and B.7 for the real dataset.

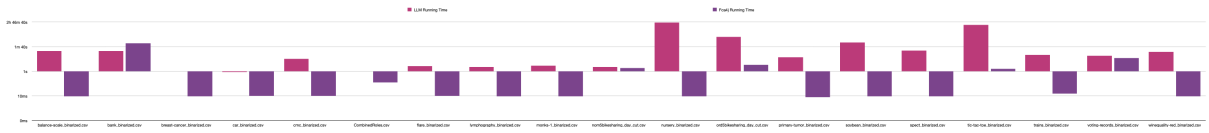

Figure B.6.

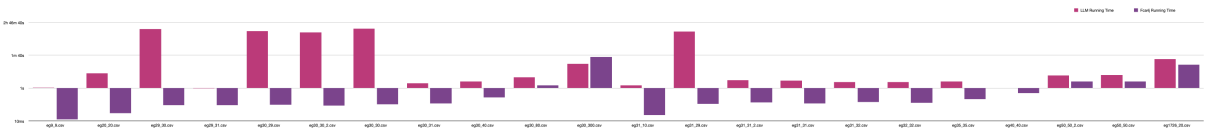

Figure B.7.

B.1.3. Role - Oscar

Example	F1-Score	Accuracy	Recall	LLM Running Time	Fca4j running time	Notes
eg9_9.csv	100 %	100 %	100 %	3s 967ms	12ms	
eg20_20.csv	100 %	100 %	100 %	862ms	30ms	
eg29_30.csv	100 %	100 %	100 %	1s 228ms	93ms	
eg29_31.csv	100 %	100 %	100 %	1s 274ms	90ms	
eg30_29.csv	100 %	100 %	100 %	1s 339ms	97ms	
eg30_30_2.csv	100 %	100 %	100 %	1s 184ms	85ms	
eg30_30.csv	100 %	100 %	100 %	1s 293ms	101ms	
eg30_31.csv	100 %	100 %	100 %	1s 265ms	111ms	
eg30_40.csv	100 %	100 %	100 %	1s 719ms	265ms	
eg30_80.csv	100 %	100 %	100 %	5s 172ms	1s 500ms	
eg30_300.csv	100 %	100 %	100 %	32m	1m 19s 265ms	
eg31_10.csv	100 %	100 %	100 %	598ms	22ms	
eg31_29.csv	100 %	100 %	100 %	1s 125ms	110ms	
eg31_31_2.csv	100 %	100 %	100 %	1s 232ms	138ms	
eg31_31.csv	100 %	100 %	100 %	1s 170ms	111ms	
eg31_32.csv	100 %	100 %	100 %	1s 250ms	141ms	
eg32_32.csv	100 %	100 %	100 %	1s 218ms	131ms	
eg35_35.csv	100 %	100 %	100 %	1s 378ms	209ms	
eg40_40.csv	100 %	100 %	100 %	2s 30ms	500ms	
eg50_50_2.csv	100 %	100 %	100 %	4s 909ms	2s 611ms	
eg50_50.csv	100 %	100 %	100 %	5s 470ms	2s 672ms	
eg1726_20.csv	100 %	100 %	100 %	18s	27s 299ms	
balance-scale_binarized.csv	100 %	100 %	100 %	1s 50ms	9ms	
bank_binarized.csv	0 %	0 %	0 %	2d 2h 11m	3m 2s 551ms	Cancelled. No errors.
breast-cancer_binarized.csv	100 %	100 %	100 %	8s 250ms	9ms	
car_binarized.csv	100 %	100 %	100 %	5s 92ms	10ms	
cmc_binarized.csv	100 %	100 %	100 %	42s 753ms	10ms	
CombinedRoles.csv	100 %	100 %	100 %	2m 22s 812ms	127ms	
flare_binarized.csv	100 %	100 %	100 %	11s 509ms	10ms	
lymphography_binarized.csv	100 %	100 %	100 %	16s 826ms	9ms	
monks-1_binarized.csv	100 %	100 %	100 %	3s 760ms	9ms	
nom5bikesharing_day_cut.csv	100 %	100 %	100 %	20s 546ms	1s 834ms	
nursery_binarized.csv	100 %	100 %	100 %	1m 49s 858ms	9ms	
ord5bikesharing_day_cut.csv	100 %	100 %	100 %	21s 745ms	3s 360ms	
primary-tumor_binarized.csv	100 %	100 %	100 %	2m 1s 869ms	8ms	
soybean_binarized.csv	0 %	0 %	0 %	4h 19m 7s 592ms	9ms	Java error: Heap space
spect_binarized.csv	100 %	100 %	100 %	8s 101ms	9ms	
tic-tac-toe_binarized.csv	100 %	100 %	100 %	17s 219ms	1s 631ms	
trains_binarized.csv	100 %	100 %	100 %	4s 326ms	16ms	
voting-records_binarized.csv	100 %	100 %	100 %	1m 53s 228ms	12s 133ms	
winequality-red_binarized.csv	100 %	100 %	100 %	59m 28s 453ms	9ms	

Figure B.8.

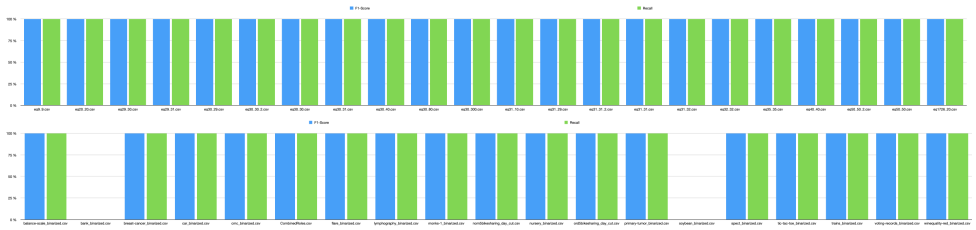

Figure B.9.

Figure B.10.

B.1.4. Few Shot - Oscar

Example	F1-Score	Recall	Precision	LLM Running Time	Fca4j Running Time	False Positives	True Positives	False Negatives	Notes
eg9_9.csv	100 %	100 %	100 %	526ms	12ms	0	34	0	
eg20_20.csv	100 %	100 %	100 %	2s 105ms	30ms	0	573	0	
eg29_30.csv	100 %	100 %	100 %	1h 40m 0s 989ms	93ms	0	3653	0	
eg29_31.csv	0 %	0 %	0 %	631ms	90ms	0	0	3398	
eg30_29.csv	100 %	100 %	100 %	50m 2s 297ms	97ms	0	3857	0	
eg30_30_2.csv	100 %	100 %	100 %	1h 42m 39s 697ms	85ms	0	3132	0	
eg30_30.csv	100 %	100 %	100 %	1h 44m 59s 832ms	101ms	0	4130	0	
eg30_31.csv	0 %	0 %	0 %	1s 390ms	111ms	0	0	4337	
eg30_40.csv	0,14 %	0,07 %	100 %	1s 858ms	265ms	0	7	10348	
eg30_80.csv	0,01 %	0 %	100 %	2s 896ms	1s 500ms	0	2	51875	
eg30_300.csv	0 %	0 %	100 %	18s 96ms	1m 19s 265ms	0	1	655243	
eg31_10.csv	100 %	100 %	100 %	1s 399ms	22ms	0	0	219	
eg31_29.csv	100 %	100 %	100 %	11h 11m 22s 706ms	110ms	0	4227	0	
eg31_31_2.csv	0 %	0 %	0 %	1s 672ms	138ms	0	0	5400	
eg31_31.csv	0 %	0 %	0 %	4s 376ms	111ms	0	0	4489	
eg31_32.csv	0,04 %	0,02 %	100 %	4s 215ms	141ms	0	1	5483	
eg32_32.csv	0,04 %	0,02 %	100 %	4s 185ms	131ms	0	1	4980	
eg35_35.csv	0,1 %	0,05 %	100 %	4s 784ms	209ms	0	4	8291	
eg40_40.csv	0,1 %	0,05 %	100 %	6s 128ms	500ms	0	10	19617	
eg50_50_2.csv	0,08 %	0,04 %	100 %	10s 264ms	2s 611ms	0	33	86548	
eg50_50.csv	0,07 %	0,04 %	100 %	10s 410ms	2s 672ms	0	30	85462	
eg1726_20.csv	100 %	100 %	100 %	9m 47s 319ms	27s 299ms	0	341613	0	
balance-scale_binarized.csv	100 %	100 %	100 %	9m 34s 105ms	9ms	0	2106	0	
bank_binarized.csv	0 %	0 %	100 %	12s 845ms	3m 2s 551ms	0	1	427586	
breast-cancer_binarized.csv	0 %	0 %	0 %	7s 473ms	9ms	9	0	1	
car_binarized.csv	100 %	100 %	100 %	6m 19s 208ms	10ms	0	8001	0	
cmc_binarized.csv	0,01 %	0,01 %	100 %	3s 206ms	10ms	0	2	39506	
CombinedRoles.csv	0 %	0 %	0 %	1h 48m 26s 641ms	127ms	0	0	2780	
flare_binarized.csv	0 %	0 %	0 %	4s 791ms	10ms	0	0	10300	
lymphography_binarized.csv	0 %	0 %	0 %	14s 526ms	9ms	0	0	51939	
monks-1_binarized.csv	100 %	100 %	100 %	10s 46ms	9ms	0	4465	0	
nom5bikesharing_day_cut.csv	0 %	0 %	100 %	6s 647ms	1s 834ms	0	1	61852	
nursery_binarized.csv	0 %	0 %	0 %	4h	9ms				Cancelled, too long.
ord5bikesharing_day_cut.csv	0 %	0 %	100 %	1h 15m 43s 826ms	3s 360ms	0	2	81275	
primary-tumor_binarized.csv	0 %	0 %	0 %	6s 527ms	8ms	0	0	279384	
soybean_binarized.csv	0 %	0 %	100 %	28m 46s 843ms	9ms	0	1	806031	
spect_binarized.csv	100 %	100 %	100 %	4m 14s 459ms	9ms	0	14534	0	
tic-tac-toe_binarized.csv	0 %	0 %	0 %	4h	1s 631ms				Cancelled, too long.
trains_binarized.csv	0 %	0 %	0 %	3m 36s 269ms	16ms	0	0	91	
voting-records_binarized.csv	0,01 %	0,01 %	100 %	2m 49s 57ms	12s 133ms	0	16	264019	
winequality-red_binarized.csv	0,01 %	0,01 %	100 %	5m 43s 436ms	9ms	0	18249	0	

Figure B.11.

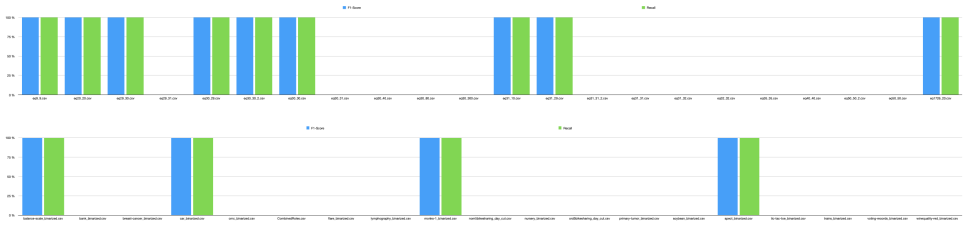

Figure B.12.

Figure B.13.

B.1.5. Self Consistency - Oscar

Example	Precision	Recall	F1-Score	LLM Running Time	Fca4j Running Time	Notes
eg9_9.csv	100 %	100 %	100 %	1s 425ms	12ms	
eg20_20.csv	100 %	100 %	100 %	728ms	30ms	
eg29_30.csv	100 %	100 %	100 %	1s 472ms	93ms	
eg29_31.csv	100 %	100 %	100 %	777ms	90ms	
eg30_29.csv	100 %	100 %	100 %	638ms	97ms	
eg30_30_2.csv	100 %	100 %	100 %	748ms	85ms	
eg30_30.csv	100 %	100 %	100 %	812ms	101ms	
eg30_31.csv	100 %	100 %	100 %	805ms	111ms	
eg30_40.csv	100 %	100 %	100 %	1s 20ms	265ms	
eg30_80.csv	100 %	100 %	100 %	2s 893ms	1s 500ms	
eg30_300.csv	100 %	100 %	100 %	5m 16s 695ms	1m 19s 265ms	
eg31_10.csv	100 %	100 %	100 %	1s 305ms	22ms	
eg31_29.csv	100 %	100 %	100 %	790ms	110ms	
eg31_31_2.csv	100 %	100 %	100 %	796ms	138ms	
eg31_31.csv	100 %	100 %	100 %	794ms	111ms	
eg31_32.csv	100 %	100 %	100 %	836ms	141ms	
eg32_32.csv	100 %	100 %	100 %	756ms	131ms	
eg35_35.csv	100 %	100 %	100 %	915ms	209ms	
eg40_40.csv	100 %	100 %	100 %	1s 265ms	500ms	
eg50_50_2.csv	100 %	100 %	100 %	3s 110ms	2s 611ms	
eg50_50.csv	100 %	100 %	100 %	3s 192ms	2s 672ms	
eg1726_20.csv	100 %	100 %	100 %	17s 879ms	27s 299ms	
balance-scale_binarized.csv	100 %	100 %	100 %	771ms	9ms	
bank_binarized.csv	0 %	0 %	0 %	16h 30m	3m 2s 551ms	Cancelled. No errors.
breast-cancer_binarized.csv	0 %	0 %	0 %	2s 208ms	9ms	
car_binarized.csv	100 %	100 %	100 %	1s 510ms	10ms	
cmc_binarized.csv	100 %	100 %	100 %	6m 45s	10ms	
CombinedRoles.csv	100 %	100 %	100 %	9m 9s	127ms	
flare_binarized.csv	100 %	100 %	100 %	11s 317ms	10ms	
lymphography_binarized.csv	100 %	100 %	100 %	12s 24ms	9ms	
monks-1_binarized.csv	100 %	100 %	100 %	2s 77ms	9ms	
nom5bikesharing_day_cut.csv	100 %	100 %	100 %	3s 234ms	1s 834ms	
nursery_binarized.csv	100 %	100 %	100 %	7m 34s	9ms	
ord5bikesharing_day_cut.csv	100 %	100 %	100 %	25s 280ms	3s 360ms	
primary-tumor_binarized.csv	100 %	100 %	100 %	2m 40s	8ms	
soybean_binarized.csv	0 %	0 %	0 %	1h 21m 32s	9ms	java.lang.OutOfMemoryError: Java heap space
spect_binarized.csv	100 %	100 %	100 %	2s 874ms	9ms	
tic-tac-toe_binarized.csv	100 %	100 %	100 %	16s 865ms	1s 631ms	
trains_binarized.csv	100 %	100 %	100 %	1s 128ms	16ms	
voting-records_binarized.csv	100 %	100 %	100 %	1m 47s	12s 133ms	
winequality-red_binarized.csv	0 %	0 %	0 %	2h 30m	9ms	Takes too long. Cancelled.

Figure B.14.

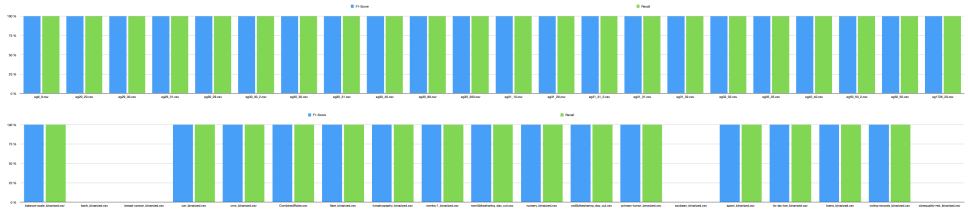

Figure B.15.

Figure B.16.

B.1.6. Step Back - Oscar

Example	F1-Score	Recall	Accuracy	LLM Running Time	Fca4j Running Time	False Positives	True Positives	False Negatives	Notes
eg9_9.csv	0 %	0 %	0 %	427ms	12ms	6	0	34	
eg20_20.csv	0 %	0 %	0 %	497ms	30ms	7	0	573	
eg29_30.csv	0 %	0 %	0 %	600ms	93ms	10	0	3653	
eg29_31.csv	0 %	0 %	0 %	569ms	90ms	9	0	3398	
eg30_29.csv	0 %	0 %	0 %	594ms	97ms	8	0	3857	
eg30_30_2.csv	0 %	0 %	0 %	576ms	85ms	7	0	3132	
eg30_30.csv	0 %	0 %	0 %	1m 26s 6ms	101ms	9	0	4130	
eg30_31.csv	0 %	0 %	0 %	646ms	111ms	9	0	4337	
eg30_40.csv	0 %	0 %	0 %	703ms	265ms	9	0	10355	
eg30_80.csv	0 %	0 %	0 %	1s 24ms	1s 500ms	10	0	51877	
eg30_300.csv	0 %	0 %	0 %	8s 868ms	1m 19s 265ms	10	0	655244	
eg31_10.csv	0 %	0 %	0 %	1s 452ms	22ms	8	0	219	
eg31_29.csv	0 %	0 %	0 %	633ms	110ms	9	0	4227	
eg31_31_2.csv	0 %	0 %	0 %	647ms	138ms	8	0	5400	
eg31_31.csv	0 %	0 %	0 %	598ms	111ms	8	0	4489	
eg31_32.csv	0 %	0 %	0 %	671ms	141ms	9	0	5484	
eg32_32.csv	0 %	0 %	0 %	596ms	131ms	9	0	4981	
eg35_35.csv	0 %	0 %	0 %	695ms	209ms	10	0	8295	
eg40_40.csv	0 %	0 %	0 %	853ms	500ms	10	0	19627	
eg50_50_2.csv	0 %	0 %	0 %	1s 214ms	2s 611ms	10	0	86581	
eg50_50.csv	0 %	0 %	0 %	1s 499ms	2s 672ms	10	0	85492	
eg1726_20.csv	0 %	0 %	0 %	49s 441ms	27s 299ms	15	0	341613	
balance-scale_binarized.csv	0 %	0 %	0 %	2s 940ms	9ms	7	0	2106	
bank_binarized.csv	0 %	0 %	37,5 %	47s	3m 2s 551ms	5	3	427584	
breast-cancer_binarized.csv	0 %	0 %	0 %	2s 422ms	9ms	9	0	9944	
car_binarized.csv	0 %	0 %	0 %	3s 733ms	10ms	8	0	8001	
cmc_binarized.csv	0 %	0 %	0 %	10s 118ms	10ms	10	0	39508	
CombinedRoles.csv	0 %	0 %	0 %	5s 519ms	127ms	10	0	2780	
flare_binarized.csv	0 %	0 %	0 %	6s 980ms	10ms	11	0	10300	
lymphography_binarized.csv	0 %	0 %	0 %	7s 228ms	9ms	7	0	51939	
monks-1_binarized.csv	0 %	0 %	0 %	4s 840ms	9ms	9	0	4465	
nom5bikesharing_day_cut.csv	0 %	0 %	0 %	7s 209ms	1s 834ms	11	0	61853	
nursery_binarized.csv	0 %	0 %	0 %	1m 32s 812ms	9ms	10	0	115201	
ord5bikesharing_day_cut.csv	0 %	0 %	0 %	13s 340ms	3s 360ms	23	0	81277	
primary-tumor_binarized.csv	0 %	0 %	0 %	42s 28ms	8ms	18	0	279384	
soybean_binarized.csv	0 %	0 %	11,11 %	49s 133ms	9ms	8	1	806031	
spect_binarized.csv	0 %	0 %	0 %	29s 499ms	9ms	15	0	14534	
tic-tac-toe_binarized.csv	0 %	0 %	0 %	11s 672ms	1s 631ms	9	0	59505	
trains_binarized.csv	0 %	0 %	0 %	8s 962ms	16ms	7	0	91	
voting-records_binarized.csv	0 %	0 %	0 %	1m 13s 177ms	12s 133ms	8	0	264035	
winequality-red_binarized.csv	0 %	0 %	0 %	8s 280ms	9ms	4	0	18250	

Figure B.17.


LLM Running Time
FreeLi Running Time
1m 40s
10s
1s
100ms
10ms
eq9_9.csv
eq30_20.csv
eq30_30.csv
eq35_31.csv
eq35_39.csv
eq35_39.csv
eq35_30_2.csv
eq35_35.csv
eq30_31.csv
eq30_40.csv
eq35_60.csv
eq30_360.csv
eq31_18.csv
eq31_39.csv
eq31_21_2.csv
eq31_31.csv
eq31_32.csv
eq32_32.csv
eq35_35.csv
eq40_40.csv
eq50_50_2.csv
eq50_50.csv
eq1709_20.csv
LLM Running Time
FreeLi Running Time
1m 40s
10s
1s
100ms
10ms
balance-scale_binarized.csv
breast-cancer_binarized.csv
cmc_binarized.csv
flare_binarized.csv
monks-1_binarized.csv
nursery_binarized.csv
primary-tumor_binarized.csv
spect_binarized.csv
trains_binarized.csv
winequality-red_binarized.csv

Figure B.18.

B.2. Claude 3.7 - Valentino

B.2.1. Zero-Shot - Valentino

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
TSZeroShotRun1	balance-scale_binarized	2105.0	0.0	1.0	1.0	0.9995	0.9998	0.9995
TSZeroShotRun1	ord5bikesharing_day_cut	81273.0	2.0	2.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	lymphography_binarized	51938.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	breast-cancer_binarized	1.0	9942.0	0.0	0.0001	1.0	0.0002	0.0001
TSZeroShotRun1	spect_binarized	14533.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSZeroShotRun1	monks-1_binarized	4464.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSZeroShotRun1	CombinedRoles	2769.0	4.0	7.0	0.9986	0.9975	0.998	0.996
TSZeroShotRun1	flare_binarized	10299.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSZeroShotRun1	nom5bikesharing_day_cut	61852.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	nursery_binarized	115189.0	4.0	8.0	1.0	0.9999	0.9999	0.9999
TSZeroShotRun1	tic-tac-toe_binarized	59504.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	car_binarized	8000.0	0.0	1.0	1.0	0.9999	0.9999	0.9999
TSZeroShotRun1	winequality-red_binarized	18249.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSZeroShotRun1	cmc_binarized	39507.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	trains_binarized	90.0	0.0	1.0	1.0	0.989	0.9945	0.989
TSZeroShotRun1	eg31_10	218.0	0.0	1.0	1.0	0.9954	0.9977	0.9954
TSZeroShotRun1	eg31_29	4226.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSZeroShotRun1	eg30_40	10354.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSZeroShotRun1	eg9_9	33.0	0.0	1.0	1.0	0.9706	0.9851	0.9706
TSZeroShotRun1	eg30_80	51876.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	eg30_30	4129.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSZeroShotRun1	eg20_20	572.0	0.0	1.0	1.0	0.9983	0.9991	0.9983
TSZeroShotRun1	eg80_80	1946952.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	eg30_31	4336.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSZeroShotRun1	eg32_32	4980.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSZeroShotRun1	eg35_35	8294.0	0.0	1.0	1.0	0.9999	0.9999	0.9999
TSZeroShotRun1	eg29_30	3652.0	0.0	1.0	1.0	0.9997	0.9999	0.9997
TSZeroShotRun1	eg29_31	3397.0	0.0	1.0	1.0	0.9997	0.9999	0.9997
TSZeroShotRun1	eg30_30_2	3131.0	0.0	1.0	1.0	0.9997	0.9998	0.9997
TSZeroShotRun1	eg40_40	19626.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSZeroShotRun1	eg50_50	85491.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	eg30_29	3856.0	0.0	1.0	1.0	0.9997	0.9999	0.9997
TSZeroShotRun1	eg31_32	5483.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSZeroShotRun1	eg31_31	4488.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSZeroShotRun1	eg30_300	231672.0	0.0	423572.0	1.0	0.3536	0.5224	0.3536
TSZeroShotRun1	eg50_50_2	86580.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	eg1726_20	341612.0	0.0	1.0	1.0	1.0	1.0	1.0
TSZeroShotRun1	eg31_31_2	5399.0	0.0	1.0	1.0	0.9998	0.9999	0.9998

Figure B.19.

B.2.2. One-Shot - Valentino

TSOneShotRun1_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
TSOneShotRun1	balance-scale_binarized	2105.0	0.0	1.0	1.0	0.9995	0.9998	0.9995
TSOneShotRun1	ord5bikesharing_day_cut	81273.0	2.0	2.0	1.0	1.0	1.0	1.0
TSOneShotRun1	lymphography_binarized	51938.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	breast-cancer_binarized	1.0	9942.0	0.0	0.0001	1.0	0.0002	0.0001
TSOneShotRun1	spect_binarized	14533.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSOneShotRun1	monks-1_binarized	4464.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSOneShotRun1	CombinedRoles	2597.0	3.0	179.0	0.9988	0.9355	0.9661	0.9345
TSOneShotRun1	flare_binarized	10299.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSOneShotRun1	nom5bikesharing_day_cut	61852.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	voting-records_binarized	264034.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	nursery_binarized	115194.0	4.0	3.0	1.0	1.0	1.0	0.9999
TSOneShotRun1	tic-tac-toe_binarized	59504.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	car_binarized	8000.0	0.0	1.0	1.0	0.9999	0.9999	0.9999
TSOneShotRun1	winequality-red_binarized	18059.0	0.0	191.0	1.0	0.9895	0.9947	0.9895
TSOneShotRun1	bank_binarized	1681.0	0.0	425897.0	1.0	0.0039	0.0078	0.0039
TSOneShotRun1	primary-tumor_binarized	279383.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	cmc_binarized	39507.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	chess_binarized	1.0	100881.0	0.0	0.0	1.0	0.0	0.0
TSOneShotRun1	trains_binarized	90.0	0.0	1.0	1.0	0.989	0.9945	0.989
TSOneShotRun1	soybean_binarized	281512.0	0.0	524520.0	1.0	0.3493	0.5177	0.3493
TSOneShotRun1	eg31_10	218.0	0.0	1.0	1.0	0.9954	0.9977	0.9954
TSOneShotRun1	eg31_29	4226.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSOneShotRun1	eg30_40	10354.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSOneShotRun1	eg9_9	33.0	0.0	1.0	1.0	0.9706	0.9851	0.9706
TSOneShotRun1	eg30_80	51876.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	eg30_30	4129.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSOneShotRun1	eg20_20	572.0	0.0	1.0	1.0	0.9983	0.9991	0.9983
TSOneShotRun1	eg80_80	1946952.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	eg30_31	4336.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSOneShotRun1	eg32_32	4980.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSOneShotRun1	eg35_35	8294.0	0.0	1.0	1.0	0.9999	0.9999	0.9999
TSOneShotRun1	eg29_30	3652.0	0.0	1.0	1.0	0.9997	0.9999	0.9997
TSOneShotRun1	eg29_31	3397.0	0.0	1.0	1.0	0.9997	0.9999	0.9997
TSOneShotRun1	eg30_30_2	3131.0	0.0	1.0	1.0	0.9997	0.9998	0.9997
TSOneShotRun1	eg40_40	19626.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSOneShotRun1	eg50_50	85491.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	eg30_29	3856.0	0.0	1.0	1.0	0.9997	0.9999	0.9997
TSOneShotRun1	eg31_32	5483.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSOneShotRun1	eg31_31	4488.0	0.0	1.0	1.0	0.9998	0.9999	0.9998
TSOneShotRun1	eg30_300	167031.0	0.0	488213.0	1.0	0.2549	0.4063	0.2549
TSOneShotRun1	eg50_50_2	86580.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	eg1726_20	341612.0	0.0	1.0	1.0	1.0	1.0	1.0
TSOneShotRun1	eg31_31_2	5399.0	0.0	1.0	1.0	0.9998	0.9999	0.9998

Figure B.20.

B.2.3. Few-Shot - Valentino

TSFewShotsRun1_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
TSFewShotsRun1	balance-scale_binarized	2106.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	ord5bikesharing_day_cut	81274.0	2.0	1.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	lymphography_binarized	51939.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	breast-cancer_binarized	1.0	9943.0	0.0	0.0001	1.0	0.0002	0.0001
TSFewShotsRun1	spect_binarized	14534.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	monks-1_binarized	4465.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	CombinedRoles	2771.0	4.0	5.0	0.9986	0.9982	0.9984	0.9968
TSFewShotsRun1	flare_binarized	10299.0	0.0	1.0	1.0	0.9999	1.0	0.9999
TSFewShotsRun1	nom5bikesharing_day_cut	61853.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	voting-records_binarized	264035.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	nursery_binarized	115191.0	3.0	6.0	1.0	0.9999	1.0	0.9999
TSFewShotsRun1	tic-tac-toe_binarized	59505.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	car_binarized	8001.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	primary-tumor_binarized	279384.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	cmc_binarized	39507.0	0.0	1.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	trains_binarized	91.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg31_10	219.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg31_29	4227.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg30_40	10355.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg9_9	34.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg30_80	51877.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg30_30	4130.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg20_20	573.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg80_80	1946953.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg30_31	4337.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg32_32	4981.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg35_35	8295.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg29_30	3653.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg29_31	3398.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg30_30_2	3132.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg40_40	19627.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg50_50	85492.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg30_29	3857.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg31_32	5484.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg31_31	4489.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg30_300	396935.0	0.0	258309.0	1.0	0.6058	0.7545	0.6058
TSFewShotsRun1	eg50_50_2	86581.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg1726_20	341613.0	0.0	0.0	1.0	1.0	1.0	1.0
TSFewShotsRun1	eg31_31_2	5400.0	0.0	0.0	1.0	1.0	1.0	1.0

Figure B.21.

B.2.4. Chain of Thought - Valentino

Deepseek_V3ChainOfThoughtRun2_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3ChainOfThoughtRun2	bank_binarized	36	0	427542	1.0	0.0001	0.0002	0.0001
V3ChainOfThoughtRun2	CombinedRoles	281	1	2495	0.9965	0.1012	0.1838	0.1012
V3ChainOfThoughtRun2	nom5bikesharing_day_cut	6339	0	55514	1.0	0.1025	0.1859	0.1025
V3ChainOfThoughtRun2	ord5bikesharing_day_cut	9819	1	71456	0.9999	0.1208	0.2156	0.1208
V3ChainOfThoughtRun2	wine_binarized	181	0	18069	1.0	0.0099	0.0196	0.0099
V3ChainOfThoughtRun2	eg31_10	219	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg31_29	4227	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_40	10355	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg9_9	34	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_80	51877	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_30	4130	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg20_20	573	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg80_80	180790	0	1766163	1.0	0.0929	0.1699	0.0929
V3ChainOfThoughtRun2	eg30_31	4337	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg32_32	4981	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg35_35	8295	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg29_30	3653	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg29_31	3398	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_30_2	3132	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg40_40	19627	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg50_50	85492	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_29	3857	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg31_32	5484	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg31_31	4489	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_300	17369	0	637875	1.0	0.0265	0.0516	0.0265
V3ChainOfThoughtRun2	eg50_50_2	86581	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg1726_20	42561	0	299052	1.0	0.1246	0.2216	0.1246
V3ChainOfThoughtRun2	eg31_31_2	5400	0	0	1.0	1.0	1.0	1.0

Figure B.22.

B.2.5. Role Prompting - Valentino

TSRolePromptingRun1_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
TSRolePromptingRun1	balance-scale_binarized	1.0	0.0	2105.0	1.0	0.0005	0.0009	0.0005
TSRolePromptingRun1	ord5bikesharing_day_cut	81275.0	2.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	lymphography_binarized	1.0	0.0	51938.0	1.0	0.0	0.0	0.0
TSRolePromptingRun1	breast-cancer_binarized	1.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	spect_binarized	1.0	0.0	14533.0	1.0	0.0001	0.0001	0.0001
TSRolePromptingRun1	monks-1_binarized	1.0	0.0	4464.0	1.0	0.0002	0.0004	0.0002
TSRolePromptingRun1	CombinedRoles	2628.0	4.0	148.0	0.9985	0.9467	0.9719	0.9453
TSRolePromptingRun1	flare_binarized	0.0	1.0	10300.0	0.0	0.0	0.0	0.0
TSRolePromptingRun1	nom5bikesharing_day_cut	61853.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	voting-records_binarized	1.0	0.0	264034.0	1.0	0.0	0.0	0.0
TSRolePromptingRun1	nursery_binarized	1.0	0.0	115196.0	1.0	0.0	0.0	0.0
TSRolePromptingRun1	tic-tac-toe_binarized	1.0	0.0	59504.0	1.0	0.0	0.0	0.0
TSRolePromptingRun1	car_binarized	1.0	0.0	8000.0	1.0	0.0001	0.0002	0.0001
TSRolePromptingRun1	winequality-red_binarized	1390.0	0.0	16860.0	1.0	0.0762	0.1415	0.0762
TSRolePromptingRun1	bank_binarized	80.0	0.0	427498.0	1.0	0.0002	0.0004	0.0002
TSRolePromptingRun1	primary-tumor_binarized	1.0	0.0	279383.0	1.0	0.0	0.0	0.0
TSRolePromptingRun1	cmc_binarized	1.0	0.0	39507.0	1.0	0.0	0.0001	0.0
TSRolePromptingRun1	chess_binarized	1.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	trains_binarized	0.0	1.0	91.0	0.0	0.0	0.0	0.0
TSRolePromptingRun1	soybean_binarized	1.0	0.0	806031.0	1.0	0.0	0.0	0.0
TSRolePromptingRun1	eg31_10	219.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg31_29	4227.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg30_40	10355.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg9_9	34.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg30_80	51877.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg30_30	4130.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg20_20	573.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg80_80	1946953.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg30_31	4337.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg32_32	4981.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg35_35	8295.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg29_30	3653.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg29_31	3398.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg30_30_2	3132.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg40_40	19627.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg50_50	85492.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg30_29	3857.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg31_32	5484.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg31_31	4489.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg30_300	655244.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg50_50_2	86581.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg1726_20	341613.0	0.0	0.0	1.0	1.0	1.0	1.0
TSRolePromptingRun1	eg31_31_2	5400.0	0.0	0.0	1.0	1.0	1.0	1.0

Figure B.23.

B.2.6. Self Consistency - Valentino

TSSelfConsistencyRun1_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
TSSelfConsistencyRun1	balance-scale_binarized	2106.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	ord5bikesharing_day_cut	81274.0	2.0	1.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	lymphography_binarized	51939.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	breast-cancer_binarized	1.0	9943.0	0.0	0.0001	1.0	0.0002	0.0001
TSSelfConsistencyRun1	spect_binarized	14534.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	monks-1_binarized	4465.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	CombinedRoles	25.0	0.0	2751.0	1.0	0.009	0.0179	0.009
TSSelfConsistencyRun1	flare_binarized	10300.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	nom5bikesharing_day_cut	61853.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	voting-records_binarized	264035.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	nursery_binarized	115193.0	4.0	4.0	1.0	1.0	1.0	0.9999
TSSelfConsistencyRun1	tic-tac-toe_binarized	59505.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	car_binarized	8001.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	winequality-red_binarized	18250.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	bank_binarized	555.0	0.0	427023.0	1.0	0.0013	0.0026	0.0013
TSSelfConsistencyRun1	primary-tumor_binarized	91669.0	0.0	187715.0	1.0	0.3281	0.4941	0.3281
TSSelfConsistencyRun1	cmc_binarized	39508.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	chess_binarized	1.0	80969.0	0.0	0.0	1.0	0.0	0.0
TSSelfConsistencyRun1	trains_binarized	91.0	0.0	0.0	1.0	1.0	1.0	1.0
TSSelfConsistencyRun1	soybean_binarized	10314.0	0.0	795718.0	1.0	0.0128	0.0253	0.0128
TSSelfConsistencyRun1	eg31_10	219	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg31_29	4227	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg30_40	10355	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg30_80	51877	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg35_35	8295	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg29_30	3653	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg29_31	3398	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg30_30_2	3132	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg40_40	19627	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg50_50	85492	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg30_29	3857	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg31_32	5484	0	0	1.0000	1.0000	1.0000	1.0000
TSSelfConsistencyRun1	eg31_31	4489	0	0	1.0000	1.0000	1.0000	1.0000

Figure B.24.

B.2.7. Step Back - Valentino

TSStepbackPromptingRun1_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
TSStepbackPromptingRun1	balance-scale_binarized	1.0	0.0	2105.0	1.0	0.0005	0.0009	0.0005
TSStepbackPromptingRun1	ord5bikesharing_day_cut	1.0	0.0	81274.0	1.0	0.0	0.0	0.0
TSStepbackPromptingRun1	lymphography_binarized	1.0	0.0	51938.0	1.0	0.0	0.0	0.0
TSStepbackPromptingRun1	breast-cancer_binarized	0.0	1.0	1.0	0.0	0.0	0.0	0.0
TSStepbackPromptingRun1	spect_binarized	1.0	0.0	14533.0	1.0	0.0001	0.0001	0.0001
TSStepbackPromptingRun1	monks-1_binarized	1.0	0.0	4464.0	1.0	0.0002	0.0004	0.0002
TSStepbackPromptingRun1	CombinedRoles	1.0	0.0	2775.0	1.0	0.0004	0.0007	0.0004
TSStepbackPromptingRun1	flare_binarized	1.0	0.0	10299.0	1.0	0.0001	0.0002	0.0001
TSStepbackPromptingRun1	nom5bikesharing_day_cut	1.0	0.0	61852.0	1.0	0.0	0.0	0.0
TSStepbackPromptingRun1	voting-records_binarized	1.0	0.0	264034.0	1.0	0.0	0.0	0.0
TSStepbackPromptingRun1	nursery_binarized	1.0	0.0	115196.0	1.0	0.0	0.0	0.0
TSStepbackPromptingRun1	tic-tac-toe_binarized	1.0	0.0	59504.0	1.0	0.0	0.0	0.0
TSStepbackPromptingRun1	car_binarized	1.0	0.0	8000.0	1.0	0.0001	0.0002	0.0001
TSStepbackPromptingRun1	bank_binarized	1.0	0.0	427577.0	1.0	0.0	0.0	0.0
TSStepbackPromptingRun1	primary-tumor_binarized	1.0	0.0	279383.0	1.0	0.0	0.0	0.0
TSStepbackPromptingRun1	cmc_binarized	1.0	0.0	39507.0	1.0	0.0	0.0001	0.0
TSStepbackPromptingRun1	chess_binarized	0.0	1.0	1.0	0.0	0.0	0.0	0.0
TSStepbackPromptingRun1	trains_binarized	91.0	0.0	0.0	1.0	1.0	1.0	1.0
TSStepbackPromptingRun1	soybean_binarized	32.0	2541.0	806000.0	0.0124	0.0	0.0001	0.0
TSStepbackPromptingRun1	eg30_40	4592	0	5763	1.0000	0.4435	0.6144	0.4435
TSStepbackPromptingRun1	eg9_9	34	0	0	1.0000	1.0000	1.0000	1.0000
TSStepbackPromptingRun1	eg30_80	2170	0	30176	1.0000	0.4183	0.5899	0.4183
TSStepbackPromptingRun1	eg30_30	1386	0	2744	1.0000	0.3356	0.5025	0.3356
TSStepbackPromptingRun1	eg20_20	573	0	0	1.0000	1.0000	1.0000	1.0000
TSStepbackPromptingRun1	eg80_80	16	0	1946937	1.0000	0.0000	0.0000	0.0000
TSStepbackPromptingRun1	eg30_31	2350	0	1987	1.0000	0.5418	0.7029	0.5418
TSStepbackPromptingRun1	eg32_32	1	0	4980	1.0000	0.0002	0.0004	0.0002
TSStepbackPromptingRun1	eg35_35	3	0	8292	1.0000	0.0004	0.0007	0.0004
TSStepbackPromptingRun1	eg29_30	2199	0	1454	1.0000	0.6020	0.7515	0.6020
TSStepbackPromptingRun1	eg29_31	1169	0	2229	1.0000	0.3440	0.5119	0.3440
TSStepbackPromptingRun1	eg30_30_2	1756	0	1376	1.0000	0.5607	0.7185	0.5607
TSStepbackPromptingRun1	eg40_40	7	0	19620	1.0000	0.0004	0.0007	0.0004
TSStepbackPromptingRun1	eg50_50	35	0	85457	1.0000	0.0004	0.0008	0.0004
TSStepbackPromptingRun1	eg30_29	2867	0	990	1.0000	0.7433	0.8528	0.7433
TSStepbackPromptingRun1	eg30_300	5966	0	595584	1.0000	0.0911	0.1669	0.0911
TSStepbackPromptingRun1	eg50_50_2	25	0	86556	1.0000	0.0003	0.0006	0.0003
TSStepbackPromptingRun1	eg1726_20	1	0	341612	1.0000	0.0000	0.0000	0.0000

Figure B.25.

B.3. Deepseek V3 - Valentino

B.3.1. Zero-Shot - Valentino

Deepseek_V3ZeroShotRun3_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3ZeroShotRun3	balance-scale_binarized	2101	0	5	1.0	0.9976	0.9988	0.9976
V3ZeroShotRun3	bank_binarized	0	0	427578	0.0	0.0	0.0	0.0
V3ZeroShotRun3	breast-cancer_binarized	0	0	9944	0.0	0.0	0.0	0.0
V3ZeroShotRun3	car_binarized	7998	0	3	1.0	0.9996	0.9998	0.9996
V3ZeroShotRun3	cmc_binarized	12761	0	26747	1.0	0.323	0.4901	0.323
V3ZeroShotRun3	flare_binarized	2139	0	6	1.0	0.9972	0.9986	0.9972
V3ZeroShotRun3	lymphography_binarized	0	0	8134	0.0	0.0	0.0	0.0
V3ZeroShotRun3	monks-1_binarized	26	5	12	0.8387	0.6842	0.7532	0.6047
V3ZeroShotRun3	postoperative_binarized	7	0	2	1.0	0.7778	0.875	0.7778
V3ZeroShotRun3	primary-tumor_binarized	12	32	299	0.2727	0.0386	0.0678	0.0353
V3ZeroShotRun3	shuttle_binarized	30	0	0	1.0	1.0	1.0	1.0
V3ZeroShotRun3	soybean_binarized	0	0	617	0.0	0.0	0.0	0.0
V3ZeroShotRun3	spect_binarized	7	13	9	0.35	0.4375	0.3889	0.25
V3ZeroShotRun3	titanic_binarized	12	90	11	0.1176	0.5217	0.1905	0.1071
V3ZeroShotRun3	vote_binarized	7	0	0	1.0	1.0	1.0	1.0
V3ZeroShotRun3	wine_binarized	5	0	0	1.0	1.0	1.0	1.0
V3ZeroShotRun3	zoo_binarized	8	22	5	0.2667	0.6154	0.3721	0.2353
V3ZeroShotRun3	nom5bikesharing_day_cut	41	9	7	0.82	0.8542	0.8367	0.7368
V3ZeroShotRun3	CombinedRoles	2775	3	2	0.9989	0.9993	0.9991	0.9982
V3ZeroShotRun3	eg31_10	8	0	211	1.0	0.0365	0.0705	0.0365
V3ZeroShotRun3	eg31_29	9	0	4218	1.0	0.0021	0.0042	0.0021
V3ZeroShotRun3	eg30_40	9	0	10346	1.0	0.0009	0.0017	0.0009
V3ZeroShotRun3	eg9_9	6	0	28	1.0	0.1765	0.3	0.1765
V3ZeroShotRun3	eg30_80	10	0	51867	1.0	0.0002	0.0004	0.0002
V3ZeroShotRun3	eg30_30	9	0	4121	1.0	0.0022	0.0043	0.0022
V3ZeroShotRun3	eg20_20	7	0	566	1.0	0.0122	0.0241	0.0122
V3ZeroShotRun3	eg80_80	9	0	1946944	1.0	0.0	0.0	0.0
V3ZeroShotRun3	eg30_31	9	0	4328	1.0	0.0021	0.0041	0.0021
V3ZeroShotRun3	eg32_32	9	0	4972	1.0	0.0018	0.0036	0.0018
V3ZeroShotRun3	eg35_35	10	0	8285	1.0	0.0012	0.0024	0.0012
V3ZeroShotRun3	eg29_30	10	0	3643	1.0	0.0027	0.0055	0.0027
V3ZeroShotRun3	eg29_31	9	0	3389	1.0	0.0026	0.0053	0.0026
V3ZeroShotRun3	eg30_30_2	7	0	3125	1.0	0.0022	0.0045	0.0022
V3ZeroShotRun3	eg40_40	10	0	19617	1.0	0.0005	0.001	0.0005
V3ZeroShotRun3	eg50_50	10	0	85482	1.0	0.0001	0.0002	0.0001
V3ZeroShotRun3	eg30_29	8	0	3849	1.0	0.0021	0.0041	0.0021
V3ZeroShotRun3	eg31_32	9	0	5475	1.0	0.0016	0.0033	0.0016
V3ZeroShotRun3	eg31_31	8	0	4481	1.0	0.0018	0.0036	0.0018
V3ZeroShotRun3	eg30_300	10	0	655234	1.0	0.0	0.0	0.0
V3ZeroShotRun3	eg50_50_2	10	0	86571	1.0	0.0001	0.0002	0.0001
V3ZeroShotRun3	eg1726_20	15	0	341598	1.0	0.0	0.0001	0.0
V3ZeroShotRun3	eg31_31_2	8	0	5392	1.0	0.0015	0.003	0.0015

B.3.2. One-Shot - Valentino

Deepseek_V3OneShotRun3_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3OneShotRun3	balance-scale_binarized	2105	0	1	1.0	0.9995	0.9998	0.9995
V3OneShotRun3	bank_binarized	23370	0	404208	1.0	0.0547	0.1036	0.0547
V3OneShotRun3	breast-cancer_binarized	1	9942	0	0.0001	1.0	0.0002	0.0001
V3OneShotRun3	car_binarized	8000	0	1	1.0	0.9999	0.9999	0.9999
V3OneShotRun3	CombinedRoles	2770	5	6	0.9982	0.9978	0.998	0.996
V3OneShotRun3	flare_binarized	10299	1	1	0.9999	0.9999	0.9999	0.9998
V3OneShotRun3	lymphography_binarized	51938	0	1	1.0	1.0	1.0	1.0
V3OneShotRun3	monks-1_binarized	4464	0	1	1.0	0.9998	0.9999	0.9998
V3OneShotRun3	ord5bikesharing_day_cut	81274	2	1	1.0	1.0	1.0	1.0
V3OneShotRun3	primary-tumor_binarized	171280	0	108104	1.0	0.6131	0.7601	0.6131
V3OneShotRun3	soybean_binarized	178	1638	805854	0.098	0.0002	0.0004	0.0002
V3OneShotRun3	spect_binarized	14533	0	1	1.0	0.9999	1.0	0.9999
V3OneShotRun3	vote_binarized	239303	0	24732	1.0	0.9063	0.9509	0.9063
V3OneShotRun3	wine_binarized	15355	0	2895	1.0	0.8414	0.9139	0.8414
V3OneShotRun3	chess_binarized	1	29327	0	0.0	1.0	0.0001	0.0
V3OneShotRun3	cmc_binarized	35166	0	4342	1.0	0.8901	0.9419	0.8901
V3OneShotRun3	nom5bikesharing_day_cut	61852	0	1	1.0	1.0	1.0	1.0
V3OneShotRun3	nursery_binarized	81087	4	34110	1.0	0.7039	0.8262	0.7039
V3OneShotRun3	tic-tac-toe_binarized	59504	0	1	1.0	1.0	1.0	1.0
V3OneShotRun3	trains_binarized	90	1	1	0.989	0.989	0.989	0.9783
V3OneShotRun3	eg31_10	218	0	1	1.0	0.9954	0.9977	0.9954
V3OneShotRun3	eg31_29	4226	0	1	1.0	0.9998	0.9999	0.9998
V3OneShotRun3	eg30_40	10354	0	1	1.0	0.9999	1.0	0.9999
V3OneShotRun3	eg9_9	33	0	1	1.0	0.9706	0.9851	0.9706
V3OneShotRun3	eg30_80	51876	0	1	1.0	1.0	1.0	1.0
V3OneShotRun3	eg30_30	4129	0	1	1.0	0.9998	0.9999	0.9998
V3OneShotRun3	eg20_20	572	0	1	1.0	0.9983	0.9991	0.9983
V3OneShotRun3	eg80_80	1479903	0	467050	1.0	0.7601	0.8637	0.7601
V3OneShotRun3	eg30_31	4336	0	1	1.0	0.9998	0.9999	0.9998
V3OneShotRun3	eg32_32	4980	0	1	1.0	0.9998	0.9999	0.9998
V3OneShotRun3	eg35_35	8294	0	1	1.0	0.9999	0.9999	0.9999
V3OneShotRun3	eg29_30	3652	0	1	1.0	0.9997	0.9999	0.9997
V3OneShotRun3	eg29_31	3397	0	1	1.0	0.9997	0.9999	0.9997
V3OneShotRun3	eg30_30_2	3131	0	1	1.0	0.9997	0.9998	0.9997
V3OneShotRun3	eg40_40	19626	0	1	1.0	0.9999	1.0	0.9999
V3OneShotRun3	eg50_50	85491	0	1	1.0	1.0	1.0	1.0
V3OneShotRun3	eg30_29	3856	0	1	1.0	0.9997	0.9999	0.9997
V3OneShotRun3	eg31_32	5483	0	1	1.0	0.9998	0.9999	0.9998
V3OneShotRun3	eg31_31	4488	0	1	1.0	0.9998	0.9999	0.9998
V3OneShotRun3	eg30_300	493953	0	161291	1.0	0.7538	0.8596	0.7538
V3OneShotRun3	eg50_50_2	86580	0	1	1.0	1.0	1.0	1.0
V3OneShotRun3	eg1726_20	336631	0	4982	1.0	0.9854	0.9927	0.9854
V3OneShotRun3	eg31_31_2	5399	0	1	1.0	0.9998	0.9999	0.9998

B.3.3. Few-Shot - Valentino

Deepseek_V3FewShotsRun1_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3FewShotsRun1	balance-scale_binarized	31	0	2075	1.0	0.0147	0.029	0.0147
V3FewShotsRun1	bank_binarized	74	0	427504	1.0	0.0002	0.0003	0.0002
V3FewShotsRun1	breast-cancer_binarized	0	108	1	0.0	0.0	0.0	0.0
V3FewShotsRun1	car_binarized	1	0	8000	1.0	0.0001	0.0002	0.0001
V3FewShotsRun1	chess_binarized	0	198	1	0.0	0.0	0.0	0.0
V3FewShotsRun1	cmc_binarized	2	0	39506	1.0	0.0001	0.0001	0.0001
V3FewShotsRun1	CombinedRoles	34	0	2742	1.0	0.0122	0.0242	0.0122
V3FewShotsRun1	flare_binarized	33	0	10267	1.0	0.0032	0.0064	0.0032
V3FewShotsRun1	lymphography_binarized	889	0	51050	1.0	0.0171	0.0337	0.0171
V3FewShotsRun1	monks-1_binarized	46	0	4419	1.0	0.0103	0.0204	0.0103
V3FewShotsRun1	nom5bikesharing_day_cut	230	0	61623	1.0	0.0037	0.0074	0.0037
V3FewShotsRun1	nursery_binarized	1	0	115196	1.0	0.0	0.0	0.0
V3FewShotsRun1	ord5bikesharing_day_cut	225	0	81050	1.0	0.0028	0.0055	0.0028
V3FewShotsRun1	primary-tumor_binarized	630	0	278754	1.0	0.0023	0.0045	0.0023
V3FewShotsRun1	soybean_binarized	295	0	805737	1.0	0.0004	0.0007	0.0004
V3FewShotsRun1	spect_binarized	290	0	14244	1.0	0.02	0.0391	0.02
V3FewShotsRun1	tic-tac-toe_binarized	182	0	59323	1.0	0.0031	0.0061	0.0031
V3FewShotsRun1	trains_binarized	91	0	0	1.0	1.0	1.0	1.0
V3FewShotsRun1	vote_binarized	403	0	263632	1.0	0.0015	0.003	0.0015
V3FewShotsRun1	eg30_40	9094	0	1261	1.0	0.8782	0.9352	0.8782
V3FewShotsRun1	eg9_9	34	0	0	1.0	1.0	1.0	1.0
V3FewShotsRun1	eg30_80	37405	0	14472	1.0	0.721	0.8379	0.721
V3FewShotsRun1	eg30_30	2635	0	1495	1.0	0.638	0.779	0.638
V3FewShotsRun1	eg20_20	573	0	0	1.0	1.0	1.0	1.0
V3FewShotsRun1	eg80_80	3172	0	1943781	1.0	0.0016	0.0033	0.0016
V3FewShotsRun1	eg30_31	3588	0	749	1.0	0.8273	0.9055	0.8273
V3FewShotsRun1	eg32_32	1	0	4980	1.0	0.0002	0.0004	0.0002
V3FewShotsRun1	eg35_35	8	0	8287	1.0	0.001	0.0019	0.001
V3FewShotsRun1	eg29_30	3024	0	629	1.0	0.8278	0.9058	0.8278
V3FewShotsRun1	eg29_31	3222	0	176	1.0	0.9482	0.9734	0.9482
V3FewShotsRun1	eg30_30_2	2600	0	532	1.0	0.8301	0.9072	0.8301
V3FewShotsRun1	eg40_40	98	0	19529	1.0	0.005	0.0099	0.005
V3FewShotsRun1	eg50_50	2528	0	82964	1.0	0.0296	0.0574	0.0296
V3FewShotsRun1	eg30_29	3574	0	283	1.0	0.9266	0.9619	0.9266
V3FewShotsRun1	eg30_300	233110	0	422134	1.0	0.3558	0.5248	0.3558
V3FewShotsRun1	eg50_50_2	1871	0	84710	1.0	0.0216	0.0423	0.0216
V3FewShotsRun1	eg1726_20	461	0	341152	1.0	0.0013	0.0027	0.0013

Figure B.28.

B.3.4. Chain of Thought - Valentino

Deepseek_V3ChainOfThoughtRun2_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3ChainOfThoughtRun2	bank_binarized	36	0	427542	1.0	0.0001	0.0002	0.0001
V3ChainOfThoughtRun2	CombinedRoles	281	1	2495	0.9965	0.1012	0.1838	0.1012
V3ChainOfThoughtRun2	nom5bikesharing_day_cut	6339	0	55514	1.0	0.1025	0.1859	0.1025
V3ChainOfThoughtRun2	ord5bikesharing_day_cut	9819	1	71456	0.9999	0.1208	0.2156	0.1208
V3ChainOfThoughtRun2	wine_binarized	181	0	18069	1.0	0.0099	0.0196	0.0099
V3ChainOfThoughtRun2	eg31_10	219	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg31_29	4227	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_40	10355	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg9_9	34	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_80	51877	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_30	4130	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg20_20	573	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg80_80	180790	0	1766163	1.0	0.0929	0.1699	0.0929
V3ChainOfThoughtRun2	eg30_31	4337	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg32_32	4981	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg35_35	8295	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg29_30	3653	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg29_31	3398	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_30_2	3132	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg40_40	19627	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg50_50	85492	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_29	3857	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg31_32	5484	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg31_31	4489	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg30_300	17369	0	637875	1.0	0.0265	0.0516	0.0265
V3ChainOfThoughtRun2	eg50_50_2	86581	0	0	1.0	1.0	1.0	1.0
V3ChainOfThoughtRun2	eg1726_20	42561	0	299052	1.0	0.1246	0.2216	0.1246
V3ChainOfThoughtRun2	eg31_31_2	5400	0	0	1.0	1.0	1.0	1.0

Figure B.29.

B.3.5. Tree of Thought - Valentino

Deepseek_V3TreeOfThoughtRun2_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3TreeOfThoughtRun2	balance-scale_binarized	2104	0	2	1.0	0.9991	0.9995	0.9991
V3TreeOfThoughtRun2	bank_binarized	0	0	427578	0.0	0.0	0.0	0.0
V3TreeOfThoughtRun2	breast-cancer_binarized	0	0	9944	0.0	0.0	0.0	0.0
V3TreeOfThoughtRun2	car_binarized	7989	0	12	1.0	0.9985	0.9992	0.9985
V3TreeOfThoughtRun2	cmc_binarized	12761	0	26747	1.0	0.323	0.4901	0.323
V3TreeOfThoughtRun2	flare_binarized	2133	0	12	1.0	0.9944	0.9972	0.9944
V3TreeOfThoughtRun2	lymphography_binarized	0	0	8134	0.0	0.0	0.0	0.0
V3TreeOfThoughtRun2	monks-1_binarized	34	5	4	0.8718	0.8947	0.8831	0.7674
V3TreeOfThoughtRun2	postoperative_binarized	9	0	0	1.0	1.0	1.0	1.0
V3TreeOfThoughtRun2	primary-tumor_binarized	12	32	299	0.2727	0.0386	0.0678	0.0353
V3TreeOfThoughtRun2	shuttle_binarized	30	0	0	1.0	1.0	1.0	1.0
V3TreeOfThoughtRun2	soybean_binarized	14	603	0	0.0227	1.0	0.0444	0.0227
V3TreeOfThoughtRun2	spect_binarized	0	0	28	0.0	0.0	0.0	0.0
V3TreeOfThoughtRun2	titanic_binarized	9	72	14	0.1111	0.3913	0.1731	0.0938
V3TreeOfThoughtRun2	vote_binarized	7	0	0	1.0	1.0	1.0	1.0
V3TreeOfThoughtRun2	wine_binarized	5	0	0	1.0	1.0	1.0	1.0
V3TreeOfThoughtRun2	zoo_binarized	8	18	5	0.3077	0.6154	0.4103	0.2581
V3TreeOfThoughtRun2	nom5bikesharing_day_cut	41	8	8	0.8367	0.8367	0.8367	0.7193
V3TreeOfThoughtRun2	CombinedRoles	2776	2	2	0.9993	0.9993	0.9993	0.9986
V3TreeOfThoughtRun2	eg31_10	32	0	187	1.0	0.1461	0.255	0.1461
V3TreeOfThoughtRun2	eg31_29	32	0	4195	1.0	0.0076	0.015	0.0076
V3TreeOfThoughtRun2	eg30_40	31	0	10324	1.0	0.003	0.006	0.003
V3TreeOfThoughtRun2	eg9_9	10	0	24	1.0	0.2941	0.4545	0.2941
V3TreeOfThoughtRun2	eg30_80	31	0	51846	1.0	0.0006	0.0012	0.0006
V3TreeOfThoughtRun2	eg30_30	31	0	4099	1.0	0.0075	0.0149	0.0075
V3TreeOfThoughtRun2	eg20_20	21	0	552	1.0	0.0366	0.0707	0.0366
V3TreeOfThoughtRun2	eg80_80	81	0	1946872	1.0	0.0	0.0001	0.0
V3TreeOfThoughtRun2	eg30_31	31	0	4306	1.0	0.0071	0.0142	0.0071
V3TreeOfThoughtRun2	eg32_32	33	0	4948	1.0	0.0066	0.0132	0.0066
V3TreeOfThoughtRun2	eg35_35	36	0	8259	1.0	0.0043	0.0086	0.0043
V3TreeOfThoughtRun2	eg29_30	30	0	3623	1.0	0.0082	0.0163	0.0082
V3TreeOfThoughtRun2	eg29_31	30	0	3368	1.0	0.0088	0.0175	0.0088
V3TreeOfThoughtRun2	eg30_30_2	31	0	3101	1.0	0.0099	0.0196	0.0099
V3TreeOfThoughtRun2	eg40_40	41	0	19586	1.0	0.0021	0.0042	0.0021
V3TreeOfThoughtRun2	eg50_50	51	0	85441	1.0	0.0006	0.0012	0.0006
V3TreeOfThoughtRun2	eg30_29	31	0	3826	1.0	0.008	0.0159	0.008
V3TreeOfThoughtRun2	eg31_32	32	0	5452	1.0	0.0058	0.0116	0.0058
V3TreeOfThoughtRun2	eg31_31	32	0	4457	1.0	0.0071	0.0142	0.0071
V3TreeOfThoughtRun2	eg30_300	31	0	655213	1.0	0.0	0.0001	0.0
V3TreeOfThoughtRun2	eg50_50_2	51	0	86530	1.0	0.0006	0.0012	0.0006
V3TreeOfThoughtRun2	eg1726_20	1727	0	339886	1.0	0.0051	0.0101	0.0051
V3TreeOfThoughtRun2	eg31_31_2	32	0	5368	1.0	0.0059	0.0118	0.0059

Figure B.30.

B.3.6. Role Prompting - Valentino

Deepseek_V3RolePromptingRun2_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3RolePromptingRun2	CombinedRoles	2773	4	3	0.9986	0.9989	0.9987	0.9975
V3RolePromptingRun2	balance-scale_binarized	2106	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	bank_binarized	84	0	427494	1.0	0.0002	0.0004	0.0002
V3RolePromptingRun2	breast-cancer_binarized	1	9943	0	0.0001	1.0	0.0002	0.0001
V3RolePromptingRun2	car_binarized	8001	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	cmc_binarized	23473	0	16035	1.0	0.5941	0.7454	0.5941
V3RolePromptingRun2	flare_binarized	2145	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	lymphography_binarized	5	8129	0	0.0006	1.0	0.0012	0.0006
V3RolePromptingRun2	monks-1_binarized	37	6	0	0.8605	1.0	0.925	0.8605
V3RolePromptingRun2	nom5bikesharing_day_cut	45	0	12	1.0	0.7895	0.8824	0.7895
V3RolePromptingRun2	postoperative_binarized	9	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	primary-tumor_binarized	36	275	0	0.1158	1.0	0.2067	0.1158
V3RolePromptingRun2	shuttle_binarized	30	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	soybean_binarized	12	605	0	0.0195	1.0	0.0381	0.0195
V3RolePromptingRun2	spect_binarized	6	12	10	0.3333	0.375	0.3529	0.2143
V3RolePromptingRun2	titanic_binarized	11	98	12	0.1009	0.4783	0.1667	0.0909
V3RolePromptingRun2	vote_binarized	7	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	wine_binarized	5	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	zoo_binarized	13	22	0	0.3714	1.0	0.5417	0.3714
V3RolePromptingRun2	eg31_10	219	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg31_29	4227	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg30_40	10355	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg9_9	34	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg30_80	51877	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg30_30	4130	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg20_20	573	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg80_80	752704	0	1194249	1.0	0.3866	0.5576	0.3866
V3RolePromptingRun2	eg30_31	4337	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg32_32	4981	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg35_35	8295	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg29_30	3653	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg29_31	3398	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg30_30_2	3132	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg40_40	19627	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg50_50	85492	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg30_29	3857	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg31_32	5484	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg31_31	4489	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg30_300	59633	0	595611	1.0	0.091	0.1668	0.091
V3RolePromptingRun2	eg50_50_2	86581	0	0	1.0	1.0	1.0	1.0
V3RolePromptingRun2	eg1726_20	147513	0	194100	1.0	0.4318	0.6032	0.4318
V3RolePromptingRun2	eg31_31_2	5400	0	0	1.0	1.0	1.0	1.0

Figure B.31.

B.3.7. Self Consistency - Valentino

Deepseek_V3SelfConsistencyRun1_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3SelfConsistencyRun1	balance-scale_binarized	2096	0	10	1.0	0.9953	0.9976	0.9953
V3SelfConsistencyRun1	bank_binarized	0	0	427578	0.0	0.0	0.0	0.0
V3SelfConsistencyRun1	breast-cancer_binarized	0	0	9944	0.0	0.0	0.0	0.0
V3SelfConsistencyRun1	car_binarized	7999	0	2	1.0	0.9998	0.9999	0.9998
V3SelfConsistencyRun1	cmc_binarized	12761	0	26747	1.0	0.323	0.4901	0.323
V3SelfConsistencyRun1	flare_binarized	2139	0	6	1.0	0.9972	0.9986	0.9972
V3SelfConsistencyRun1	lymphography_binarized	0	0	8134	0.0	0.0	0.0	0.0
V3SelfConsistencyRun1	monks-1_binarized	21	4	18	0.84	0.5385	0.6563	0.5814
V3SelfConsistencyRun1	postoperative_binarized	7	0	2	1.0	0.7778	0.875	0.7778
V3SelfConsistencyRun1	primary-tumor_binarized	12	32	299	0.2727	0.0386	0.0678	0.0353
V3SelfConsistencyRun1	shuttle_binarized	30	0	0	1.0	1.0	1.0	1.0
V3SelfConsistencyRun1	soybean_binarized	0	0	617	0.0	0.0	0.0	0.0
V3SelfConsistencyRun1	spect_binarized	0	0	28	0.0	0.0	0.0	0.0
V3SelfConsistencyRun1	titanic_binarized	12	73	11	0.1412	0.5217	0.2222	0.125
V3SelfConsistencyRun1	vote_binarized	7	0	0	1.0	1.0	1.0	1.0
V3SelfConsistencyRun1	wine_binarized	5	0	0	1.0	1.0	1.0	1.0
V3SelfConsistencyRun1	zoo_binarized	11	22	2	0.3333	0.8462	0.4783	0.3235
V3SelfConsistencyRun1	nom5bikesharing_day_cut	34	1	22	0.9714	0.6071	0.7458	0.5965
V3SelfConsistencyRun1	CombinedRoles	2774	4	2	0.9986	0.9993	0.9989	0.9978
V3SelfConsistencyRun1	eg31_10	8	0	211	1.0	0.0365	0.0705	0.0365
V3SelfConsistencyRun1	eg31_29	9	0	4218	1.0	0.0021	0.0042	0.0021
V3SelfConsistencyRun1	eg30_40	9	0	10346	1.0	0.0009	0.0017	0.0009
V3SelfConsistencyRun1	eg9_9	6	0	28	1.0	0.1765	0.3	0.1765
V3SelfConsistencyRun1	eg30_80	10	0	51867	1.0	0.0002	0.0004	0.0002
V3SelfConsistencyRun1	eg30_30	8	0	4122	1.0	0.0019	0.0039	0.0019
V3SelfConsistencyRun1	eg20_20	7	0	566	1.0	0.0122	0.0241	0.0122
V3SelfConsistencyRun1	eg80_80	9	0	1946944	1.0	0.0	0.0	0.0
V3SelfConsistencyRun1	eg30_31	7	0	4330	1.0	0.0016	0.0032	0.0016
V3SelfConsistencyRun1	eg32_32	9	0	4972	1.0	0.0018	0.0036	0.0018
V3SelfConsistencyRun1	eg35_35	9	0	8286	1.0	0.0011	0.0022	0.0011
V3SelfConsistencyRun1	eg29_30	12	0	3641	1.0	0.0033	0.0065	0.0033
V3SelfConsistencyRun1	eg29_31	9	0	3389	1.0	0.0026	0.0053	0.0026
V3SelfConsistencyRun1	eg30_30_2	7	0	3125	1.0	0.0022	0.0045	0.0022
V3SelfConsistencyRun1	eg40_40	9	0	19618	1.0	0.0005	0.0009	0.0005
V3SelfConsistencyRun1	eg50_50	10	0	85482	1.0	0.0001	0.0002	0.0001
V3SelfConsistencyRun1	eg30_29	8	0	3849	1.0	0.0021	0.0041	0.0021
V3SelfConsistencyRun1	eg31_32	9	0	5475	1.0	0.0016	0.0033	0.0016
V3SelfConsistencyRun1	eg31_31	8	0	4481	1.0	0.0018	0.0036	0.0018
V3SelfConsistencyRun1	eg30_300	10	0	655234	1.0	0.0	0.0	0.0
V3SelfConsistencyRun1	eg50_50_2	10	0	86571	1.0	0.0001	0.0002	0.0001
V3SelfConsistencyRun1	eg1726_20	13	0	341600	1.0	0.0	0.0001	0.0
V3SelfConsistencyRun1	eg31_31_2	8	0	5392	1.0	0.0015	0.003	0.0015

Figure B.32.

B.3.8. Step Back - Valentino

Deepseek_V3StepBackPromptingRun1_combined

Technique	File	TP	FP	FN	Precision	Recall	F1	Accuracy
V3StepBackPromptingRun1	balance-scale_binarized	2100	0	6	1.0	0.9971	0.9986	0.9971
V3StepBackPromptingRun1	bank_binarized	0	0	427578	0.0	0.0	0.0	0.0
V3StepBackPromptingRun1	breast-cancer_binarized	0	0	9944	0.0	0.0	0.0	0.0
V3StepBackPromptingRun1	car_binarized	7999	0	2	1.0	0.9998	0.9999	0.9998
V3StepBackPromptingRun1	cmc_binarized	12761	0	26747	1.0	0.323	0.4901	0.323
V3StepBackPromptingRun1	flare_binarized	2141	0	4	1.0	0.9981	0.9991	0.9981
V3StepBackPromptingRun1	lymphography_binarized	0	0	8134	0.0	0.0	0.0	0.0
V3StepBackPromptingRun1	monks-1_binarized	37	6	0	0.8605	1.0	0.925	0.8605
V3StepBackPromptingRun1	postoperative_binarized	9	0	0	1.0	1.0	1.0	1.0
V3StepBackPromptingRun1	primary-tumor_binarized	36	275	0	0.1158	1.0	0.2067	0.1158
V3StepBackPromptingRun1	shuttle_binarized	30	0	0	1.0	1.0	1.0	1.0
V3StepBackPromptingRun1	soybean_binarized	0	0	617	0.0	0.0	0.0	0.0
V3StepBackPromptingRun1	spect_binarized	11	17	5	0.3929	0.6875	0.5	0.3929
V3StepBackPromptingRun1	titanic_binarized	16	99	7	0.1391	0.6957	0.2319	0.1391
V3StepBackPromptingRun1	CombinedRoles	2776	2	2	0.9993	0.9993	0.9993	0.9986

Figure B.33.

B.4. TEST RESULTS Arame and Yazmín prompts

Figure B.34.: Enter Caption

Step-back prompting

Step-back prompting - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	83.33	83.33	73.33	83.33
eg2_9.csv	91.67	91.67	86.67	91.67
eg5_5.csv	69.40	87.50	62.22	73.61
eg8_8.csv	63.75	50.00	38.46	55.41
eg15_15.csv	45.01	11.97	8.77	16.07

Step-back prompting - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	93.33	75.00	66.33	76.30
eg5_5.csv	67.33	91.67	61.05	73.20
eg8_8.csv	86.44	58.34	54.17	68.66
eg15_15.csv	24.80	8.97	6.68	12.29

Step-back prompting - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	88.89	100.00	88.89	93.33
eg2_9.csv	41.67	50.00	40.00	44.44
eg5_5.csv	95.83	91.67	88.43	93.61
eg8_8.csv	100.00	98.61	98.61	99.29
eg15_15.csv	100.00	99.57	99.57	99.79

Step-back prompting - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	66.67	83.33	56.67	72.22
eg2_9.csv	38.96	41.67	24.52	35.86
eg5_5.csv	42.70	33.33	23.45	34.21
eg8_8.csv	90.00	47.22	46.14	55.87
eg15_15.csv	58.94	1.49	1.32	2.59

Step-back prompting - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	100.00	100.00	100.00	100.00
eg5_5.csv	70.37	70.83	58.75	70.59
eg8_8.csv	100.00	79.17	79.17	59.17
eg15_15.csv	86.22	10.26	8.70	14.39

Chain of thought

Chain of thought - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	100.00	100.00	100.00	100.00
eg5_5.csv	75.00	75.00	60.00	75.00
eg8_8.csv	77.78	29.17	26.92	42.42
eg15_15.csv	3.92	2.14	1.41	2.77

Chain of thought - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	93.33	100.00	100.00	96.33
eg5_5.csv	96.63	100.00	100.00	99.00
eg8_8.csv	78.34	41.67	37.97	53.93
eg15_15.csv	3.76	2.94	1.91	3.03

Chain of thought - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	87.50	87.50	80.00	87.50
eg5_5.csv	59.52	50.00	41.67	54.29
eg8_8.csv	86.84	79.17	74.14	82.56
eg15_15.csv	100.00	99.68	99.68	99.84

Chain of thought - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	75.00	75.00	60.00	75.00
eg5_5.csv	100.00	100.00	100.00	100.00
eg8_8.csv	91.67	91.67	84.62	91.67
eg15_15.csv	7.79	13.79	5.19	9.45

Chain of thought - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	100.00	100.00	100.00	100.00
eg5_5.csv	100.00	100.00	100.00	100.00
eg8_8.csv	57.36	43.75	33.04	49.56
eg15_15.csv	50.00	50.00	50.00	50.00

Self consistency

Self consistency - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	100.00	100.00	100.00	100.00
eg5_5.csv	66.67	100.00	66.67	80.00
eg8_8.csv	100.00	95.83	95.83	97.87
eg15_15.csv	20.00	1.28	1.22	2.41

Self consistency - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	100.00	100.00	100.00	100.00
eg5_5.csv	100.00	100.00	100.00	100.00
eg8_8.csv	94.74	75.00	72.00	83.72
eg15_15.csv	60.00	1.92	1.90	3.73

Self consistency - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	80.00	100.00	80.00	88.89
eg2_9.csv	100.00	100.00	100.00	100.00
eg5_5.csv	100.00	100.00	100.00	100.00
eg8_8.csv	100.00	100.00	100.00	100.00
eg15_15.csv	100.00	100.00	100.00	100.00

Self consistency - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	80.00	100.00	80.00	88.89
eg5_5.csv	100.00	100.00	100.00	100.00
eg8_8.csv	96.00	100.00	96.00	97.96
eg15_15.csv	34.15	8.97	7.65	14.21

Self consistency - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	100.00	100.00	100.00	100.00
eg5_5.csv	100.00	100.00	100.00	100.00
eg8_8.csv	100.00	100.00	100.00	100.00
eg15_15.csv	100.00	100.00	100.00	100.00

Tree of thoughts

Tree of thoughts - ChatGPT-4o

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	83.33	75.00	65.00	78.57
eg2_9.csv	90.00	100.00	90.00	94.45
eg5_5.csv	58.33	87.50	53.85	70.00
eg8_8.csv	48.28	58.33	35.90	52.83
eg15_15.csv	11.11	1.28	1.16	2.30

Tree of thoughts - DeepSeek-V3

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	86.67	100.00	86.67	92.59
eg2_9.csv	90.00	100.00	90.00	94.45
eg5_5.csv	72.73	100.00	72.73	84.21
eg8_8.csv	82.35	58.33	51.85	68.29
eg15_15.csv	65.52	12.18	11.45	20.54

Tree of thoughts - Claude 4 Sonnet

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	75.00	75.00	60.00	75.00
eg2_9.csv	100.00	87.50	87.50	92.86
eg5_5.csv	100.00	100.00	100.00	100.00
eg8_8.csv	29.79	58.33	24.56	39.44
eg15_15.csv	100.00	0.64	0.64	1.27

Tree of thoughts - Mistral Large

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	77.50	87.50	70.00	81.95
eg2_9.csv	40.00	50.00	40.00	44.45
eg5_5.csv	42.86	37.50	25.00	40.00
eg8_8.csv	15.79	12.50	7.50	13.95
eg15_15.csv	0.93	0.64	0.38	0.76

Tree of thoughts - Gemini 2.5 Pro

Example	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
eg2_3.csv	100.00	100.00	100.00	100.00
eg2_9.csv	100.00	100.00	100.00	100.00
eg5_5.csv	100.00	100.00	100.00	100.00
eg8_8.csv	100.00	100.00	100.00	100.00
eg15_15.csv	11.11	1.28	1.16	2.30

Figure B.35.: Enter Caption

B.5. Graph results Arame and Yazmin

B.6. zero shot

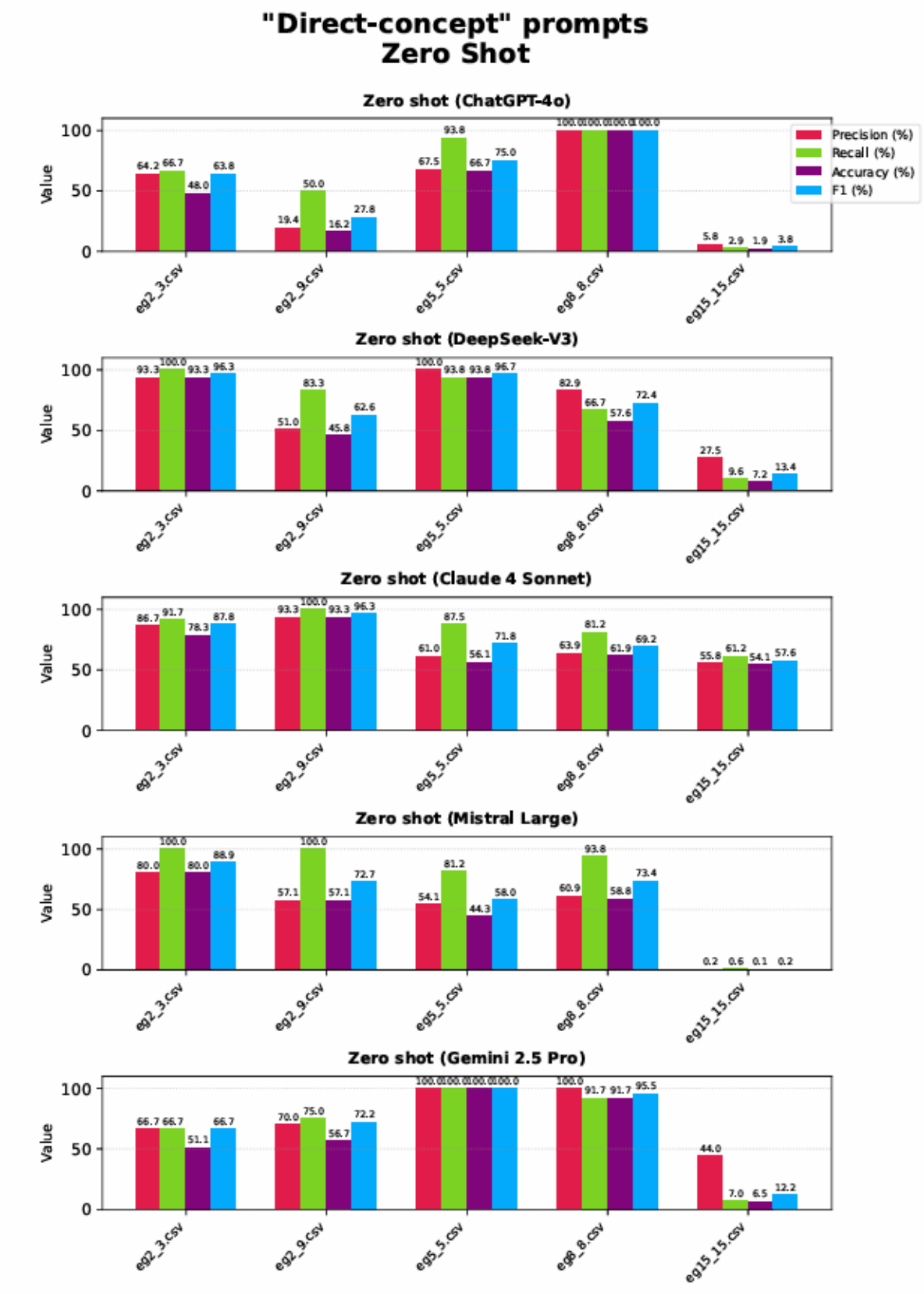

Figure B.36.: Enter Caption

B.7. few shot

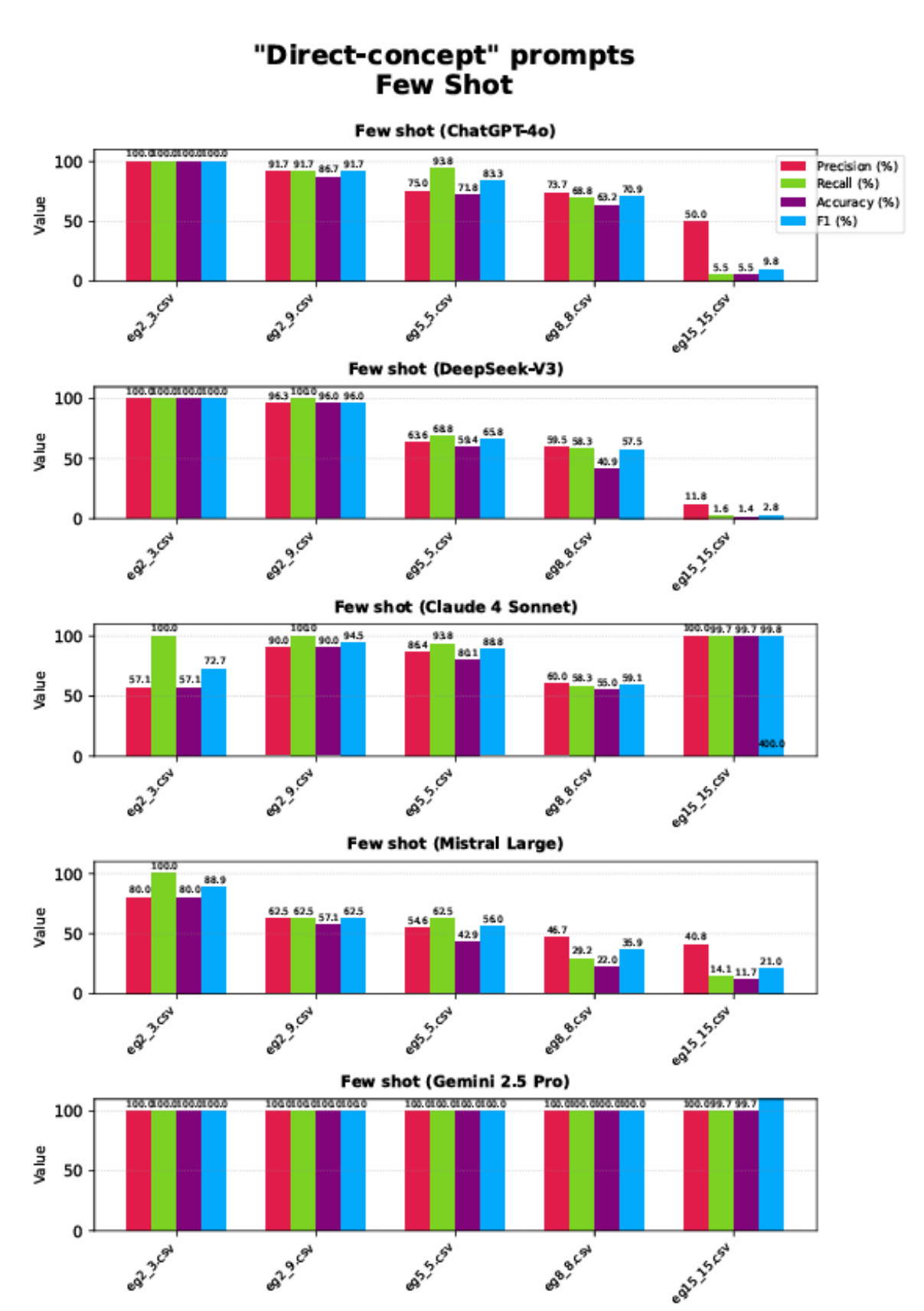

Figure B.37.: Enter Caption

B.8. Role Prompting

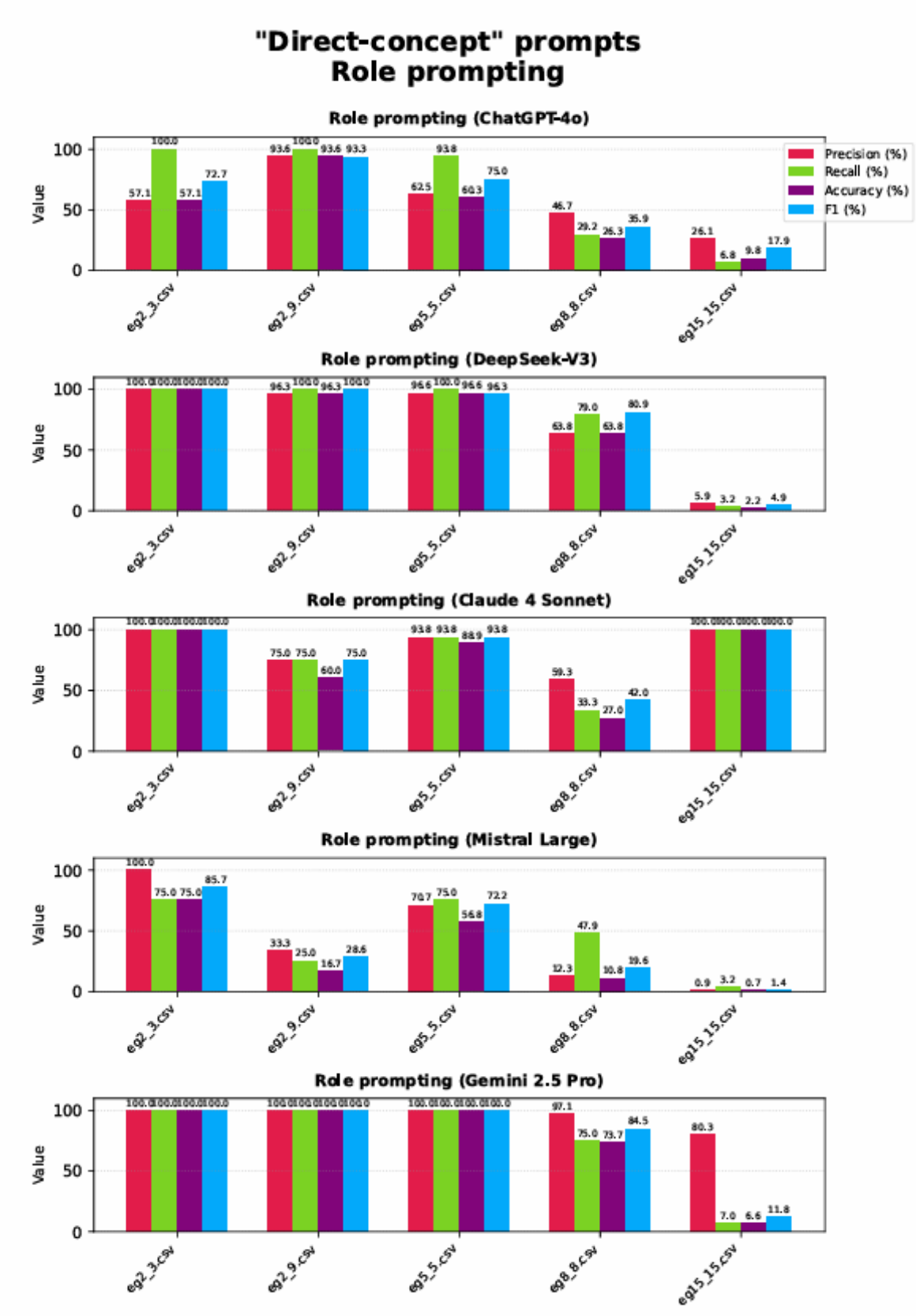

Figure B.38.: Enter Caption

B.9. Contextual Prompting

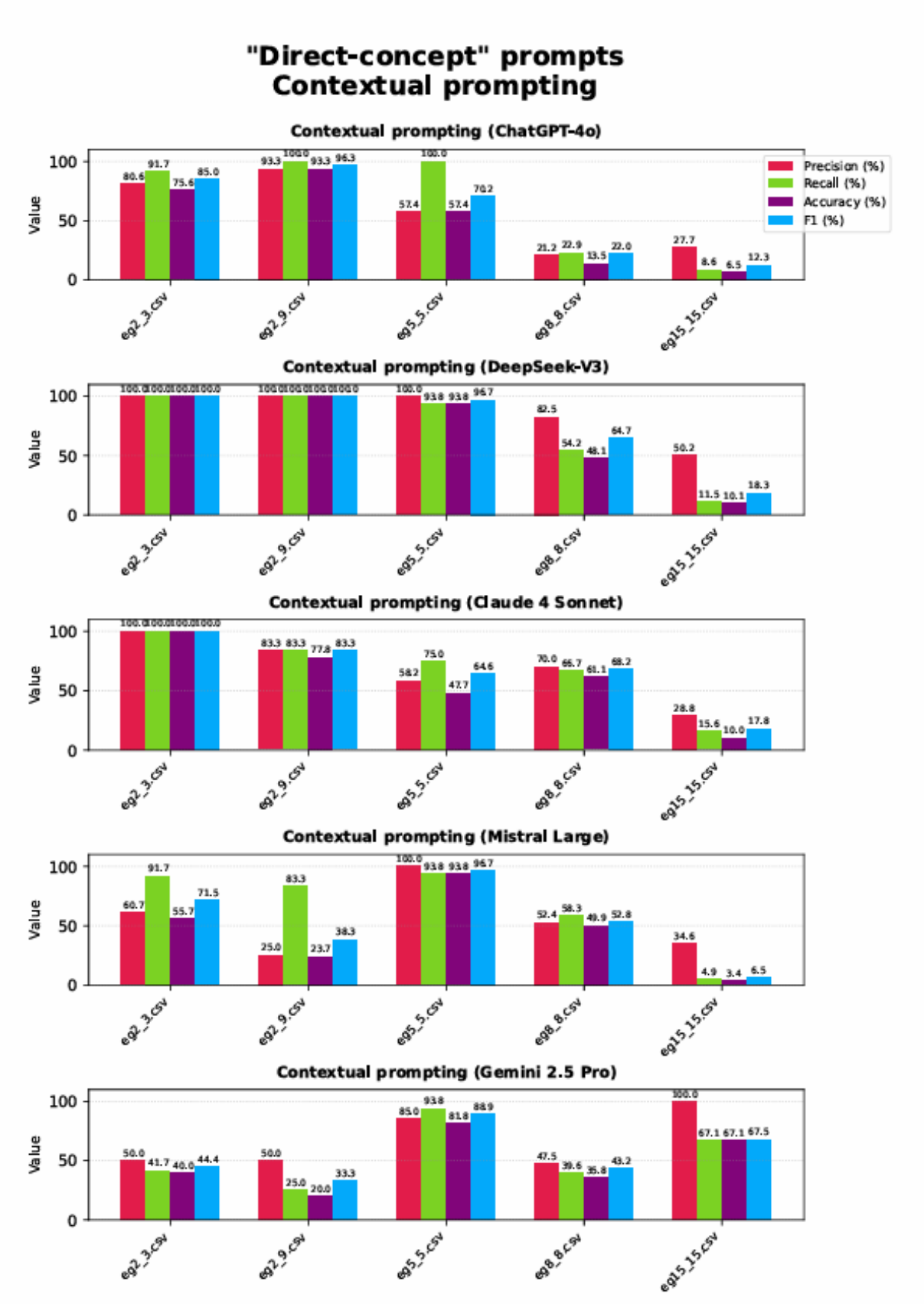

Figure B.39.: Enter Caption

B.10. Step-back prompting

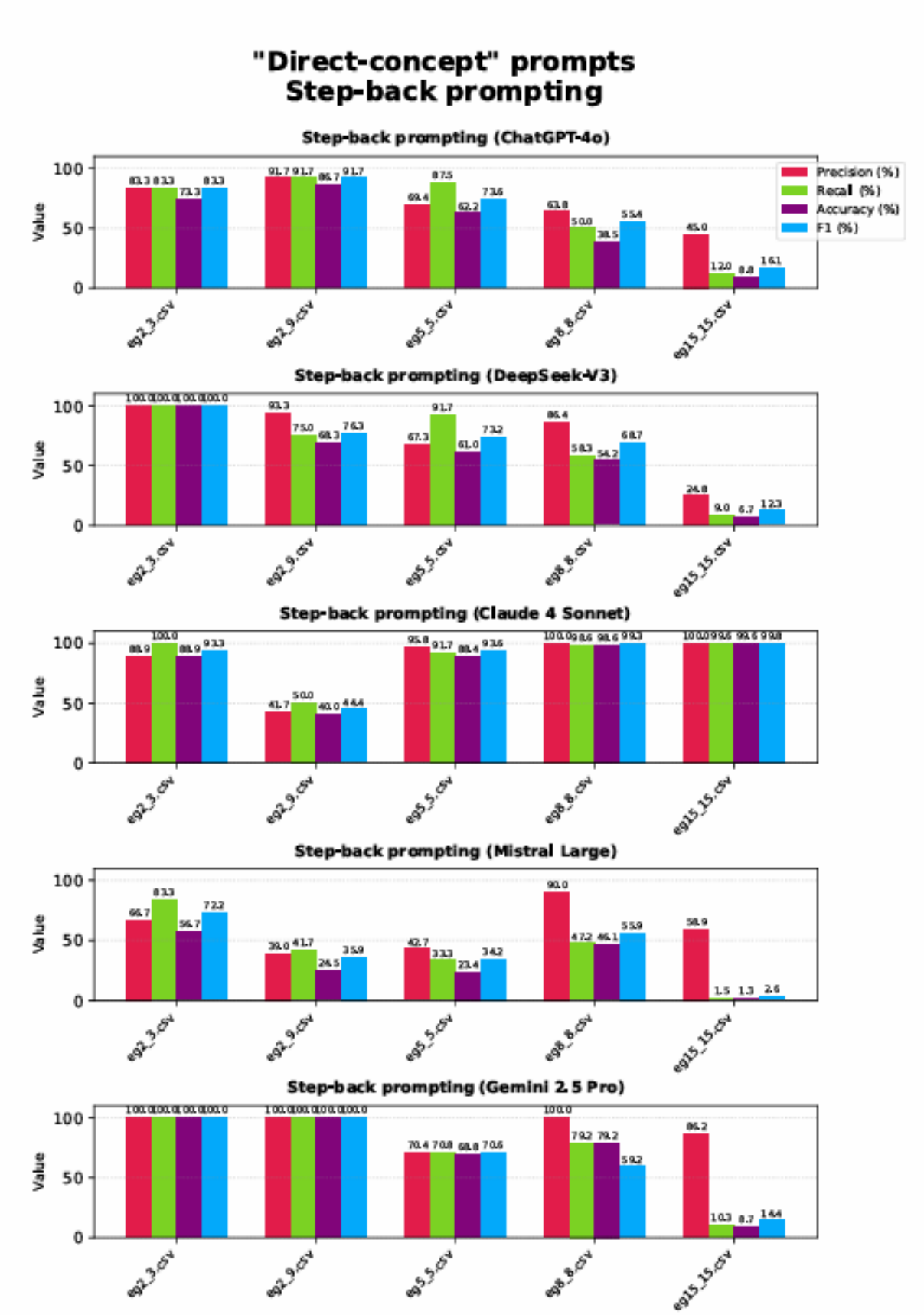

Figure B.40.: Enter Caption

B.11. Chain Of Thoughts

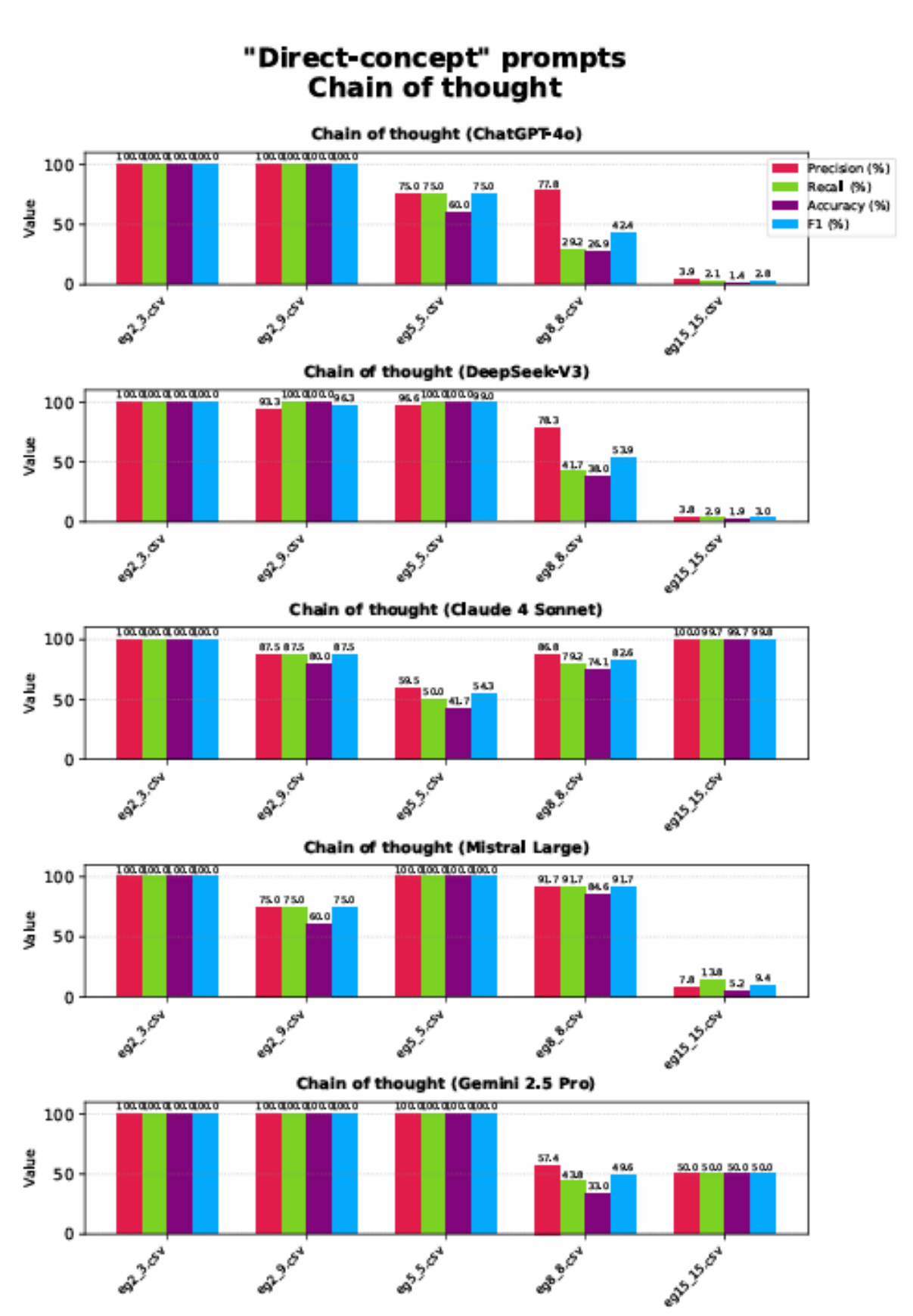

Figure B.41.: Enter Caption

B.12. Self consistency

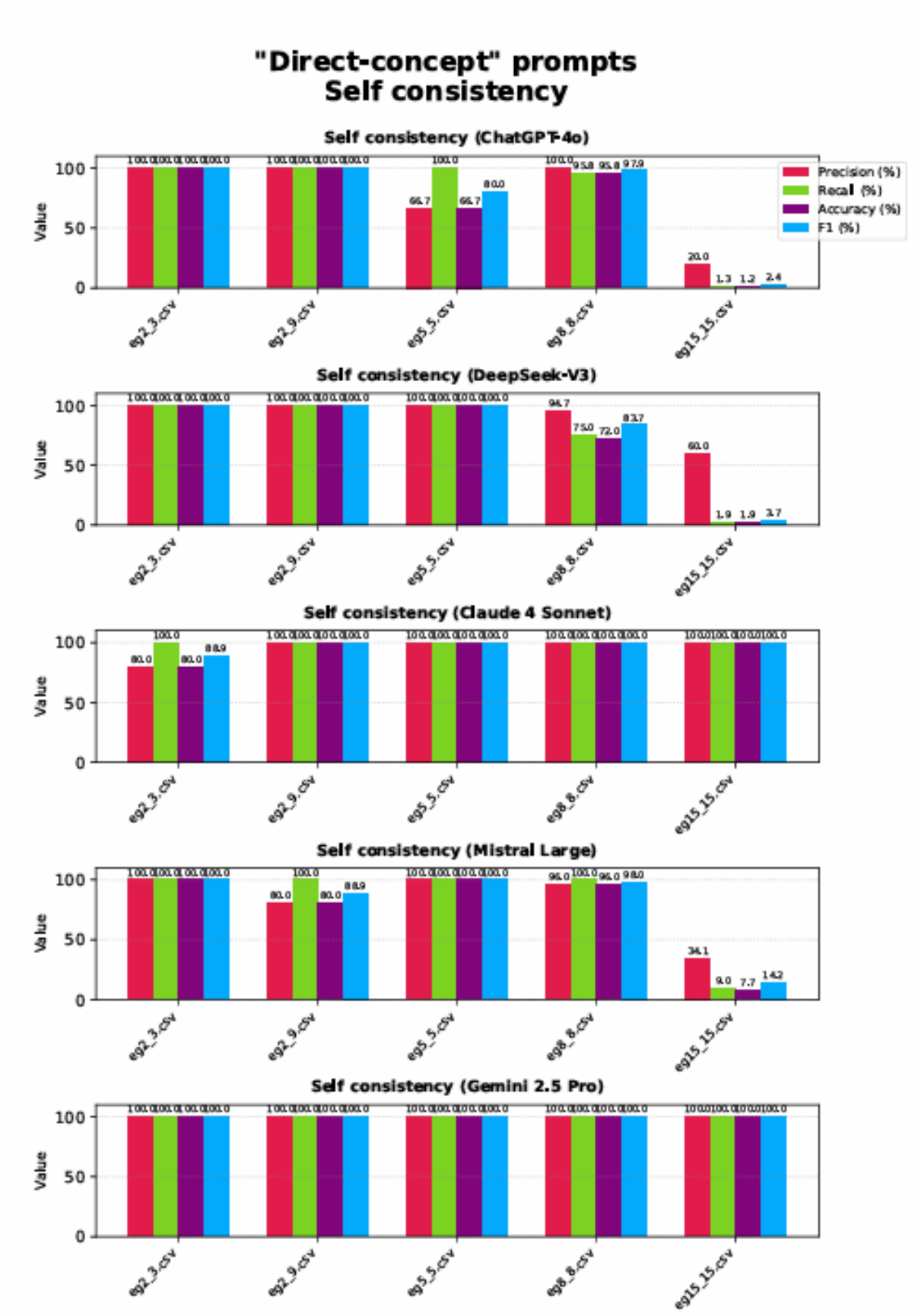

Figure B.42.: Enter Caption

B.13. Tree of Thoughts

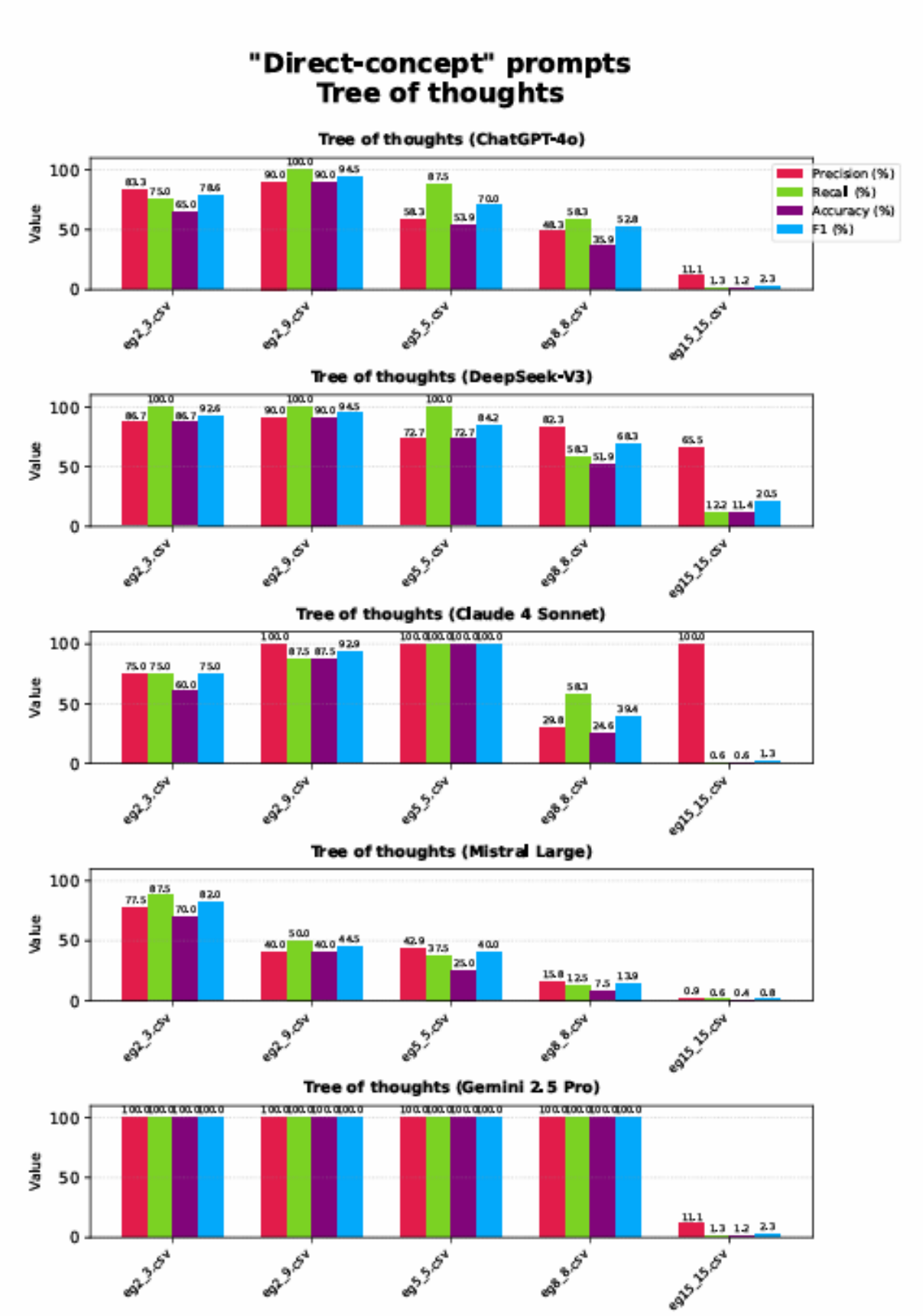

Figure B.43.: Enter Caption

Bibliography

[And11] Simon Andrews. In-close2, a high performance formal concept miner. In Simon Andrews, Simon Polovina, Richard Hill, and Babak Akhgar, editors, Conceptual Structures for Discovering Knowledge - 19th International Conference on Conceptual Structures, ICCS 2011, Derby, UK, July 25-29, 2011. Proceedings, volume 6828 of Lecture Notes in Computer Science, pages 50–62. Springer, 2011.

[BV10] Radim Belohlávek and Vilém Vychodil. Discovery of optimal factors in binary data via a novel method of matrix decomposition. J. Comput. Syst. Sci., 76(1):3–20, 2010.

[CR20] Claudio Carpineto and Giovanni Romano. An experimental study of automatic detection and measurement of counterfeit in brand search results. ACM Trans. Web, 14(2):6:1–6:35, 2020.

[DG19] Dheeru Dua and Casey Graff. Uci machine learning repository, 2019. University of California, Irvine, School of Information and Computer Sciences.

[Gan10] Bernhard Ganter. Two basic algorithms in concept analysis. In Léonard Kwuida and Baris Sertkaya, editors, Formal Concept Analysis, 8th International Conference, ICFCA 2010, Agadir, Morocco, March 15-18, 2010. Proceedings, volume 5986 of Lecture Notes in Computer Science, pages 312–340. Springer, 2010.

[GL21] Alain Gutierrez and LIRMM. FCA4J: A java toolkit for formal concept analysis. https://www.lirmm.fr/fca4j/, 2021. Accessed 20 May 2025.

[GO12] Bernhard Ganter and Sergei Obiedkov. Conceptual Exploration, volume 4136 of Lecture Notes in Computer Science. Springer, 2012.

[GW99] Bernhard Ganter and Rudolf Wille. Formal Concept Analysis - Mathematical Foundations. Springer, 1999.

[JKK21] Radek Janostik, Jan Konecny, and Petr Krajca. Lincbo: Fast algorithm for computation of the duquenne-guigues basis. Inf. Sci., 572:223–240, 2021.

[Kuz93] Sergei O. Kuznetsov. A fast algorithm for computing all intersections of objects in a finite semi-lattice. Automatic Documentation and Mathematical Linguistics, 27(5):11–21, 1993.