





**DISSERTATION APPROVAL SHEET**

Title of Dissertation: Trustworthy Domain-Specific AI for Structured Knowledge Retrieval and Reasoning

Name of Candidate: Ryan Barron  
ryanb4@umbc.edu  
Doctor of Philosophy, 2025  
Graduate Program: Computer Science

**Dissertation and Abstract Approved:**

Signature

Signed by:  
**Dr. Cynthia Matuszek**  
8DABA3DBE05540F...  
Dr. Cynthia Matuszek  
cmat@umbc.edu

Associate Professor  
Title

Computer Science & Electrical Engineering  
Department  
9/29/2025 | 5:13 PM EDT

Signature:

Signed by:  
**Dr. Cynthia Matuszek**  
8DABA3DBE05540F...  
Dr. Cynthia Matuszek  
cmat@umbc.edu

Associate Professor  
Title

Computer Science & Electrical Engineering  
Department  
9/29/2025 | 5:14 PM EDT

NOTE: \*The Approval Sheet with the original signature must accompany the thesis or dissertation. No terminal punctuation is to be used.

EDUCATION

---

- **University of Maryland, Baltimore County (UMBC)** Baltimore, MD  
*Ph.D. in Computer Science (Anticipated)* *Dec. 2025*
  - Research: Large Language Models, Knowledge Graphs, HPC, NLP
- **University of Maryland, Baltimore County (UMBC)** Baltimore, MD  
*Master of Science in Computer Science; GPA: 3.74* *May 2023*
- **University of Maryland, Baltimore County (UMBC)** Baltimore, MD  
*Graduate Certificate in College Teaching and Learning Science* *May 2023*
- **University of Maryland, Baltimore County (UMBC)** Baltimore, MD  
*Bachelor of Science in Computer Science; GPA: 3.756, Magna Cum Laude* *Jan. 2021*
- **University of Maryland, Baltimore County (UMBC)** Baltimore, MD  
*Bachelor of Arts in Political Science; GPA: 3.756, Magna Cum Laude* *Jan. 2021*
  - Honors: Pi Sigma Alpha – Political Science Honor Society
  - Student Government Association Finance Board Representative
- **Carroll Community College (CCC)** Westminster, MD  
*Associate of Applied Science: Cybersecurity* *May 2018*
  - Student Government Association Senator

EXPERIENCE

---

- **Los Alamos National Laboratory (LANL)** Los Alamos, NM  
*Graduate Research Assistant* *Oct. 2022 – Present*
  - **Research Areas:** Conducted research in Large Language Models, Knowledge Graphs, Tensor Factorization, High-performance Computing, and Natural Language Processing.
  - **Collaboration:** Collaborated with Theoretical Chemistry and Cybersecurity Analytics teams on interdisciplinary projects.
- **University of Maryland, Baltimore County (UMBC)** Baltimore, MD  
*Adjunct Lecturer, Department of Computer Science* *Aug. 2022 – Jul. 2023*
  - **Course Instruction:** Taught CMSC 201, 210, 310, 341, and 462. Focused on programming, data structures, and data science in Python and C++.
- **University of Maryland, Baltimore County (UMBC)** Baltimore, MD  
*Graduate Teaching Assistant, Department of Computer Science* *Jan. 2021 – Jan. 2023*
  - **Instructional Support:** Supported CMSC 304, 421, 426, and 473/673 through grading, office hours, guest lectures, and course development.
- **University of Maryland, Baltimore County (UMBC)** Baltimore, MD  
*Undergraduate Teaching Fellow & Assistant, Department of Computer Science* *Aug. 2019 – Dec. 2020*
  - **Peer Instruction:** Led discussion sections, graded assignments, and mentored students in CMSC 201 and CMSC 341.

1. **Barron, R. C.**, Eren, M. E., Stanev, V., Matuszek, C., Alexandrov, B. S. *Topic Modeling and Link-Prediction for Material Property Discovery*. In *DocEng '25: 25th ACM Symposium on Document Engineering*, Sep. 2–5, 2025, University of Nottingham, UK. 4 pages.
2. Bhattarai, M., **Barron, R.**, Eren, M. E., Vu, M., Grantcharov, V., Boureima, I., Stanev, V., Matuszek, C., Valtchinov, V., Rasmussen, K. *HEAL: Hierarchical Embedding Alignment Loss for Improved Retrieval and Representation Learning*. In *ICLR '25 Workshop on Scaling Self-Improving Foundation Models without Human Supervision*, Apr. 21, 2025, Singapore. 10 pages.
3. **Barron, R.**, Eren, M. E., Truong, D. P., Matuszek, C., Wendelberger, J., Dorn, M. F., Alexandrov, B. S. *Matrix Factorization for Inferring Associations and Missing Links*. *arXiv preprint arXiv:2503.04680*, 2025.
4. **Barron, R. C.**, Eren, M. E., Serafimova, O. M., Matuszek, C., Alexandrov, B. S. *Bridging Legal Knowledge and AI*. In *ICAAIL '25: 20th International Conference on Artificial Intelligence and Law*, Jun. 16–20, 2025, Chicago, IL, USA. 10 pages.
5. **Barron, R. C.**, Grantcharov, V., Wanna, S., Eren, M. E., Bhattarai, M., Solovyev, N., Tompkins, G., Nicholas, C., Rasmussen, K., Matuszek, C. *Domain-Specific Retrieval-Augmented Generation Using Vector Stores, Knowledge Graphs, and Tensor Factorization*. In *ICMLA '24: 23rd IEEE Conference on Machine Learning and Applications—Special Session on ML for NLP*, Dec. 18–20, 2024, Miami, FL, USA. 8 pages.
6. **Barron, R.**, Eren, M. E., Bhattarai, M., Boureima, I., Matuszek, C., Alexandrov, B. S. *Binary Bleed: Fast Distributed and Parallel Method for Automatic Model Selection*. In *IEEE High Performance Extreme Computing Conference (HPEC)*, pp. 1–8, 2024.
7. Eren, M. E., **Barron, R.** *Catch'em All: Classification of Rare, Prominent, and Novel Malware Families*. In *12th International Symposium on Digital Forensics and Security (ISDFS)*, pp. 1–6, 2024. IEEE.
8. **Barron, R.**, Eren, M. E. *Cyber-Security Knowledge Graph Generation by Hierarchical Nonnegative Matrix Factorization*. In *12th International Symposium on Digital Forensics and Security (ISDFS)*, pp. 1–6, 2024. IEEE.
9. Wanna, S., Solovyev, N., **Barron, R.**, Eren, M. E., Bhattarai, M., Rasmussen, K. Ø, Alexandrov, B. S. *Topictag: Automatic Annotation of NMF Topic Models Using Chain of Thought and Prompt Tuning with LLMs*. In *Proceedings of the ACM Symposium on Document Engineering*, pp. 1–4, 2024.
10. Bhattarai, M., Kaymak, M. C., **Barron, R. C.**, Nebgen, B., Rasmussen, K. Ø, Alexandrov, B. S. *Defense of Supervised and Unsupervised Machine Learning Models*. Technical Report, 2023.
11. Higgins, P., **Barron, R.**, Engel, D., Matuszek, C. *A Comparative Analysis of VR-Based and Real-World Human-Robot Collaboration for Small-Scale Joining*. In *IEEE Conference on Virtual Reality and 3D User Interfaces Workshop (VRW)*, pp. 845–846, 2023.
12. Higgins, P., **Barron, R.**, Lukin, S., Engel, D., Matuszek, C. *A Collaborative Building Task in VR vs. Reality*. In *International Symposium on Experimental Robotics*, pp. 11–21, 2023. Springer Nature Switzerland Cham.
13. Solovyev, N., **Barron, R.**, Bhattarai, M., Eren, M. E., Rasmussen, K. Ø, Alexandrov, B. S. *Interactive Distillation of Large Single-Topic Corpora of Scientific Papers*. In *2023 International Conference on Machine Learning and Applications (ICMLA)*, pp. 1000–1005, 2023. IEEE.
14. Bhattarai, M., Kaymak, M. C., **Barron, R.**, Nebgen, B., Rasmussen, K. Ø, Alexandrov, B. S. *Robust Adversarial Defense by Tensor Factorization*. In *2023 International Conference on Machine Learning and Applications (ICMLA)*, pp. 308–315, 2023. IEEE.
15. Higgins, P., **Barron, R.**, Engel, D., Matuszek, C. *Lessons from a Small-Scale Robot Joining Experiment in VR*. 2023.

16. Higgins, P., **Barron, R.**, Matuszek, C. *Head Pose as a Proxy for Gaze in Virtual Reality*. In *5th International Workshop on Virtual, Augmented, and Mixed Reality for HRI*, 2022.
17. Higgins, P., **Barron, R.**, Matuszek, C. *Head Pose for Object Deixis in VR-Based Human-Robot Interaction*. In *2022 31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 610–617, 2022. IEEE.
18. Kebe, G. Y., Higgins, P., Jenkins, P., Darvish, K., Sachdeva, R., **Barron, R.**, Winder, J., Engel, D., Raff, E., Ferraro, F. *A Spoken Language Dataset of Descriptions for Speech-Based Grounded Language Learning*. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

## CERTIFICATIONS

---

- **Apple Ads Certification** — Apple, 2024
- **Neo4j-backed Chatbot Certification** — Neo4j, November 2023
- **Neo4j & Large Language Models Fundamentals** — Neo4j, November 2023
- **Applying the QM Rubric** — Quality Matters, October 2021
- **Social/Behavioral Research Course** — CITI Program, May 2021

## SKILLS

---

- **Programming Languages:** Python, C, C++, Go, Fortran, UNIX shell
- **Software & Tools:** Git, Docker,  $\text{\LaTeX}$ , Neo4j, Conda, LLM Prompting
- **Technical Areas:** Tensor Factorization, Knowledge Graphs, Machine Learning, Natural Language Processing, Robotics
- **Other Skills:** Academic research, teaching, training, consultation

## ABSTRACT

This dissertation presents a production-ready architecture that addresses the challenge of transforming vast volumes of unstructured, domain-specific text into structured and actionable knowledge representations that enable both retrieval and reasoning. Traditional pipelines for knowledge discovery remain fragmented, relying on brittle keyword extraction, ad hoc topic modeling, and disconnected reasoning modules. In response, this work proposes an integrated architecture that unifies semi-automatic corpus curation, semantic structuring, and retrieval systems into a scalable, interpretable pipeline.

The research introduces two novel techniques: Binary Bleed, an adapted binary search method that significantly reduces low-rank search complexity, particularly useful in Non-negative Matrix Factorization (NMF), and based on it, Hierarchical NMF (HNMF) with automatic selection of the number of latent features (HNMFk), a depth-adaptive topic modeling method that produces interpretable, Subject Matter Expert (SME) guided taxonomies. These representations are simultaneously instantiated as a typed, structured-based Knowledge Graph and a semantically aligned Vector Store, which includes the extracted latent features and is synchronized via an event-driven substrate.

To perform contextual information access, the research introduces a new form of Retrieval-Augmented Generation (RAG), named Tensor-Structured RAG (T-SRAG), capable of dynamically routing queries across retrieval paths. The system improves semantic fidelity and reduces hallucinations by incorporating a contrastive loss alignment technique that maps both document and query embeddings to hierarchical topic structures.

Beyond retrieval, the framework introduces a new Artificial Intelligence (AI) reasoning by identifying and completing missing links in the knowledge graph through tensor-based link prediction.

This enables inferential responses grounded in citation structure rather than generative hallucination from a Large Language Model (LLM).

The presented applications show validation across cybersecurity, law, materials science, and healthcare, demonstrating significant improvements in retrieval precision, early trend detection, as well as AI hypotheses generation and hallucination mitigation at scale. Overall, the dissertation delivers a new deployable, modular foundation for domain-specific, trustworthy AI systems capable of both retrieving and reasoning over structured knowledge.



Trustworthy Domain-Specific AI  
for Structured Knowledge Retrieval and Reasoning

by

Ryan C. Barron

Dissertation submitted to the Faculty of the Graduate School of the  
University of Maryland, Baltimore County in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
Fall 2025

**Advisor:**

Dr. Cynthia Matuszek

**Committee Members:**

Dr. Charles Nicholas

Dr. Tim Finin

Dr. Roberto Yus

Dr. Kim Rassmussen

Dr. Manish Bhattarai

© Copyright  
Ryan C. Barron  
2025

## Dedication

To my mother,  
whose unwavering belief in my potential lit the path forward,  
even when I was unsure of it myself.

You nurtured my love for learning long before I ever stepped into a classroom,  
and your quiet strength as a teacher inspired my own academic journey.  
It was through you that I learned the value of patience, of guiding others,  
and of striving toward knowledge, not just for achievement, but for understanding.

This work, and who I've become along the way,  
are rooted in your example, your encouragement, and your love.  
Thank you for always reaching higher and for lifting me with you.

## Acknowledgments

This dissertation has been assigned LA-UR-25-27945. Computational resources was provided by the Los Alamos National Laboratory's Institutional Computing Program supported by the U.S. Department of Energy National Nuclear Security Administration under Contract No. 89233218CNA000001.

I am deeply grateful to my advisor, Dr. Cynthia Matuszek, whose exceptional guidance, honest feedback, and steady support shaped not only this dissertation but also my growth as a researcher. Her clarity of thought, commitment to rigorous reasoning, and unwavering encouragement gave me confidence at every stage of this journey.

I extend sincere thanks to my committee members: Dr. Charles Nicholas, Dr. Tim Finin, Dr. Roberto Yus, Dr. Kim Rasmussen, and Dr. Manish Bhattarai. Their thoughtful critiques, insightful questions, and generous feedback helped sharpen the scope and depth of this work.

I am thankful to the Department of Computer Science and Electrical Engineering at the University of Maryland, Baltimore County. Their support through teaching assistantships, travel funding, and research infrastructure made this project possible. I am also especially grateful to Door to Virtue Lodge #46 in Westminster, Maryland, for their steady presence and brotherly encouragement, which grounded me during the most demanding phases of this journey.

Several faculty members offered mentorship that had a lasting impact on my academic development. Dr. Ben Johnson was a steady guide when I first arrived at the university. His encouragement, clarity, and example of instructional joy helped set the tone for my graduate path. Dr. Christopher Marron supported me through several teaching assistantships and served as a consistent mentor whose balance of precision and patience shaped my understanding of systems and instruction. Dr. Mohammad Donyaee, both as a professor and teaching mentor, provided structure, perspective, and dependable guidance during formative periods of my training. Though

not one of my instructors, Dr. Don Engel played a meaningful role in my research development. His careful guidance in crafting and refining research papers helped elevate the quality of my work and broaden its reach into interdisciplinary spaces. Professor Tompkins also offered professional insight and encouragement, both as a teacher and a mentor.

Other professors, instructors, and mentors who have guided and encouraged me over the years also deserve my sincere thanks. Bobby Keller, Marlene Titus, Matt Day, Robert Brown, and Janice Stencil each played a role in shaping my academic path. Whether through teaching, or mentorship their influence helped lay the foundation on which this work stands.

This research was also shaped by the colleagues and friends with whom I shared offices, labs, and long sessions of writing and problem solving. Kasra Darvish, Ves Grantcharov, Valentine Stanev, Vladimir Valtchinov, Olga Serafimova, Pdraig Higgins, Youssouf Kebe, Adam Berlier, Anjali Bhatt, Prajna Bhandary, and Nadja Bzhilyanskaya contributed sharp ideas, reliable collaboration, and a spirit of trust and friendship that sustained motivation throughout.

I am especially thankful for the friends whose presence made this journey not only possible but also deeply meaningful. Maksim Eren offered generous support in every dimension, emotional, social, and academic, and was a constant source of encouragement and perspective. Nick Solovyev was a close collaborator, contributing thoughtful insight and a shared sense of purpose throughout the research. Colin Veison provided steady companionship and editorial feedback that helped carry this work forward. Brian Bell, Michael Geyer, Liz Ruiz Carlos, Dillon Fornaro, Nishi Patel, and Sina Sontowski offered humor, kindness, and dependable support that helped me stay grounded through challenging times. Selma Wanna contributed both academic insight and social encouragement, helping me navigate obstacles with clarity and care.

I would like to specifically thank my undergraduate computer science advisor, Dr. Abhijit Dhutt,

for encouraging me to pursue a graduate degree in computer science, which ultimately led me to complete my dissertation.

I am also grateful to those who supported my professional growth along the way. Jean-Marc Henriette became a mentor during the middle stages of my education, and over time, a trusted friend whose guidance and friendship shaped both my scholarly direction and personal confidence. Jeff Kite offered early opportunities to develop practical skills in a professional setting, helping me build confidence and technical fluency. Each of you contributed something essential, whether mentorship, opportunity, or belief, and I carry your influence with deep gratitude.

I reserve special thanks for Boian Alexandrov. His insight, depth of thought, and boundless curiosity set a high standard for analytical rigor throughout this work. Boian played a central role in shaping the experimental design that underlies several key chapters. His willingness to engage with every level of the research, from conceptual foundations to technical implementation, turned challenges into opportunities. Working with him was both deeply instructive and profoundly motivating, and his influence will remain with me far beyond this dissertation.

Each person named here made a vital contribution to this journey. Whether through guidance, encouragement, collaboration, or friendship, your presence shaped this work in meaningful ways. I am honored to have had the opportunity to complete this research in your company, and I hope these pages reflect the generosity and insight you have shared with me.

# Contents

List of Tables . . . . .	x
List of Figures . . . . .	xii
<b>I Introduction and Background</b>	<b>1</b>
<b>1 Motivation and Thesis</b>	<b>2</b>
1.1 Motivation . . . . .	2
1.2 Thesis Statement . . . . .	3
1.3 Design Overview . . . . .	4
<b>2 Background</b>	<b>7</b>
2.1 Latent Variables and Factor Analysis . . . . .	7
2.2 Non-negative Matrix Factorization (NMF) . . . . .	10
2.3 Non-negative Matrix Factorization with Automatic Model Determination (NMFk) . . . . .	11
2.4 Logistic Matrix Factorization (LMF) . . . . .	14
2.5 Hierarchical Non-negative Matrix Factorization with Automatic Model Determination (HNMFk) . . . . .	17

2.6	Hierarchical Embedding Alignment Loss (HEAL)	20
2.7	Binary Search	21
2.8	Knowledge Graph Foundations	22
2.9	Latent Feature Extraction via Matrix and Tensor Factorization for Knowledge Graph Construction	22
2.10	Vector Stores for Semantic Retrieval	25
2.11	LLM-Based Agent Reasoning	27
<b>3</b>	<b>Literature Review</b>	<b>29</b>
3.1	Refined Data Collection	29
3.2	Factorization Innovation	31
3.3	Binary Directional Search	33
3.4	Law	36
3.5	Malware	40
<b>II</b>	<b>Methods</b>	<b>49</b>
<b>4</b>	<b>Data Acquisition</b>	<b>50</b>
4.1	Citation–Network Expansion and Initial Pruning	50
4.2	Semantic Query Filtering	55
4.3	Embedding and LLM-Based Corpus Pruning	57
4.4	Vocabulary Standardization: Acronym Resolution and Lexical Consolidation	59
4.5	Conclusion	62
<b>5</b>	<b>Latent Feature Extraction &amp; Topic Modeling</b>	<b>63</b>

5.1	Binary Bleed . . . . .	64
5.2	Automatic Topic Labelling with Embedding-Aligned LLM Prompts . . . . .	80
5.3	Conclusion . . . . .	82
<b>6</b>	<b>Knowledge-Graph &amp; Vector Store Retrieval Augmented Generation</b>	<b>84</b>
6.1	Unified Data-Infrastructure for Retrieval-Augmented Generation . . . . .	85
6.2	Conclusion . . . . .	90
<b>7</b>	<b>AI Reasoning</b>	<b>91</b>
7.1	Link Prediction . . . . .	91
7.2	Graph reasoning over latent knowledge graphs . . . . .	111
7.3	Conclusion . . . . .	112
<b>III</b>	<b>Applications &amp; Results</b>	<b>113</b>
<b>8</b>	<b>SMART-SLIC Framework</b>	<b>114</b>
8.1	Binary Bleed . . . . .	115
8.2	Cyber-Security KG via HNMFk . . . . .	120
8.3	Law . . . . .	123
8.4	Malware Analysis Research . . . . .	137
8.5	Conclusion . . . . .	141
<b>9</b>	<b>AI-Reasoning: Missing Links</b>	<b>143</b>
9.1	PPI Datasets . . . . .	144
9.2	Synthetic Data . . . . .	146
9.3	Materials . . . . .	160

9.4 Conclusion . . . . .	166
<b>10 Question-Answering</b>	<b>168</b>
10.1 Malware Analysis Research . . . . .	169
10.2 New Mexico Law . . . . .	173
10.3 Materials . . . . .	182
10.4 Conclusion . . . . .	184
<b>11 Conclusion</b>	<b>185</b>
11.1 Conclusion . . . . .	189
<b>References</b>	<b>190</b>

# List of Tables

1	Acronyms used in this dissertation . . . . .	xix
5.1	Chunks Pre/Post Traversal Sorts on two resources . . . . .	69
6.1	Core KG schema with provenance and monotone <code>version</code> properties. . . . .	86
7.1	Summary of the notation styles used in the paper. . . . .	92
8.1	NM Constitutional Depth-0 H-Clusters . . . . .	127
8.2	NM Statutory Depth-0 H-Clusters . . . . .	128
8.3	NM Appeals Court Case Law Depth-0 H-Clusters . . . . .	129
8.4	NM Supreme Court Case Law Depth-0 H-Clusters . . . . .	130
8.5	Neo4j Node and Edge Overview . . . . .	133
8.6	Labels for Topic Clusters . . . . .	137
9.1	The statistics of the five PPI datasets include the number of proteins and positive and negative interaction pairs of each PPI network when represented as a matrix. . .	145

9.3	Predicted scores assigned to materials whose labels were fully masked during training. Known superconductors (top block) score significantly higher than non-superconductors (bottom block), with a clear separation near 0.5. . . . .	165
9.2	Performance comparison of BNMFk, RNMFk, WNMFk, and their LMF-based ex- tensions across five protein–protein interaction datasets. Bold values indicate the best classical score per block; dashes indicate metrics not reported by uncertainty-quantified variants. . . . .	167
10.1	TMD corpus statistics. . . . .	182

# List of Figures

1.1	End-to-end pipeline: SME-seeded corpus → Binary Bleed rank search → HNMFk topic tree → synchronised KG + vector store → hybrid RAG agent. . . . .	6
4.1	BUNIE’s distillation pipeline. <b>Panel A:</b> The SME selects specific papers, shown as a bag-of-words wordcloud. <b>Panel B:</b> The dataset expands through the citation network. The wordcloud of a selected subset shows how the vocabulary differs from the core. <b>Panel C:</b> Human-in-the-loop pruning removes data. The wordcloud begins to resemble the core. <b>Panel D:</b> An embedding heuristic removes papers far from the core. The dataset becomes more compact. The wordcloud approaches parity with the original. <b>Panel E:</b> NMFk topic modeling produces H-clustering factorization to remove clusters lacking a core paper. A dense, yet substantial set of documents indicates a successful distillation. The wordcloud closely resembles the original. As indicated by the <b>Ellipse</b> , iterations can be made for refinement or the data extracted in repose for downstream analysis. . . . .	52
4.2	2-dimensional representation of hypersphere pruning. Core papers are red dots, citations/ references are blue squares, and hyperspheres are dotted red circles. . . . .	53

5.1	Primitive operations for asymptotic analysis. . . . .	68
5.2	Traversal sorts: preorder (red), inorder (green), postorder (blue). . . . .	68
5.3	Vanilla: Part 1 . . . . .	76
5.4	Vanilla: Part 2 . . . . .	76
5.5	Binary Bleed Vanilla. . . . .	77
5.6	Early Stop: Part 1 . . . . .	78
5.7	Early Stop: Part 2 . . . . .	78
8.1	NMFk (top row) and K-means (bottom row), Vanilla (Left) and Early stopping (Right). $\forall k_{optimal} = k_{true}$ . . . . .	115
8.2	Standard NMFk (left) and K-means (right), Vanilla, and Early Stop over Pre-order, and Post-order traversal sorting. . . . .	117
8.3	Binary Bleed reduction on distributed Rescal (Purple), and distributed NMF (Green).119	
8.4	Word cloud of selected topics and their interpreted labels. The method is hierar- chically applied three times, selecting a topic to extract its subtopics at each level. Selected topics from each level are shown in each row of the figure. . . . .	120
8.5	Distribution of the top ten document categories, based on the arXiv author assign- ments. Each topic corresponds to the word clouds from Figure 8.4. Selected topics from each level are shown in each row of the figure. . . . .	122
8.6	New Mexico Supreme/Appeals case counts per year. . . . .	124
8.7	10 latent topics from <b>Constitutional Provisions</b> . . . . .	124
8.8	985 latent topics from <b>Statutory Sections</b> . . . . .	125
8.9	420 latent topics from <b>Court of Appeals Cases</b> . . . . .	125
8.10	248 latent leaf topics from <b>Supreme Court Cases</b> . . . . .	126

8.11 Examination of ‘Estoppel’ relating to being a keyword in topics, vs bag of words vocabulary. . . . .	134
8.12 25 questions queried across five different LLM channels. “Attempts” indicate the percentage of responses that tried to answer the question, and “accuracy” represents the percentage of correct attempts. . . . .	135
8.13 Comparison of metrics (ROUGEL, NLI entailment, SummaC, and FactCC) across models: Nemotron-70B-Instruct, Claude 3 Opus, GPT-4o, Gemini Pro, and our system. . . . .	136
8.14 The KG schema. Images generated with DALL·E [170]. . . . .	139
8.15 Keyword ‘cybercrime’ graph search. A single keyword (green) and linked documents (light blue) are returned. The documents also link affiliated institutions (yellow) and the country of the institutions (red). . . . .	141
9.1 <b>Dog Dataset</b> - Four binary images are used as Boolean latent features to generate the synthetic data of shape $400 \times 16$ . . . . .	147
9.2 Results for Dog Data across methods including $\text{BNMF}_{k_{\text{kmeans}}}$ , $\text{NMF}_k$ , and $\text{WNMF}_k$ . Boolean thresholding is not used for $\text{NMF}_k$ and $\text{WNMF}_k$ . The first row presents violin plots visualizing the rank $k$ predictions at different test-set size levels. The second row displays RMSE scores for the test set, demonstrating the missing link prediction performance. The results are reported for each rank $k$ on the x-axis, with the dark/dashed vertical line across columns being the true rank $k = 4$ . . . . .	148

9.3 Results for Dog Data across methods, including BNMFk, NMFk, and WNMfK, evaluated under different Boolean thresholding techniques. The Boolean thresholding techniques are denoted with the subscripts of **kmeans**, **otsu**, and **search** (coordinate descent). The first row presents violin plots visualizing the rank  $k$  predictions at different test-set sizes. The second row displays RMSE scores for the test set, demonstrating the missing link prediction performance. The results are reported for each rank  $k$  on the x-axis, with the dark and dashed vertical line across each column representing the true rank  $k = 4$ . . . . . 149

9.4 Results for Dog Data across methods, including BNMFk, NMFk, and WNMfK, evaluated under different Boolean thresholding techniques. The Boolean thresholding techniques are denoted with the subscripts of **kmeans**, **otsu**, and **search** (coordinate descent). The first row presents the change in RMSE after making predictions on the non-abstained samples, where a negative change indicates improved performance (lower RMSE) for the given fraction of abstained samples. The second row illustrates the fraction of abstained predictions, representing the proportion of cases where the model opted not to make a prediction due to uncertainty. The results are reported for each rank  $k$  on the x-axis, with the dark and dashed vertical line across each column indicating the true rank  $k = 4$ . . . . . 150

9.5 Computation times for each method are shown in the log scale on the Dog Data. The results indicate that methods leveraging coordinate descent (**search**) for Boolean thresholding require significantly longer computation times than other approaches. . . 151

9.6 **Swimmer Dataset** - Dataset of 16 swimmer images. The first and third rows are the images with real-valued intensities ranging from 0 to 19. The second and fourth rows display the Boolean versions obtained after applying Otsu thresholding. For analysis, the Boolean versions of the dataset are represented as a matrix of size  $1024 \times 256$ . . . . . 152

9.7 The performance of each method on the Swimmer Data is shown. The first column features a violin plot displaying the distribution of  $k$  predictions for each method, with the y-axis representing the methods and the x-axis representing the predicted  $k$  values. The second through final columns present results at each  $k$  decomposition value, with the x-axis corresponding to the rank  $k$ . The dark dashed vertical lines indicate this dataset's true rank of  $k = 16$ . The Boolean thresholding techniques are denoted with the subscripts of **kmeans**, **otsu**, and **search** (coordinate descent), while the **uniform** subscript refers to running BNMFk without Boolean thresholding on the latent factors. NMFk and WNMfK, without subscripts, are the results of not using Boolean techniques with these methods. . . . . 153

9.8 Results for  $k$  predictions on Gaussian Data are shown for WNMfK and RNMFk methods across different data sparsity levels, represented by the increasing test-set size on the x-axis. The y-axis indicates the predicted  $k$  values. The results demonstrate that  $k$  predictions remain accurate for both methods until high levels of sparsity are reached, while WNMfK results in more accurate  $k$  predictions. Line color and style differentiate between different true  $k$  values. For instance, a solid line with circle markers represents decompositions where the matrix rank was  $k = 2$ . . . 155

9.9	Link prediction performance on test-set sizes of 10% (red lines) and 50% (blue lines) is shown for WNMfK and RNMfK, with RMSE scores plotted on a log scale. The x-axis represents the $k$ values, displaying results at different $k$ decompositions. Line markers are used to differentiate between the true $k$ values. For example, solid red lines with circle markers represent matrices with a true rank of $k = 2$ and a test-set size of 10%. . . . .	156
9.10	Pearson Correlation Coefficients with CI between UQ and errors at the test set are reported across different sparsity levels or test-set sizes that include 0.1, 0.5, and 0.9 (columns) and true matrix ranks $k$ including $k \in [2, 3]$ (rows). The shared x-axis differentiates between the methods WNMfK and RNMfK using hues, while the shared y-axis represents the Pearson correlation values. Bars with different colors correspond to results at various $k$ decompositions. . . . .	157
9.11	<b>Relative-Std-to-Mean Ratio of Uncertainty plot</b> provides a quantitative perspective on the behavior of uncertainty estimates across varying sparsity levels (or test set size) for each $k$ true. The top figure shows the standard deviation (std) at test points divided by the overall average of certainty (UQ). The bottom figure investigates certainty saturation by normalizing the std-to-mean uncertainty ratio at the top plot with the correlation between the UQ and errors at test points. . . . .	158
9.12	Mean hit@1 and hit@3 scores (bars, annotated) of the BNMF+LMF ensemble after masking every known superconducting link and an equal number of random zeros for each of the four benchmark compounds . . . . .	162

9.13	Split-violin plot of posterior probabilities for 24 superconducting edges (green). [See also Table 9.3.] 192 non-superconducting edges (blue) were added for comparison. (Note that these are identical for the four violins.) Lines denote the median and 25th/75th percentiles, with numbers showing cumulative counts. Superconductors concentrate near 1, non-superconductors near 0, indicating a clear separation. . . . .	164
10.1	Two question types, document and topic, showing LLM percent attempts and correct with <b>SMART-SLIC</b> RAG and without RAG. . . . .	170
10.2	MRR and top 10 hit rate percentage per data category over several embedding methods.	178

# List of Abbreviations

Table 1: Acronyms used in this dissertation

---

<b>Acronym</b>	<b>Definition</b>
AI	Artificial Intelligence
ANN	Approximate Nearest Neighbour
BERT	Bidirectional Encoder Representations from Transformers
BUNIE	Bibliographic Utility Network Information Expansion
BM25	Best Matching 25
BMF	Boolean matrix factorization
BNMF	Binary Non-negative Matrix Factorization
BNMF <sup>k</sup>	Binary NMF with Automatic Model Determination
CoT	Chain of Thought
CP	Connected Papers
CPD	Canonical Polyadic Decomposition
CSV	Comma-Separated Values

---

*Continued on next page*

---

<b>Acronym</b>	<b>Definition</b>
CV	Computer Vision
DOE	Department of Energy
DPR	Dense Passage Retrieval
GNN	Graph Neural Network
GPT	Generative Pre-trained Transformer
GloVe	Global Vectors for Word Representation
HDP	Hierarchical Dirichlet Processes
HEAL	Hierarchical Embedding Alignment Loss
HITL	Human in the Loop
hLDA	hierarchical Latent Dirichlet Allocation
HNMF	Hierarchical Non-negative Matrix Factorization
HNMF <sub>k</sub>	Hierarchical NMF with Automatic Model Determination
HNSW	Hierarchical Navigable Small-World graph
ICA	Independent Component Analysis
IVF	Inverted File Indexing
KG	Knowledge Graph
KNN	K-Nearest Neighbors
LLM	Large Language Model
LMF	Logistic Matrix Factorization
LANL	Los Alamos National Laboratory
MF	Matrix Factorization

---

*Continued on next page*

---

<b>Acronym</b>	<b>Definition</b>
MDL	Minimum Description Length
MCMC	Markov Chain Monte Carlo
MPI	Message Passing Interface
MRR	Mean Reciprocal Rank
nCRP	nested Chinese Restaurant Process
NMF	Non-negative Matrix Factorization
NMFk	Non-negative Matrix Factorization with Automatic Model Determination
NMRA	New Mexico Rules Annotated
PCA	Principal Component Analysis
PH	Pulmonary Hypertension
PPI	Protein-Protein Interaction
PQ	Product Quantization
PR AUC	Precision-Recall – Area Under Curve
QA	Question Answering
RAG	Retrieval-Augmented Generation
RMSE	Root Mean Squared Error
RNMFk	Recommender NMF with Automatic Model Determination
ROC AUC	Receiver Operating Characteristic – Area Under Curve
SeNMFk	Semantic NMF with automatic model determination
SciNCL	Neighborhood Contrastive Learning for Scientific Document Representations with Citation Graph Embeddings

---

*Continued on next page*

---

<b>Acronym</b>	<b>Definition</b>
SME	Subject Matter Expert
SVD	Singular Value Decomposition
T-SRAG	Tensor-Structured Retrieval-Augmented Generation
TD	Tucker Decomposition
TF-IDF	Term Frequency–Inverse Document Frequency
TMD	Transition-Metal Dichalcogenide
t-SNE	t-Distributed Stochastic Neighbour Embedding
UQ	Uncertainty Quantification
UMAP	Uniform Manifold Approximation and Projection
VS	Vector Store
WNMF	weighted NMF
WNMFk	Weighted NMF with Automatic Model Determination

---

# Part I

## Introduction and Background

# Chapter 1

## Motivation and Thesis

### 1.1 Motivation

Across fields like science, law, cybersecurity, and engineering, professionals are turning to AI systems to make sense of massive volumes of unstructured text. But as the scale of information grows, so does the need for systems that are not only accurate but also explainable, auditable, and aligned with domain-specific knowledge. Experts need more than just answers; they need insight that can be traced, trusted, and integrated into decision-making workflows.

While recent progress in LLMs, vector-based retrieval, and matrix factorization has been impressive, the tools often operate in silos. Keyword-based methods still let in too much irrelevant data. Flat topic models miss hierarchical structure. Dense retrieval pipelines, such as those used in RAG, frequently hallucinate because their underlying vector stores lack semantic grounding. Despite the power of GPU-accelerated computation, high-dimensional search algorithms like HNSW and instruction-tuned LLMs, these components remain loosely coupled in current implementations.

This dissertation introduces an end-to-end framework that brings these parts together into a coherent, modular pipeline. The system is designed to be transparent, scalable, and adaptable, enabling it to work across domains and be tuned and extended by domain experts themselves. By tightly integrating symbolic reasoning, embeddings, and interpretable topic hierarchies, the framework supports knowledge discovery that is both robust and explainable.

## 1.2 Thesis Statement

By integrating an SME-seeded citation-expansion and embedding-pruning loop, a Binary Bleed-accelerated Hierarchical NMFk model for automatic rank discovery, and an event-synchronized dual symbolic-vector knowledge layer orchestrated by a hybrid T-SRAG agent, this dissertation delivers a scalable, interpretable, and domain-portable framework that enables both retrieval and reasoning over large, multi-domain text corpora.

This claim is substantiated through:

- 1. Corpus Curation.** A multi-stage expansion-pruning loop that combines citation-graph walks, embedding hypersphere filters, and LLM-assisted relevance voting.
- 2. Hierarchical Modelling.** The Binary Bleed algorithm for rapid rank search and HNMFk for depth-adaptive, interpretable topic trees.
- 3. Unified Knowledge Layer.** Dual instantiation of topic-document relationships as typed triples in a knowledge graph and semantically aligned vectors in a high-performance store.
- 4. Hybrid Retrieval & Reasoning.** A T-SRAG agent that dynamically blends graph walks, dense vector retrieval, and tensor-based link prediction for grounded answers.

### 1.3 Design Overview

Figure 1.1 presents a high-level overview of the proposed system architecture. The pipeline begins with a curated seed corpus, selected by domain experts to ensure initial relevance and quality. From this trusted foundation, citation networks are traversed outward to collect additional candidate documents. These are subsequently filtered through a multi-stage refinement process, including keyword-based screening, embedding-based similarity scoring, and rapid expert validation via binary relevance judgments. The result is a focused and domain-specific corpus suitable for downstream analysis.

To identify the latent thematic structure within this corpus, the system employs a novel rank estimation algorithm, Binary Bleed, which efficiently determines the optimal number of topics. This method operates via a binary search strategy that iteratively narrows the candidate space, significantly reducing computational overhead compared to exhaustive approaches. Once the rank is established, the HNMFk algorithm is applied to derive a tree-structured topic hierarchy. This hierarchy captures coarse-grained topics at higher levels and increasingly fine-grained subtopics toward the leaves. Each node is automatically annotated with representative keywords to facilitate human interpretability.

The resulting document-topic associations are then dual-encoded: first, as triples embedded in a knowledge graph, and second, as dense vector representations indexed in a high-performance vector store. A synchronization mechanism ensures consistency between the symbolic representations. At the interface layer, a hybrid RAG agent acts as an intelligent query orchestrator. Given a user query, it dynamically chooses between symbolic reasoning over the knowledge graph (suitable for structured, multi-hop queries), vector-based retrieval (for semantically fuzzy queries), or

a combination of both. Retrieved evidence is then passed to an LLM, which generates fluent, citation-backed responses. In summary, the system transforms a small, expert-curated set of documents into a robust, semantically organized, and interoperable knowledge system. It supports transparent, context-aware interactions through the integration of symbolic reasoning and retrieval mechanisms.

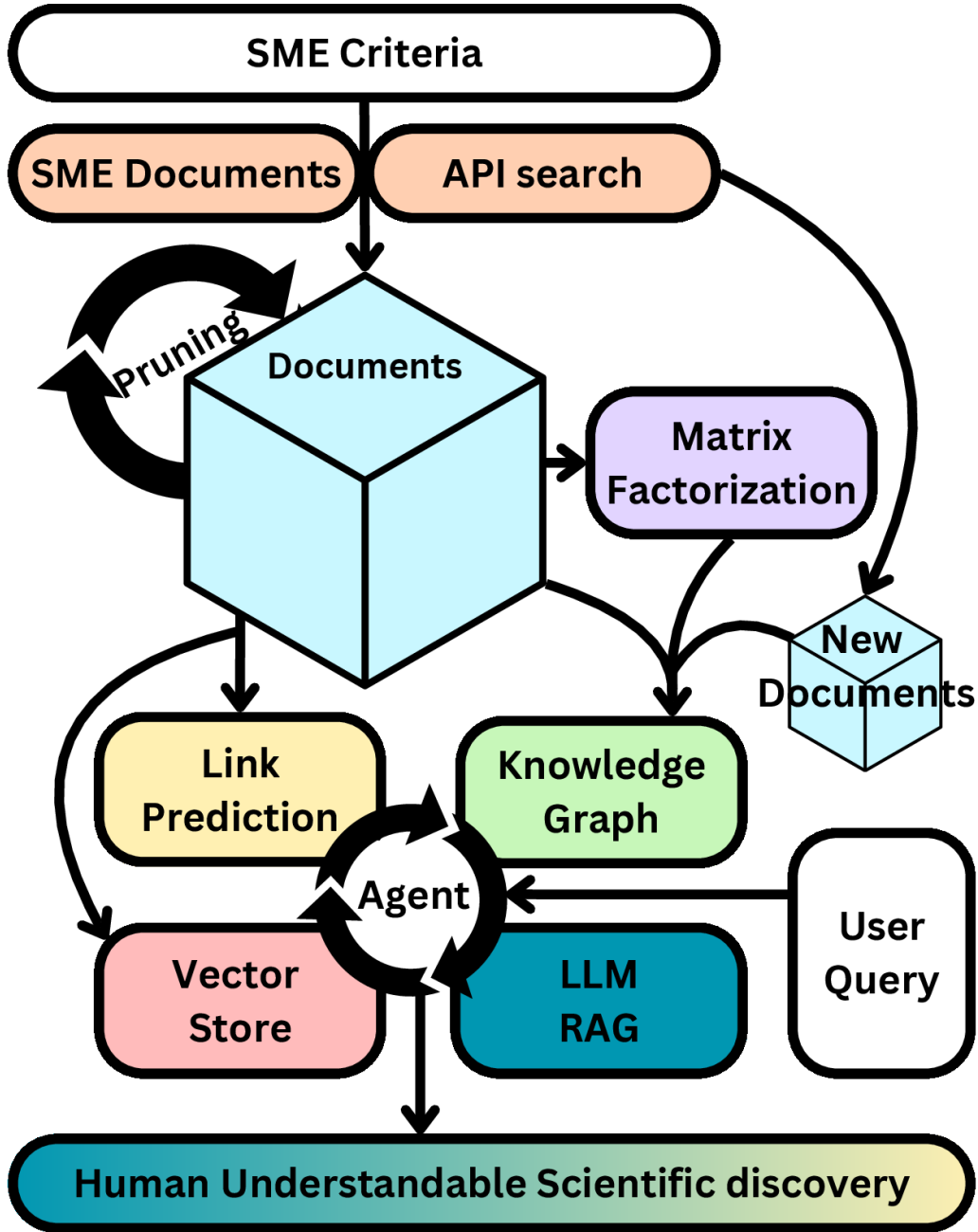


Figure 1.1: End-to-end pipeline: SME-seeded corpus → Binary Bleed rank search → HNMFk topic tree → synchronised KG + vector store → hybrid RAG agent.

# Chapter 2

## Background

### 2.1 Latent Variables and Factor Analysis

In principle, it is challenging to directly link collected data to the underlying processes that generated it. Every dataset consists only of directly observable quantities (such as state variables), while the underlying processes are typically complex, interdependent, and manifest simultaneously across multiple observables. In most cases, it is impossible to measure even known processes because they are intertwined directly, and often there are unknown processes that remain latent, not directly observable, and need to be discovered. These hidden, unobserved, or latent variables or features are referred to as latent variables [16].

Latent variable extraction reduces the dimensionality of extensive collections of observables to a smaller set of latent variables, where each observable can be represented as a combination of these latent factors. This form of dimensionality reduction is central to factor analysis [81], which underpins many modern approaches in data mining and data-driven science.

### 2.1.1 High-Dimensional Data and Tensor Representations

Most contemporary big data sources are inherently high-dimensional and are most naturally represented as tensors rather than as two-dimensional matrices [119, 45]. Common examples include imagery, spatiotemporal sensor networks, and large-scale text corpora. Such datasets typically encode the signatures of multiple concurrent latent processes (e.g., explosions, earthquakes, topic distributions), each imprinting distinct patterns on observable state variables (e.g., temperature, pressure, density, token counts) across various dimensions (e.g., spatial, temporal, lexical).

The identification of these latent processes requires specialized unsupervised machine learning (ML) methods capable of handling high-dimensional data. Traditional techniques such as clustering, principal component analysis (PCA) [72], singular value decomposition (SVD) [111], independent component analysis (ICA) [95], NMF [124], and neural networks (NNs) have proven highly effective for matrix data. However, flattening high-dimensional tensor data into matrix form destroys essential cross-dimensional correlations. Therefore, established dimension-reduction techniques must be generalized to tensor data to preserve and exploit the inherent structure of modern datasets fully.

### 2.1.2 Importance of Nonnegativity Constraints

Incorporating non-negativity constraints in latent feature extraction ensures that the resulting components are physically interpretable [44]. Such part-based representations [55] arise because, under a non-negativity constraint, reconstructions of the data can only be formed by additive combinations of the extracted features, making these features correspond to meaningful and interpretable parts of the original data. Many types of state variables, such as density, energy, spectral intensities, and population counts, are naturally nonnegative, and others can often be transformed into nonnegative forms as required.

### 2.1.3 Example: Blind Source Separation

Consider, for instance, a scenario in which microphones are distributed throughout a noisy room where multiple speakers are engaged in conversation. Each microphone captures a complex mixture of voices, background sounds, and noise. In this setting, the latent features correspond to the individual voices, signals that cannot be recorded independently but can be extracted through analysis of the observed mixtures. The process of extracting these latent features effectively reduces the dimensionality of the data and identifies low-dimensional subspaces [189, 214] that succinctly represent the entire dataset.

### 2.1.4 Tensor Factorization: A Modern Approach

One of the most powerful and versatile tools for latent feature extraction from high-dimensional data is tensor factorization (TF) [44]. The goal of TF is to decompose a tensor into a set of factor matrices (one for each tensor mode) and a small core tensor. The factor matrices capture the latent features in each respective dimension of the tensor.

Two principal forms of TF are widely used:

- Canonical Polyadic Decomposition (CPD) [82], also known as PARAFAC
- Tucker Decomposition (TD) [216], a higher-order generalization of matrix factorization

These methods extend the principles of classical factor analysis to tensor data, providing a rigorous mathematical framework for uncovering the complex latent structures embedded in modern, high-dimensional datasets.

## 2.2 Non-negative Matrix Factorization (NMF)

NMF is an unsupervised learning method based on low-rank approximation that reduces dimensionality. NMF approximates a given observation matrix  $\mathbf{X} \in \mathbb{R}_+^{n \times m}$  with non-negative entries, as a product of two non-negative matrices, i.e.,  $\mathbf{X} \approx \mathbf{WH}$  and  $\hat{\mathbf{X}} = \mathbf{WH}$ , where  $\mathbf{W} \in \mathbb{R}_+^{n \times k}$ , and  $\mathbf{H} \in \mathbb{R}_+^{k \times m}$ , and usually  $k \ll m, n$ . Here,  $n$  is the number of features (rows),  $m$  is the number of samples (columns), and the small dimension  $k$  is the low-rank of the approximation. This factorization is performed through a non-convex minimization with non-negativity constraint, utilizing the multiplicative updates algorithm [124], and the Frobenius norm as the distance metric, with an objective function:

$$\min_{\mathbf{W} \in \mathbb{R}_+^{n \times k}, \mathbf{H} \in \mathbb{R}_+^{k \times m}} \|\mathbf{X} - \mathbf{WH}\|_F^2, \quad (2.1)$$

which allows NMF to be treated as a Gaussian mixture model [70]. In Equation 2.1, the factors  $\mathbf{W}$  and  $\mathbf{H}$  are the solution of the optimization problem, the latent factors, estimated via alternative updates, with update rules in Equations 2.2 and 2.3 respectively, and the performance of the minimization is evaluated by the relative reconstruction error in Equation 2.4:

$$\mathbf{W} = \mathbf{W} \odot \frac{\mathbf{XH}^T}{\mathbf{WHH}^T}, \quad (2.2) \quad \mathbf{H} = \mathbf{H} \odot \frac{\mathbf{W}^T\mathbf{X}}{\mathbf{W}^T\mathbf{WH}}, \quad (2.3) \quad \text{Relative Error} = \frac{\|\mathbf{X} - \mathbf{WH}\|_F^2}{\|\mathbf{X}\|_F^2}. \quad (2.4)$$

## 2.3 Non-negative Matrix Factorization with Automatic Model

### Determination (NMFk)

The NMF minimization requires prior knowledge of the latent dimensionality,  $k$ , and the number of latent features, which is often unknown. Choosing too small a value for  $k$  leads to a poor approximation of the observables in  $\mathbf{X}$  (a problem called under-fitting). At the same time, setting  $k$  too large makes the extracted features hard to interpret, as they also capture noise in the data (a case of over-fitting). In essence, selecting  $k$  is equivalent to estimating the number of model parameters, a well-known complex problem.

In general, the existing partial solutions to this problem are heuristic. Among these solutions is Automatic Relevance Determination (ARD) [143], which was first modified for Principal Component Analysis [25] and then for NMF [159, 208]. Another approach is based on the assumed stability of the NMF solution and was proposed to identify the number of stable clusters in the observational matrix  $\mathbf{X}$  [27]. A recent model selection technique, NMFk [5], has successfully decomposed the largest dataset of human cancer genomes [7]. NMFk integrates classical NMF-minimization with custom clustering and Silhouette statistics [187] and combines the accuracy of the minimization and robustness/stability of the NMF solutions when a modified bootstrap procedure (i.e., generation of a random ensemble of slightly perturbed input matrices) is applied to estimate the number of latent features. Recently, NMFk was applied to many synthetic datasets with a predetermined number of latent features, and it demonstrated its superior performance of correctly estimating  $k$  compared to the other known heuristics [160]. The exceptional performance of the NMFk method as a model selection was also demonstrated both in practice [6] and in a large set of synthetic cancer genomes with a predetermined number of latent features [99]. In addition, it was shown that NMFk performs

better than spherical k-means and other methods for topic extraction [220]. Therefore, NMFk is the core factorization method with automatic model selection and is adopted to WNMFk, RNMFk, and BNMFk for heuristically estimating  $k$ . For completeness, the pseudocode for it is in Algorithm 1, and a description of it is as follows:

---

**Algorithm 1** NMFk( $\mathbf{X}$ ,  $k^{min}$ ,  $k^{max}$ ,  $M$ ,  $Sill\_thr = 0.8$ )

---

**Require:** :  $\mathbf{X} \in \mathbb{R}_+^{n \times m}$ ,  $k^{min}$ ,  $k^{max}$ ,  $r$

- 1: **for**  $k$  in  $k^{min}$  to  $k^{max}$  **do** ▷ Start and end process for NMFk
- 2:     **for**  $q$  in 1 to  $M$  **do** ▷ Num. of Perturbations on each k
- 3:          $\mathbf{X}_{::q} = \text{Perturb}(\mathbf{X})$  ▷ Resampling  $\mathbf{X}$  to create a random ensemble
- 4:          $\mathbf{W}_{::kq}, \mathcal{H}_{::kq} = \text{NMF}(\mathbf{X}_{::q}, k)$
- 5:     **end for**
- 6:      $\mathbf{W}^{all} = [\mathbf{W}_{::k1}, \dots, \mathbf{W}_{::kM}]$  and  $\mathcal{H}^{all} = [\mathcal{H}_{::k1}, \dots, \mathcal{H}_{::kM}]$
- 7:      $\hat{\mathbf{W}}, \hat{\mathcal{H}} = \text{customCluster}(\mathbf{W}^{all}, \mathcal{H}^{all})$
- 8:      $\tilde{\mathbf{W}}_{::k} = \text{medians}(\hat{\mathbf{W}})$
- 9:      $\mathcal{H}_{::k}^{reg} = \text{NNLS}(\mathbf{X}, \tilde{\mathbf{W}}_{::k})$  ▷ Column-wise regression of  $\mathbf{H}$  with  $\tilde{\mathbf{W}}$  and column of  $\mathbf{X}$
- 10:      $\mathbf{s}_k = \text{clusterStability}(\hat{\mathbf{W}})$
- 11:      $\mathbf{err}_k = \text{reconstructErr}(\mathbf{X}, \tilde{\mathbf{W}}_{::k}, \mathbf{H}_{::k}^{reg})$  ▷ Column-wise reconstruction error for L-statistics
- 12: **end for**
- 13:  $\mathbf{err}^{all} = [\mathbf{err}_{k^{min}}, \dots, \mathbf{err}_{k^{max}}]$
- 14:  $k^{opt} = \text{PvalueAnalysis}(\mathbf{err}^{all}, k^{min}, k^{max}, \mathbf{s}_k, Sill\_thr)$  ▷ Predicted k value using Wilcoxon
- 15: **return**  $\tilde{\mathbf{W}}_{::k^{opt}}, \mathcal{H}_{::k^{opt}}^{reg}, k^{opt}$

**Ensure:**  $k = k^{opt}, \tilde{\mathbf{W}}_{::k^{opt}} \in \mathbb{R}_+^{n \times k}, \mathcal{H}_{::k^{opt}}^{reg} \in \mathbb{R}_+^{k \times m}, \mathbf{X} = \tilde{\mathbf{W}}_{::k^{opt}} \mathcal{H}_{::k^{opt}}^{reg}$

---

1. *Resampling*: Based on the observable matrix,  $\mathbf{X}$ , NMFk creates an ensemble of  $M$  random

matrices,  $[\mathbf{X}::q]_{q=1,\dots,M}$ , with means equal to the original matrix  $\mathbf{X}$ . Each one of these random matrices  $\mathbf{X}::q$  is generated by perturbing the elements of  $\mathbf{X}$  by a small uniform noise, such that:  $\mathbf{X}_{ijq} = \mathbf{X}_{ij} + \delta_{ijq}$ , for each  $q = 1, \dots, M$ , where  $\delta_{ijq}$  is the small error that is part of random distribution, different error for each  $q$ .

2. *NMF minimization*: The Frobenius norm-based multiplicative updates (MU) algorithm [124] is used for different numbers of latent features,  $k$ , in an interval  $[k^{min}, k^{max}]$ , for each generated  $M$  random matrices.
3. *Custom clustering*: For each  $k \in [k^{min}, k^{max}]$ , NMF minimizations of the  $M$  random matrices,  $[\mathbf{X}::q]_{q=1,\dots,M}$ , results in  $M$  pairs  $[\mathbf{W}::kq; \mathcal{H}::kq]_{q=1,\dots,M}$ . Further, NMFk clusters the set of the  $M * k$  latent features, the columns of  $\mathbf{W}::kq$ . The NMFk custom clustering is similar to k-means, but it holds exactly one column for each cluster from each of the  $M$  NMF solutions. This constraint is needed since each NMF minimization gives exactly one solution  $\mathbf{W}::kq$  with the same number of columns,  $k$ . In clustering, the cosine similarity metric measures the similarity between the columns. Here, cosine similarity determines the degree of similarity between two vectors in an inner product space.
4. *Robust  $\mathbf{W}$  and  $\mathbf{H}$  for each  $k$* : The medians of the clusters,  $\tilde{\mathbf{W}}::k$ , are the robust solution for each explored  $k$ . The corresponding mixing coefficients  $\mathcal{H}::k^{reg}$  are calculated by regression of  $\mathbf{X}$  on  $\tilde{\mathbf{W}}::k$ .
5. *Cluster stability via Silhouette statistics*: NMFk explores the stability of the obtained clusters, for each  $k$ , by calculating their Silhouettes [187]. Silhouette scores quantify the cohesion and separability of the clusters. The Silhouette values range between  $[-1, 1]$ , where  $-1$  means an unstable cluster and  $+1$  means perfect stability.

6. *Reconstruction error*: Another metric NMFk uses is the relative reconstruction error,  $R = \|\mathbf{X} - \mathbf{X}_{::k}^{rec}\| / \|\mathbf{X}\|$ , where  $\mathbf{X}_{::k}^{rec} = \tilde{\mathbf{W}}_{::k} * \mathcal{H}_{::k}^{reg}$ , which measures the accuracy of the reproduction of initial data by a given solution and the number of latent features  $k$ .
7. *L-statistics*: NMFk uses L-statistics [218] to automatically estimate the number of latent features. To calculate L-statistics for each  $k$ , NMFk records the distributions of the column reconstruction errors,  $\mathbf{e}_i = \|\mathbf{X}_{:j} - \mathbf{X}_{:j}^{rec}\| / \|\mathbf{X}_{:j}\|$ ;  $j = 1, \dots, m$ . L-statistics compares the distributions of column errors for different  $k$  by a two-sided Wilcoxon rank-sum test to evaluate if two samples are taken from the same population [84].
8. *NMFk final solution*: The optimal number of latent features,  $k^{opt}$ , is the largest stable cluster count with low reconstruction error. The Wilcoxon rank-sum test computes the p-value for  $k^{opt}$ , ensuring subsequent column error distributions remain statistically similar, indicating noise fitting. L-statistics and a *minimum* Silhouette threshold of 0.80 guide selection, preventing overfitting. The corresponding  $\tilde{\mathbf{W}}_{::k^{opt}}$  and  $\mathcal{H}_{::k^{opt}}^{reg}$  are the robust solutions for the low-rank factor matrices.

## 2.4 Logistic Matrix Factorization (LMF)

LMF extends the principles of matrix factorization to binary data, where the observed matrix  $\mathbf{X} \in \{0, 1\}^{n \times m}$  represents the presence or absence of interactions or a known link [110]. Unlike NMF, which minimizes reconstruction error using the Frobenius norm, LMF incorporates the logistic regression model to capture the likelihood of binary observations. In the binary interaction matrix  $\mathbf{X} \in \{0, 1\}^{n \times m}$ , each entry  $\mathbf{X}_{ij}$  represents the presence ( $X_{ij} = 1$ ) or absence ( $X_{ij} = 0$ ) of a link between node  $i$  and node  $j$ . However, in many real-world datasets, some interactions are

unobserved or missing (i.e., the missing link). To account for this, binary mask matrix is defined  $\mathbf{M} \in \{0, 1\}^{n \times m}$ , where  $\mathbf{M}_{ij} = 1$  if  $\mathbf{X}_{ij}$  is observed or a known link, and  $\mathbf{M}_{ij} = 0$  for the missing links. For instance, in PPI networks,  $\mathbf{X}_{ij} = 1$  might indicate a known interaction between proteins  $i$  and  $j$ , while  $\mathbf{X}_{ij} = 0$  represents the absence of an interaction. If the interaction status between proteins  $i$  and  $j$  has not been experimentally determined,  $\mathbf{M}_{ij} = 0$  is used to indicate the missing entry at  $\mathbf{X}_{ij}$ . In LMF,  $\mathbf{X}$  is approximated as:

$$\hat{\mathbf{X}} = \sigma(\mathbf{W}\mathbf{H} + \mathbf{b}_r + \mathbf{b}_c), \quad (2.5)$$

where  $\mathbf{W} \in \mathbb{R}^{n \times k}$  is the row latent feature matrix,  $\mathbf{H} \in \mathbb{R}^{k \times m}$  is the column latent feature matrix,  $\mathbf{b}_r \in \mathbb{R}^{n \times 1}$  is the row bias vector (broadcasted across columns),  $\mathbf{b}_c \in \mathbb{R}^{1 \times m}$  is the column bias vector (broadcasted across rows), and the element-wise logistic sigmoid function:

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \quad (2.6)$$

Row and column biases are incorporated into the LMF model to account for systematic variations in the data. The row bias  $\mathbf{b}_r \in \mathbb{R}^{n \times 1}$  captures row-specific effects, such as a node’s general tendency to form links across all columns. Similarly, the column bias  $\mathbf{b}_c \in \mathbb{R}^{1 \times m}$  models column-specific effects, such as a particular node’s propensity to attract connections. These biases are added to the matrix reconstruction, allowing the model to better fit the data by accounting for individual node-specific tendencies. For example, in a PPI network, the row bias  $\mathbf{b}_r$  could represent the inherent interaction likelihood of a specific protein across all other proteins. In contrast, the column bias  $\mathbf{b}_c$  adjusts for the overall connectivity tendency of a target protein across all rows.

Each entry  $\hat{\mathbf{X}}_{ij}$  in the reconstructed matrix represents the predicted probability of  $\hat{\mathbf{X}}_{ij}$ . The optimization objective in LMF minimizes the negative log-likelihood of the binary observations

under the logistic model:

$$\underset{\mathbf{W}, \mathbf{H}, \mathbf{b}_r, \mathbf{b}_c}{\text{minimize}} - \sum_{i=1}^n \sum_{j=1}^m \mathbf{M}_{ij} \left( \mathbf{X}_{ij} \log \hat{\mathbf{X}}_{ij} + (1 - \mathbf{X}_{ij}) \log(1 - \hat{\mathbf{X}}_{ij}) \right) + \lambda \left( \|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2 + \|\mathbf{b}_r\|_2^2 + \|\mathbf{b}_c\|_2^2 \right), \quad (2.7)$$

where  $\lambda$  is the regularization parameter to prevent overfitting. The optimization problem is solved using gradient-based methods. The gradients for each parameter are computed as follows:

$$\frac{\partial L}{\partial \mathbf{W}} = \mathbf{M} \odot (\hat{\mathbf{X}} - \mathbf{X}) \mathbf{H}^T + \lambda \mathbf{W}, \quad (2.8)$$

$$\frac{\partial L}{\partial \mathbf{H}} = \mathbf{W}^T \left( \mathbf{M} \odot (\hat{\mathbf{X}} - \mathbf{X}) \right) + \lambda \mathbf{H}, \quad (2.9)$$

$$\frac{\partial L}{\partial \mathbf{b}_r} = \sum_{j=1}^m \mathbf{M} \odot (\hat{\mathbf{X}} - \mathbf{X}) + \lambda \mathbf{b}_r, \quad (2.10)$$

$$\frac{\partial L}{\partial \mathbf{b}_c} = \sum_{i=1}^n \mathbf{M} \odot (\hat{\mathbf{X}} - \mathbf{X}) + \lambda \mathbf{b}_c, \quad (2.11)$$

where

$$L = \frac{1}{2} \|\mathbf{M} \odot (\hat{\mathbf{X}} - \mathbf{X})\|_F^2 + \frac{\lambda}{2} \left( \|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2 + \|\mathbf{b}_r\|_F^2 + \|\mathbf{b}_c\|_F^2 \right). \quad (2.12)$$

The parameters  $\mathbf{W}$ ,  $\mathbf{H}$ ,  $\mathbf{b}_r$ , and  $\mathbf{b}_c$  are updated iteratively using gradient descent:

$$\mathbf{W} \leftarrow \mathbf{W} - \eta \frac{\partial L}{\partial \mathbf{W}}, \quad \mathbf{H} \leftarrow \mathbf{H} - \eta \frac{\partial L}{\partial \mathbf{H}}, \quad \mathbf{b}_r \leftarrow \mathbf{b}_r - \eta \frac{\partial L}{\partial \mathbf{b}_r}, \quad \mathbf{b}_c \leftarrow \mathbf{b}_c - \eta \frac{\partial L}{\partial \mathbf{b}_c}, \quad (2.13)$$

where  $\eta$  is the learning rate. LMF is particularly well-suited for sparse, binary datasets, as it naturally handles probabilistic modeling of binary outcomes. Adding biases for rows and columns allows for improved flexibility in capturing systematic variations in data.

## 2.5 Hierarchical Non-negative Matrix Factorization with Automatic Model Determination (HNMF<sub>k</sub>)

Hierarchical NMF<sub>k</sub> (HNMF<sub>k</sub>) recursively applies the model-selection variant of NMF (Section 2.3) to uncover a multi-resolution latent structure in  $\mathbf{X}$ . Starting from the whole matrix, each call to NMF<sub>k</sub> produces a low-rank approximation  $\mathbf{X} \approx \widetilde{\mathbf{W}}\widetilde{\mathbf{H}}$  and a data-driven estimate  $\hat{k}$  of the number of coherent topics at the current resolution. Samples are then assigned to topics with the maximum-loading rule (row-wise arg max of  $\widetilde{\mathbf{H}}$  when clustering on  $\mathbf{H}$ , or column-wise arg max of  $\widetilde{\mathbf{W}}$  when clustering on  $\mathbf{W}$ ). Each topic becomes a child node that is factorized again, yielding a tree-structured decomposition  $\mathcal{T}$  whose depth is controlled by (i) a user-supplied limit  $d_{\max}$ , (ii) a minimum-sample threshold  $n_{\min}$ , and (iii) the stopping condition  $\hat{k} = 1$  or a single populated cluster. The process terminates when all remaining nodes are declared leaves. The resulting hierarchy exposes global themes near the root and increasingly specialized subtopics toward the leaves, mirroring the coarse-to-fine semantics observed in real-world corpora [64, 80].

**Automatic model selection at every level.** Because NMF<sub>k</sub> estimates  $\hat{k}$  independently in every node, HNMF<sub>k</sub> adapts its branching factor to local data complexity: dense, heterogeneous regions of  $\mathbf{X}$  receive more children, whereas sparse or homogeneous regions collapse early. This avoids the under/over-fitting pitfalls of fixed-depth binary splits and has been shown to outperform both spherical  $k$ -means trees and heuristic hierarchical NMF baselines on text, malware, and genomic datasets [64, 5].

**Scalable HPC implementation.** Algorithm 2 encapsulates the parallel implementation used in the experiments.<sup>1</sup> Job scheduling is handled by Message Passing Interface (MPI) [152], allowing independent nodes of  $\mathcal{T}$  to be factorized concurrently across an arbitrary number of compute nodes. Check-pointing makes the procedure failure-resilient and enables incremental continuation on large corpora (hundreds of millions of non-zeros). Memory footprint is bounded because only the factors  $\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}}$  and sample indices are stored for completed nodes; raw slices of  $\mathbf{X}$  are regenerated on-demand via a user-supplied callback when deeper recursion is required.

---

<sup>1</sup>Source code in [63].

---

**Algorithm 2** HNMFk( $\mathbf{X}, K^{(0)}, d_{\max}, n_{\min}$ )

---

**Require:**  $\mathbf{X} \in \mathbb{R}_+^{n \times m}$ , candidate range  $K^{(0)}$ , depth limit  $d_{\max}$ , sample threshold  $n_{\min}$ .

```
1: Initialise queue  $\mathcal{Q} \leftarrow \{(\text{Root}, \mathbf{X}, K^{(0)}, 0, \emptyset)\}$ 
2: while  $\mathcal{Q} \neq \emptyset$  do ▷ breadth-first traversal
3:   Pop ( $\text{name}, \mathbf{X}_c, K^{(c)}, d, \mathcal{I}$ )  $\leftarrow \mathcal{Q}$ 
4:   if  $d = d_{\max}$  or  $\min |\mathbf{X}_c| \leq n_{\min}$  then
5:     continue ▷ declare leaf
6:   end if
7:    $(\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}}, \hat{k}) \leftarrow \text{NMFk}(\mathbf{X}_c, K^{(c)})$ 
8:   if  $\hat{k} = 1$  then
9:     continue ▷ no further separability
10:  end if
11:  clusters  $\leftarrow \arg \max(\widetilde{\mathbf{H}})$  ▷ or  $\arg \max(\widetilde{\mathbf{W}})$ 
12:  for  $c \leftarrow 1$  to  $\hat{k}$  do
13:     $\mathcal{I}_c \leftarrow \{j : \text{clusters}_j = c\}$ 
14:    if  $|\mathcal{I}_c| < n_{\min}$  then
15:      continue
16:    end if
17:     $K^{(c)} \leftarrow \text{NEXT RANGE}(\hat{k}, |\mathcal{I}_c|)$ 
18:     $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\text{name}_c, \mathbf{X}_{:\mathcal{I}_c}, K^{(c)}, d+1, \mathcal{I}_c)\}$ 
19:  end for
20: end while
21: return hierarchy  $\mathcal{T}$  with factors at every node
```

---

**Complexity.** For each node, HNMFk inherits the  $O(\hat{k}nm)$  cost of one NMFk fit (dominated by the multiplicative-update iterations) and adds  $O(\hat{k}m)$  for clustering. Parallel execution reduces the critical path to  $O(\text{depth} \times \text{max-node-time})$ , making HNMFk practical for matrices with millions of documents and vocabulary sizes well above  $10^5$  on commodity clusters.

## 2.6 Hierarchical Embedding Alignment Loss (HEAL)

HEAL fine-tunes a pretrained encoder so that its embedding geometry mirrors the multi-level cluster structure obtained with HNMFk. Documents (and synthetic queries) that share a common ancestor high in the hierarchy are pulled together, while those that diverge early are pushed apart. This is achieved by extending supervised contrastive learning with level-specific positive sets and depth-weighted penalties, injecting hierarchical semantics without altering the encoder architecture [21].

The label and positive-set definition are: For a mini-batch of  $N$  samples  $\{(x_i, \mathbf{y}_i)\}_{i=1}^N$  with depth- $L$  hierarchical labels  $\mathbf{y}_i = (y_i^{(0)}, \dots, y_i^{(L-1)})$ ,

$$\mathcal{P}(i, \ell) = \{p \mid y_p^{(\ell)} = y_i^{(\ell)}, p \neq i\},$$

defines the positive set for sample  $i$  at level  $\ell$ .

Level-wise contrastive loss with normalised embeddings  $\mathbf{h}_i = f_\theta(x_i)/\|f_\theta(x_i)\|_2$  and temperature  $\tau$ ,

$$\mathcal{L}_{i,\ell} = -\frac{1}{|\mathcal{P}(i, \ell)|} \sum_{p \in \mathcal{P}(i, \ell)} \log \frac{\exp(\mathbf{h}_i^\top \mathbf{h}_p / \tau)}{\sum_{a=1}^N \exp(\mathbf{h}_i^\top \mathbf{h}_a / \tau)} \quad (\mathcal{P}(i, \ell) \neq \emptyset).$$

Depth penalties are applied so that  $\lambda_0 > \lambda_1 > \dots > \lambda_{L-1}$ , where shallower (coarser) levels

receive larger weights,

$$\lambda_\ell = \frac{2^{L-\ell-1}}{2^L - 1}, \quad \sum_{\ell=0}^{L-1} \lambda_\ell = 1,$$

The overall loss is defined as:

$$\mathcal{L}_{\text{HEAL}} = \frac{1}{N} \sum_{\ell=0}^{L-1} \lambda_\ell \sum_{i=1}^N \mathcal{L}_{i,\ell}.$$

HEAL is optimised on batches containing both documents and optional synthetic queries, each inheriting the hierarchical label of its source document. Once training converges, the encoder  $f_{\theta^*}$  replaces the original embedding model in the retrieval index, giving hierarchy-aware nearest-neighbour search for downstream RAG.

## 2.7 Binary Search

Binary search is a divide-and-conquer algorithm for locating the position of a key  $x$  in a sorted array  $A[1 \dots n]$  in  $O(\log n)$  time, improving on the linear  $O(n)$  scan by systematically discarding half of the remaining search space at every step [118]. Starting with the full interval `low`= 1 and `high`=  $n$ , the algorithm iteratively computes the midpoint `mid`=  $\lfloor (\text{low} + \text{high})/2 \rfloor$ , compares  $A[\text{mid}]$  against  $x$ , and continues the search in the lower or upper sub-array accordingly. If  $A$  is immutable, the worst-case number of probes is  $\lceil \log_2 n \rceil + 1$ , which is asymptotically optimal for any comparison-based search on ordered data [18]. Its cache-friendly access pattern and logarithmic complexity make binary search a building block of fundamental algorithms such as interpolation search, fractional cascading, and balanced search trees [32]. Moreover, the monotone-predicate version extends beyond arrays, enabling efficient root finding, parametric search, and optimisation when the objective is unimodal or monotone in the decision variable [212]. Robust implementations guard against overflow in the midpoint computation and offer variants that return insertion points

for unsuccessful searches, facilitating duplicate handling and range queries [19].

## 2.8 Knowledge Graph Foundations

A KG is a heterogeneous, edge-labeled multi-graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$  in which each edge  $(h, r, t) \in \mathcal{E}$  asserts that the head entity  $h \in \mathcal{V}$  and the tail entity  $t \in \mathcal{V}$  participate in a semantic relation  $r \in \mathcal{R}$  [59, 89]. Nodes are typed by an ontology  $\mathcal{O}$  whose schema defines class hierarchies, permitted relations, and attribute vocabularies. Formally,  $\mathcal{O}$  is a first-order theory that supports sub-class (`is_a`) and sub-property reasoning, thereby enabling deductive closure ( $\mathcal{G} \mapsto \text{Cl}_{\mathcal{O}}(\mathcal{G})$ ) [86].

Downstream tasks such as entity resolution, rule mining, and knowledge graph embeddings depend on both the structural sparsity ( $|\mathcal{E}| \ll |\mathcal{V}|^2$ ) and the logical regularities imposed by  $\mathcal{O}$  [164].

Construction pipelines use (I) ontology engineering (schema design, URI conventions), (ii) information extraction (token classification, dependency-based relation detection, or LLM-driven triple generation), and (iii) canonicalization (string normalization, synonym collapse, co-reference clustering) [88].

## 2.9 Latent Feature Extraction via Matrix and Tensor Factorization for Knowledge Graph Construction

The rapid growth of unstructured data in scientific, technical, and societal domains has created a need for robust frameworks to organize and extract actionable knowledge. KGs have emerged as a powerful solution, structuring entities and their relationships in a way that enables reasoning, retrieval, and analytics. However, traditional KG construction methods, particularly those based on LLMs, face substantial challenges. These include hallucination, lack of domain control, diffi-

culty with schema alignment, and limited interpretability. To address these issues, an alternative paradigm based on matrix and tensor factorization has been proposed and successfully applied by multiple research groups.

### **2.9.1 Benefits of Matrix and Tensor Factorization Approaches**

#### **Interpretability and Semantic Coherence**

One of the most prominent advantages of matrix factorization techniques, particularly NMF, is their ability to uncover latent features in a non-negative and additive manner. These features naturally correspond to interpretable topics or semantic clusters, which are ideal for ontology formation in KGs. Unlike embedding-based approaches, where latent dimensions are opaque, NMF topics are readily traceable to sets of keywords or phrases, enabling transparent semantic labeling.

Hierarchical extensions such as HNMFk go further by organizing the latent space into coarse-to-fine topic trees, facilitating structured and multi-scale knowledge extraction. This recursive decomposition aligns naturally with ontological hierarchies and enables the construction of multi-layered KGs, where nodes represent concepts at various levels of abstraction and edges reflect subtopic relations.

#### **Multi-Modal Knowledge Graphs**

Another key contribution of tensor/matrix factorization-based KG construction is the integration of multiple modalities of information. Observable metadata (e.g., document authors, affiliations, citations, publication venues, etc.) can be combined with latent semantic features (e.g., topics, clusters, named entities) to form richly annotated KGs. In the Smart-SLIC architecture, for instance, TF-IDF matrices are constructed from both raw text and category labels, and their joint

factorization reveals topic-specific vocabularies and hidden relationships between documents.

This multimodal integration enhances the quality of the knowledge graph by grounding abstract topics in concrete data. It also facilitates domain-specific search, clustering, and missing link prediction, and improves explainability in contrast to LLM-generated graphs, which often lack such structured layering.

### **Resistance to Hallucinations and Structural Noise**

Perhaps the most critical benefit of factorization-based KG construction is its immunity to hallucinations. Unlike LLMs, which may generate plausible-sounding but factually incorrect triples, factorization-derived topics are extracted directly from co-occurrence statistics in the input corpus. This ensures grounding in actual, true content and reduces the risk of fabricating spurious entities or relations.

In systems like HEAL, where hierarchical topics guide additional semantic alignment in the Vector Store of the embedded documents that are in the same HNMFk leaf cluster, helps the retrieval-augmented generation. The use of factorization-derived labels significantly reduces false positive retrievals and improves factual consistency, tuning in the same time the utilized LLMs directly to the domain specific topic. Empirical evaluations have shown significant drops in hallucination severity and higher precision in question answering when KGs are built using latent topic models instead of LLM extractions [21].

This novel design also addresses additional weaknesses often found in LLM-generated KGs, including sparsity, ontological inconsistency, and ambiguity. By relying on interpretable and semantically structured decompositions, factorization-based methods ensure broader coverage of specialized topics, adherence to consistent taxonomies, and disambiguation of similarly named entities through

topic context.

## 2.10 Vector Stores for Semantic Retrieval

A vector store (VS) maintains a set of document embeddings, each represented as a  $d$ -dimensional vector  $\{\mathbf{z}_i\}_{i=1}^N \subset \mathbb{R}^d$  associated with corresponding document payloads [114]. Given a query embedding  $\mathbf{q} \in \mathbb{R}^d$ , the VS retrieves the  $k$  nearest neighbors according to a similarity metric  $S(\mathbf{q}, \mathbf{z}_i)$ , commonly cosine similarity or inner product [104]. For efficient querying, vector stores utilize approximate nearest-neighbor (ANN) indexing algorithms such as hierarchical navigable small-world graphs (HNSW) [146], inverted file indexing (IVF) combined with product quantization (PQ) [104, 11], or scalar quantization. These methods achieve sub-linear query latency and reduce computational costs while preserving high recall rates.

Typical semantic retrieval pipelines involving vector stores proceed through several distinct stages. Initially, documents undergo segmentation, frequently implemented through recursive text-splitting strategies designed to produce semantically coherent chunks adhering to token limits [128]. These segments are then transformed into embedding vectors using transformer-based language models, such as Bidirectional Encoder Representations from Transformers (BERT) or Dense Passage Retrieval (DPR) encoders [114]. The resulting embeddings are subsequently indexed via ANN structures, facilitating rapid and scalable retrieval operations [146, 104]. Lastly, hybrid scoring techniques are employed to merge lexical relevance scores (e.g., BM25 [186]) with dense semantic embedding scores. The combined scoring function is typically expressed as  $S_{\text{hyb}} = \lambda S_{\text{BM25}} + (1 - \lambda) S_{\text{dense}}$ , allowing systems to leverage complementary lexical and semantic strengths [125].

Vector stores play dual roles within contemporary retrieval architectures. First, they provide essential context for RAG, grounding large language models by supplying retrieved documents

relevant to a query or prompt, thus enhancing response quality [128]. Second, vector stores serve as foundational components in knowledge-graph construction processes, where embeddings can seed graph vertices or identify semantically overlapping textual segments for establishing relational links [114, 128].

### 2.10.1 Link Prediction in Relational Graphs

Let  $\mathcal{A} \in \{0, 1\}^{n \times n}$  denote the adjacency matrix of an undirected graph. **Link prediction** seeks a score matrix  $\hat{\mathcal{A}} \in [0, 1]^{n \times n}$  such that  $\hat{\mathcal{A}}_{uv} \approx \Pr[(u, v) \in \mathcal{E}]$ . Approaches fall into three categories:

- a) Heuristic indices (Common-Neighbors, Adamic–Adar, Katz) exploit local walk statistics but cannot model higher-order semantics.
- b) Graph-neural encoders learn node embeddings via message passing and decode edge existence with  $\sigma(\mathbf{h}_u^\top \mathbf{h}_v)$ , achieving state-of-the-art accuracy at the cost of significant GPU memory.
- c) Low-rank factorization approximates  $\mathcal{A} \approx \mathbf{W}\mathbf{W}^\top$  or  $\mathcal{A} \approx \mathbf{W}\mathbf{H}$ , where conditioning on the Frobenius norm supplies a convex objective that scales to millions of vertices on a single node. Boolean, weighted, and logistic variants accommodate binary, count, and implicit-feedback graphs, respectively.

Evaluation relies on edge-wise metrics (ROC AUC, PR AUC) and ranking metrics (MRR, Hits@K). Confidence awareness is increasingly required; bootstrap ensembles or Bayesian posteriors furnish per-edge uncertainty, enabling abstention policies when  $\text{Var}[\hat{\mathcal{A}}_{uv}]$  exceeds a risk threshold.

These three artifacts, knowledge graphs for structured reasoning, vector stores for semantic retrieval, and uncertainty-aware link prediction, constitute the infrastructural backbone upon which the subsequent thesis contributions are deployed.

## 2.11 LLM-Based Agent Reasoning

LLM-based agents significantly extend the capabilities of LLMs by integrating them with external environments and toolkits of callable APIs. This architecture, exemplified in systems like AutoGPT [184] and OpenAI’s function-calling interface [168], enables an agent not only to interpret user inputs or environmental observations, but also to explicitly reason about which tools to invoke in order to complete complex, multi-step tasks. At the core of such an agent is a policy  $\pi(c_t, m_t)$  that formulates each LLM prompt using the current context  $c_t$  and retrieved memory  $m_t$ , an environment  $\mathcal{E}$  that provides observations  $o_t$  and tracks state transitions, and a toolkit  $\mathcal{T}$  of callable tools or APIs for executing external actions.

At each interaction cycle, the agent first prepares the context by gathering the current observation and relevant memory, then assembles them into a structured prompt. It then uses chain-of-thought (CoT) prompting [231] to generate intermediate reasoning steps, explicitly documenting its deliberative process. Based on this reasoning, the agent either emits a final natural language response or invokes an external tool  $a_t \in \mathcal{T}$  to gather more information or perform an action. Upon receiving the outcome  $o_{t+1}$  of the tool execution, the agent updates its memory with the tuple  $(o_t, \text{CoT trace}, a_t, o_{t+1})$ , enabling iterative refinement and long-term context awareness.

Several architectural variants of this core agent loop have been proposed to improve reasoning robustness and task performance. ReAct agents [238] interleave explicit “Thought:” and “Action:” steps within their reasoning traces, enabling dynamic tool use with fine-grained control. Reflexion agents [195] add a self-reflection step after each action to critique and adjust their strategies. Self-Refine agents [145] perform full post-hoc reviews of their reasoning traces to iteratively correct errors before committing to final outputs. In more complex setups, multi-agent collaboration frame-

works [133] deploy multiple specialized agents, such as planners, coders, and critics, that exchange messages and coordinate to collectively solve complex tasks.

The performance of these LLM agents is typically evaluated along four key dimensions. Success rate measures the fraction of tasks completed correctly and reflects the agent’s overall reliability. Tool-usage efficiency measures the proportion of invoked tools that were both effective and necessary, thereby indicating cost-effectiveness. Reasoning depth refers to the average length and complexity of the chain-of-thought traces required to solve tasks, offering insight into the agent’s cognitive sophistication. Finally, latency captures the end-to-end time required per cycle, including both LLM inference and tool execution, which is critical for real-time or interactive applications. As LLM infrastructure continues to mature, researchers are exploring how to balance these trade-offs while improving memory strategies, improving response interpretability, and facilitating autonomous orchestration across environments and APIs. Together, these developments push LLM-based agents toward greater robustness, scalability, and autonomy in handling real-world workflows.

# Chapter 3

## Literature Review

### 3.1 Refined Data Collection

This section summarizes techniques and prior works applied in forging a highly specific dataset of research papers.

1) Topic Modeling & Tensor Decomposition: A common approach to topic modeling is NMF [173, 233], and when applied to a document-word matrix, identifies latent patterns of the corpus. Semantic NMF with automatic model determination (SeNMFk) is an NMF extension incorporating the text's semantic structure leveraged in [220, 218, 67]. Term frequency-inverse document frequency (TF-IDF) and the co-occurrence/word-context matrices are often used. Although common, more advanced methods exist.

2) Document Embeddings & Transformers: Vector representations of a text were previously used for dimensional mapping, cross-comparisons, and similarity analysis [121]. Standard models for learning word embeddings have been Global Vectors for Word Representation (GloVe)[179] and

Word2Vec [157]. Transformers have been used for large language models (LLM) as internal states, a popular example is BERT [52]. Here, the SciNCL transformer generates document embeddings [171], where the document mapping includes citation data.

3) Data Visualization and Tools: Several publicly available citation networks and topic modeling tools are available to explore and analyze research papers, such as Topic Modeling Tool [62], and Stanford Topic Modeling Toolbox [50]. While the tools gather topical data from inputs, they lack visualization. Alternatively, 'Connected Papers' (CP) has visuals and is a resource for discovering scientific literature [4]. While CP is applicable for co-citation and bibliographic literature exploration, this work's tool advances bibliographic utility by creating a specialized document dataset leveraging a citation network coupled with human-in-the-loop and machine learning. Another research visualization tool, designed to handle the apex of Covid-19 research production, is explained in Ref. [68]. The tool cleaned the text (tokenized, removed stop-words, & punctuation & capitalization), constructed a TF-IDF matrix, then reduced dimensions for graphing through t-distributed stochastic neighbor embedding (t-SNE). Here, Uniform Manifold Approximation and Projection (UMAP) [151] is used to reduce 768-dimensional embeddings output by the 'neighborhood contrastive learning for scientific document representations with citation graph embeddings' (SciNCL) model [171] to two dimensions.

4) Human in the Loop (HITL): User feedback has recently been adopted into several schemes, including OpenAI's ChatGPT [167] and Google's BARD [74]. In [73], a knowledge graph is built from text-prompting a user with feedback in every response to provide an acceptable retail-item recommendation. Differences between the Bibliographic Utility Network Information Expansion (BUNIE) method used in this dissertation and [73] exist, as the study's structure provides one recommendation, whereas BUNIE offers a dataset. Interactive modes also differ from ours in using

click-and-drag selection to delete rather than prompts. Furthermore, BUNIE removes papers at the HITL phase, whereas [73] requests positive and negative feedback about recommendations.

A HITL work more similar to BUNIE in Ref. [243] aims to build labeled image datasets for Computer Vision (CV) applications. A user labels a few images, which are extrapolated to all in the image’s cluster, and then the images are model-evaluated for reassignment. Like BUNIE, the process is iterated to convergence but differs in the direct user influence of datum retention.

## 3.2 Factorization Innovation

### 3.2.1 Knowledge Graphs

Recent studies have demonstrated the application of knowledge graphs in identifying emerging trends, uncovering hidden connections, and predicting future research directions within scientific domains [147, 181, 138, 224, 226, 194, 1, 42]. Integrating semantic analysis with graph-based structures allows for a more nuanced understanding of the thematic evolution in deep learning and cybersecurity [166, 196]. These advances highlight the potential of knowledge graphs to transform the exploration and exploitation of scientific literature, moving beyond traditional search mechanisms to enable a deeper, contextually rich engagement with content [10, 249]. To achieve these results, care must be given to the specific process for constructing and analyzing knowledge graphs, as studied and explained in [96, 41, 237, 225, 215].

Similar to this work, [49] scrapes the web for information relevant to the domain-specific knowledge. However, their domain is in medicine, and ours is in cybersecurity. Further, they do not use decompositions to construct the graph, but rather vocabularies. Similarly, [71] uses dates, topics, and events as entities to make the knowledge graph, but omits the use of decomposition to organize

large data. This work’s decompositions are enhanced by incorporating the semantic structure of the text, allowing for the estimation of the number of topics, which enables a coherent separation of the latent topics and accurate document clustering. This method is supported by the findings in [220, 67]. Building on [220, 67], this work’s approach is expanded through hierarchical decomposition and incorporates semi-supervised labels of arXiv categories in joint factorization to refine this work’s capabilities.

### 3.2.2 Hierarchical Topic Modeling

A standard method for hierarchical topic modeling is through a probabilistic approach, in which the corpus is treated as a mixture of topics that can be modeled with a probability distribution. One such method is hierarchical Latent Dirichlet Allocation (hLDA) using the nested Chinese Restaurant Process (nCRP) beforehand [75]. This model allows for arbitrarily large branching factors in topic hierarchies, with the nCRP aiding in inferring the appropriate depth of the topic tree. Another LDA-based approach is Hierarchical Dirichlet Processes (HDP). In this method, a set of Dirichlet processes is coupled via their base measure, which is distributed according to a Dirichlet process [213]. The authors introduce two Markov Chain Monte Carlo (MCMC) sampling schemes for posterior inference under hierarchical Dirichlet process mixtures and test the approaches on multiple datasets.

More recent approaches to hierarchical topic modeling make use of variational auto-encoders, recurrent neural networks, and transformers [109, 77, 222, 156, 76]. While successful, these methods face a common problem in deep learning: successful models heavily depend on the quantity and quality of the training data. HSNMFk-SPLIT is a favorable approach to hierarchical topic modeling since it is instance-based. Only the data that is being modeled is required for computation. [14]

### 3.3 Binary Directional Search

This section provides a brief review of several machine learning dimensionality reduction and clustering algorithms that require hyperparameter selection for  $k$ . Binary Bleed uses binary search to optimize  $k$ , so this work reviews related binary search studies. Finally, Binary Bleed optimizes the hyperparameter  $k$  over a search range, making an automatic model determination, or  $k$  selection, another foundation. Other works have optimized their respective algorithms for parallel and distributed contexts in binary search and automatic model determination.

Applications of  $k$  Search: Optimizing for  $k$  has utility in several domains, particularly clustering, where optimal cluster count search computations can be expensive. Typically,  $k$  is a user-provided parameter that must be refined through trial and error. Algorithms that require user-specified or trial-discovered  $k$  include K-means Clustering [144], K-medoids Clustering [115], K-medians Clustering [103], Fuzzy C-means Clustering [20], Mini-Batch K-means [193], Spherical K-means [90], Elkan K-means [60], NMF Clustering [233], Symmetric NMF Clustering [219], and RESCAL Clustering [165].

Each algorithm can use the silhouette score to determine the ideal  $k$  heuristically. In this work’s experiments, detailed in the results section, Binary Bleed’s performance is analyzed when operating with the silhouette score and Davies Boulding to test NMFk, RESCAL, and K-means algorithms.

Binary Search: Several aspects of binary search are relevant to this work, including optimizing search, modifying the binary search algorithm, and parallelizing the binary search process. Noisy Binary Search [113] minimizes the number of comparisons to find an optimal coin in a sequence of coin flips by using biased coin flips to compare elements in a sorted sequence indirectly. It identifies the position where the probability of observing heads changes from below to above a given target

threshold. Flip positions are determined from previous outcomes, focusing the flips on areas with the highest uncertainty about crossing the target probability. This approach is similar to Binary Bleed, which narrows searches in the search space using a thresholding mechanism combined with an observed score. Differently, while Noisy Binary Search selects the next target based on the score of the current item, Binary Bleed is designed for methods like NMF, where silhouette scores obtained from different  $k$  selections are independent of each other. Consequently, the silhouette score cannot decide the next  $k$  to visit. Instead, for the  $k$  search space pruning heuristic, Binary Bleed follows the hypothesis that the silhouette score will remain low after the ideal  $k$  due to the overfitting phenomenon. Once a stable  $k$  is found, all the lower  $k$  values are ignored, and higher or lower  $k$  values require visitation until overfitting is observed.

Another work, [217], sought to optimize the binary search algorithm by modifying it to check the bounds of a stack and adjust these bounds, thereby increasing or narrowing the current search space to make the search process more efficient. This was required to measure the resting position of human eyes, which changes over time. Similar to [217], a modified binary search was described by [33], which checks at both ends of a sub-array and the middle to reduce the number of iterations through many sub-arrays to find the search value. Both [217] and [33] reduce needed checks in binary search, similar to this work. However, this work precomputes the search space and iteratively prunes values. Binary Bleed does not terminate upon finding an optimal candidate. Instead, it adjusts the search space and continues exploring the parameter direction to further optimize the results.

Some works have achieved parallelization of the binary search with specific constraints. One work, [39], partitions the search array into smaller sub-arrays, searching each in parallel, which is similar to [33] but operates all arrays concurrently, and so the similarities to this work are the same with the addition of parallelizing the problem. Binary Bleed may operate parallel jobs across single

or multiple compute nodes, each handling a portion of the  $k$  search space. Another work, Parallel Binary Search in [3] defines two arrays,  $A$  and  $B$ , where  $|A| < |B|$ . Parallelization distributes  $A$ 's elements across resources to find the smallest element in  $B$  greater than or equal to each element in  $A$ . While this approach to parallelization is similar to parallelizing the  $k$  range in this work, the objectives differ. Binary Bleed aims to find the largest or smallest optimal value in the  $k$  space rather than minimizing array indices.

Importantly, distributed and parallel binary searches are often used interchangeably but address different challenges. In a parallel search, data is small enough to fit in memory, so evaluations of different  $k$  values are concurrent on other processors or nodes. In contrast, distributed search requires multiple processors or nodes to handle single  $k$  evaluations for data too large for memory. In the context of this work, comparing the search term to the current term  $n$  in binary search is analogous to model computation at  $k = n$ . This computation at  $k = n$  may exceed memory capacity for large datasets, necessitating a distributed solution where the data for a single calculation is divided among multiple resources. This uses Binary Bleed to minimize the  $k$  search space with distributed NMF from this work's prior work [23, 22]. This approach allows parallel evaluations to concurrently minimize  $k$ , while distributed evaluations manage the computation for each  $k$  in large datasets, ensuring efficient utilization of resources.

K Search: Several algorithms have optimized parameters or  $k$ , including those that utilize parallelization and distributed contexts. For textures, [206] uses distributed calculations from constraints and backtracking error correction, allowing bulk calculation with minor corrections. While this work can distribute an NMF  $k$  operation larger than memory, there is no need to backtrack in the  $k$  optimization. Either an optimal  $k$  is found or not, and can be done in parallel and/or distributed calculations.

In an alternative method to approximate the number of  $k$  in a dataset, [136] uses hash functions with prototypes. This approach adaptively chooses a set of prototypes and binary codes to represent the data sample distribution, approximating neighbor relationships. The distribution in this algorithm is derived from learning the weights of the hash codes rather than selecting  $k$ . Therefore, while distribution is utilized in their method, this work directly employs distribution and parallelization to determine the optimal  $k$  value.

Another optimization can be observed in the context of K-Nearest Neighbors (KNN) [83], where training examples are sorted into a binary search tree based on the two points that are the furthest apart in the dataset. The furthest data points are identified at each node to construct the next partition. This method attempts to optimize the  $k$  selection for KNN. However, it is peculiar to KNN rather than a general parameter search optimization method, as this work with Binary Bleed proposed.

Similar to [83], the parallel  $k$  search in [198] utilizes GPU-accelerated parallel processing and specific sorting algorithms, such as bitonic sort, that are efficient under synchronous operation conditions. While this approach reduces the number of operations required in a distributed environment, it is designed explicitly for algorithms that benefit from GPU operations and truncated sorting methods. In contrast, this work aims to optimize the  $k$  parameter more generally and does not rely on a specific model, sorting method, or primitive operation.

## 3.4 Law

This section reviews contributions across RAG domains, semantic search using vector embeddings, knowledge graph construction, NMF, and legal information systems.

### 3.4.1 NMF for Pattern Discovery

In the legal domain, NMF has proven valuable for analyzing complex textual data, such as case law and statutes, and facilitating topic discovery and clustering. For instance, [28] applied NMF to legal documents to extract latent topics and visualize relationships, demonstrating how NMF’s interpretable structure aids researchers in identifying underlying topics not readily apparent in raw text. They even applied NMF hierarchically to find fine-grained topics; however, they did not have a way to automatically approximate the number of clusters at each decomposition. More recently, [130] proposed a guided semi-supervised NMF approach for topic discovery in legal documents, using domain knowledge to steer factorization and ensure the extracted topics remain highly relevant. This semi-supervised extension bridges the gap between fully automated unsupervised techniques and expert-driven analysis. NMF’s utility in legal contexts is particularly significant, providing an interpretable framework for analyzing large textual datasets.

### 3.4.2 Retrieval-Augmented Generation

RAG has emerged as a foundational approach for improving AI systems across various domains, including law. For instance, [128] introduced a framework that dynamically retrieves relevant documents to ground generative outputs, achieving notable gains in accuracy. Building on this idea, [79] proposed a retrieval-augmented pretraining method that integrates external knowledge for improved downstream task performance. In contrast, [101] demonstrated the effectiveness of retrieval in open-domain question answering. These advances lay the groundwork for applying RAG to the legal sector, where the method’s ability to ground LLMs in authoritative texts reduces hallucinations and increases accuracy in tasks such as case law retrieval, statutory reasoning, and judgment prediction. Notable examples include CBR-RAG, which incorporates Case-Based Reasoning to

structure retrieval for legal question answering (QA)[232], and LegalBench-RAG. This benchmark suite tailors evaluation metrics to the demands of legal information synthesis [182]. Parallel work has demonstrated RAG’s capabilities in other domains, such as malware data analysis, by combining embeddings, KGs, and NMF [15]. Other works demonstrate how LLMs can dynamically decide when and what to retrieve to enhance legal reasoning [122] and how multi-step legal judgment prediction can benefit from iterative retrieval and generation [228], further underscoring the effectiveness of combining retrieval strategies with LLMs.

### 3.4.3 Semantic Search with Vector Embeddings

Semantic search operates on dense vector representations to find the deeper semantic relationships in texts, going beyond keyword matching to proper context retrieval [114, 79]. This search is especially valuable in the legal domain, where queries often demand conceptual rather than surface-level understanding. Domain-specific pretraining has become increasingly important, as seen in LEGAL-BERT [35], which outperforms general-purpose models by better capturing nuances of legal language. Benchmarks like LeCaRD [142] show the effectiveness of dense retrievers, such as SBERT, in legal case retrieval of Chinese law and the abilities of hybrid approaches that integrate both lexical and dense retrieval methods. Hierarchical transformer architectures [93] and long-context models like Longformer [17] further address the challenges of lengthy legal documents. In contrast, citation-driven approaches such as SPECTER [47] leverage metadata to improve retrieval. Furthermore, work on neighborhood contrastive learning for scientific document representations (SciNCL) [171] demonstrates how controlled nearest neighbor sampling over citation graph embeddings can provide continuous similarity. This strategy could also inform citation-based retrieval improvements in legal domains. Challenges persist in scaling, explainability, and integrating heterogeneous data sources

into real-world legal workflows.

### 3.4.4 Knowledge Graphs in Legal Research

In the legal domain, researchers have used KGs for knowledge extraction [203], constructed domain-specific graphs tailored to legal contexts [54], and explored their potential for addressing trust, privacy, and transparency concerns [31]. Recent studies have demonstrated the effectiveness of KGs in recommending similar legal cases [53], linking case law with statutes for improved retrieval [247], and using graphs to enhance legal case law search in Chinese legal systems [24]. Beyond retrieval, KGs have been incorporated into knowledge-aware machine reading systems for legal question answering [69]. These works demonstrate KGs' capacity to represent intricate legal concepts, structure regulatory frameworks, and encourage reasoning over legal data.

### 3.4.5 Legal Systems and Case Retrieval

Legal information systems have evolved rapidly with the advent of neural architectures and hybrid retrieval pipelines, enabling precise tasks such as precedent retrieval, statute matching, and judgment prediction [34, 185, 37]. Benchmarks like LeCaRD [142] and LexGLUE [36], along with LEGAL-BERT-based systems [35], have demonstrated the capability of these neural methods to improve accuracy in analyzing legal corpora. In particular, researchers have leveraged structured reasoning with transformers and graph representations to link statutes and precedents, as evidenced in the COLIEE competition [183]. Nonetheless, several limitations remain: data scarcity and jurisdictional bias continue to restrict the generalizability of such models. At the same time, resource-intensive retrievers like BERT-based cross-encoders [248] have challenges scaling to large-scale legal databases. Earlier works in juris-informatics have highlighted the potential of automating

legal reasoning and document analysis [9], laying the groundwork for modern approaches that fuse knowledge graphs, transformers, vector stores, and agent-oriented RAG pipelines to deliver more explainable and efficient legal research workflows.

## 3.5 Malware

Recent methods for building RAG-assisted [128] chatbot applications rely on unstructured text stored in vector databases for QA tasks [134]. Although the integration of KGs in AI systems is not novel [197], increasingly, researchers are leveraging them to improve LLM reasoning while simultaneously addressing the reliability issues [174, 175, 132]. Despite the benefits, integrating domain-specific knowledge into chatbots requires substantial effort. The prior work is reviewed for standard chatbot designs, domain knowledge integration in RAG pipelines, and the necessary steps for constructing KGs.

### 3.5.1 KGs in RAG Pipelines

Building a sophisticated chatbot requires knowledge of various research fields. Prior works rarely present a fully engineered system like ours. Instead, most efforts focus on improving specific aspects of RAG pipelines, e.g., retriever design [38, 114], query intent recognition [207], and KG reasoning [106, 139, 107, 129]. this work’s approach resembles past methods that leverage chain-of-thought [230] prompting on KGs [107, 139]; in conjunction with LLM-agents to enhance reasoning capabilities [239, 123, 188]. In addition to incorporating these state-of-the-art techniques, this work’s RAG pipeline modifies its retrieval method to use KNN with the Levenshtein metric as an entry point for context search, replacing the cosine distance. Also, a “highly-specific” knowledge base is constructed for targeted QA tasks.

Although expensive and time-consuming, a handful of prior works incorporate domain-knowledge into their RAG pipelines [202, 58, 150, 235]; however, the majority either use existing KGs built broadly on medical literature [202, 150]; or do not disclose any details regarding their dataset construction [235]. This work’s method is “highly-specific” because it was driven by subject matter expertise, which informed its dataset curation and cleaning techniques [201, 63].

### 3.5.2 KG Development

Developing knowledge graphs requires building a corpus, defining an ontology, and extracting the relevant entity-relation triplets from unstructured text.

**Corpus Building.** The term “highly-specific” and explain this work’s dataset collection method. A key feature of this work’s dataset collection is the use of unsupervised methods [63] to decompose corpora into document clusters with finer specificity than the author-provided tags available on open-access websites. This differs significantly from prior approaches [1, 58, 244]. Latent-topic information from this work’s NMFk method was leveraged to filter and select the most relevant data for this work’s knowledge base, as well as to prune documents based on citation information and embedding distances. This work’s text cleaning pipeline is informed by subject matter experts (SME) [63, 200], thus going beyond standard methods by incorporating expert-derived rules for document cleaning, e.g, acronym and entity standardization.

**KG Construction.** This work’s ontology is shaped by traditional methods, i.e., relying on SME design and capturing task-specific features. Latent information from the decomposition process introduced in this work [63] is incorporated into the knowledge graph as entities, representing an innovation over previous approaches. For entity and relation extraction, the methodology extends beyond conventional learning-based techniques [105] by leveraging recent advancements involving

LLM-agents [134, 241], rather than relying on traditional LLM prompting methods [223, 149, 245, 57]. This approach yields non-sparse KGs, meaning the average out-degree of entities [140, 43] is high. No prior work integrates these methods into their knowledge graph construction process.

### 3.5.3 Link Prediction

Matrix factorization (MF) techniques have been widely used for predicting missing links in networks by leveraging latent structure and graph embedding methods to infer potential but previously unseen connections [124, 40, 236, 178, 110, 161, 155]. In the context of NMF, early research addressed incomplete observations in user-rating data [116], introducing weighted NMF (WNMF) for collaborative filtering tasks. Further adaptations incorporated node attributes to tackle sparsity in link prediction, such as GJSNMF [210] and JWNMF [209]. In contrast, others focused on recommender systems using constraints [91] or federated learning [66, 65]. Ensemble NMF approaches [56] split large graphs into subproblems for better scalability, and other work introduced graph regularization or deep architectures [148, 240], often addressing extreme sparsity with additional complexity.

LMF [110] offers a probabilistic alternative for modeling binary or implicit feedback, with extensions that incorporate neighborhood structure [135, 137] or symmetry constraints [177] to capture local and global relationships in network data. Beyond real-valued methods, Boolean or binary factorizations are useful in logical structures, employing 0/1 constraints to represent observed links. Other work on Boolean Matrix Factorization (BMF) investigated logical and arithmetic operations [51]. This work integrates Otsu’s method [172] and k-means clustering [108] into Boolean factorization as thresholding mechanisms. Similarly, [8] employed Otsu’s method and k-means clustering to segment brain diffusion imaging based on a reduced feature space obtained using NMF. While [8] applied these techniques to classify the features of interest in the images, such as white matter,

gray matter, and cerebrospinal fluid, using the latent factors, this work’s approach differs in that it integrates them directly into the factorization process to impose Boolean constraints on the latent factors. Rank selection remains a critical challenge in these various MF formulations, with recent attempts at cross-validation [126], bootstrap-based stability analyses [29], or other regularization approaches [100] that studied balancing model complexity and predictive accuracy.

Besides missing link prediction in conventional networks, MF has significantly impacted AI reasoning within KGs [227, 30], where relationships among entities are represented as triple-based structures. Symbolic reasoning methods in KGs often suffer from scalability issues and incomplete knowledge, leading to the adoption of MF-based embedding techniques that uncover latent structures for knowledge base completion [164]. Decomposing adjacency matrices corresponding to KG relations allows AI systems to predict new connections, improving coverage and inferential power. These factorizations also benefit recommendation systems by fusing user-item interactions with explicit semantic links from the KG, offering more explainable recommendations [227]. In the life sciences, large-scale biological graphs have utilized MF to hypothesize previously unknown interactions among genes, proteins, or drugs [250], highlighting the versatility of factorization-based approaches for reasoning-driven discovery.

Despite the progress of MF in missing link prediction and AI reasoning, two key challenges remain. First, Uncertainty Quantification (UQ) is critical yet largely unaddressed. Most methods yield only point estimates and cannot measure how trustworthy each prediction is, although such measures are vital in high-stakes domains like healthcare and finance [135, 92]. Second, the problem of optimal rank selection persists, with underestimation or overestimation leading to suboptimal performance [126, 158]. To address these gaps, this work proposes novel Weighted, Boolean, and Recommender NMF $k$  factorization methods, each incorporating automated rank determination and

deterministic thresholding. These methods are further enriched through ensemble strategies and logistic components, and a bootstrap-based UQ mechanism is introduced that allows an abstention option for highly uncertain predictions. Combining thresholding, ensemble-driven rank selection, and confidence scoring aims to enhance the reliability, interpretability, and applicability of MF-based link prediction in classic networks and knowledge-graph-driven AI systems.

### 3.5.4 Similar Systems

Several academic and open source systems come close to the scope of this work by combining collection pipelines, structured knowledge graphs, and vector retrieval with generation. The comparison below follows the lifecycle of a knowledge-intensive system. It considers how each system collects and filters data, how it builds and maintains a knowledge graph and a vector store, how it handles missing structure or incomplete links, and how it performs retrieval augmented generation. The discussion then explains where this work differs and how it fills the remaining gaps with a tensor-structured view of retrieval and reasoning.

A first family of systems focuses on general-purpose retrieval and generation. REALM pre-trains retrieval together with masked language modeling, so lookup is active during pretraining, fine-tuning, and inference [79]. RAG joins dense retrieval with sequence-to-sequence generation over a vector index [128]. DPR is a strong dual encoder retriever that supplies relevant passages [114]. FiD aggregates many retrieved passages inside the decoder to strengthen answer synthesis [101]. ATLAS shows that pretrained retrieval augmented models can reach strong few-shot performance on knowledge-intensive tasks [102]. KILT frames system-level evaluation with a shared snapshot and a common interface, which improves comparability [180]. These systems are useful for open domain question answering and usually use a large collection of texts specifically curated

to represent broad knowledge across many subjects, and they do not prescribe expert-guided domain selection, explicit hierarchical semantics or a synchronized symbolic layer. This work of this dissertation keeps the strengths of dense retrieval and strong readers but it places domain definition and data structuring before indexing. Structure is added through an interpretable hierarchy and it synchronizes knowledge layers so that retrieval and reasoning can respect domain boundaries and semantics.

A second family increases the effectiveness of the data with its structure. KG FiD uses a knowledge graph into Fusion in the Decoder and uses graph topology with a graph neural network re-ranker for overall improvements in open domain question answering [242]. GraphRAG constructs a knowledge graph over a corpus, organizes it into hierarchical communities with pre-computed summaries, and uses graph-guided routing together with text retrieval for both local neighborhood search and global community-level reasoning [154, 153]. Recent hybrid approaches explicitly combine graph and vector paths so an answer can draw on both relational and textual evidence, as in HybGRAG and HybridRAG [127, 2]. Earlier graph and text fusion, such as GRAFT Net, blends structured knowledge base signals with unstructured passages for question answering [205]. These systems show that structure can improve retrieval and generation quality. They do not specify how to build a domain-specific corpus or how to expose an interpretable hierarchy that can be examined and reused across domains. This work fills that gap by placing domain selection up front and further add a hierarchical topic tree with interpretable factors and by keeping a synchronized graph and vector store that shares the same semantics.

Several open source frameworks operationalize a combined graph and vector store in end to end deployments. RAGFlow integrates a conventional vector pipeline with an ingestion stage that extracts entities and relations to construct a knowledge graph and then exposes that graph at query

time for multi-hop reasoning together with dense retrieval and re-ranking [98, 97]. LightRAG builds a knowledge graph during indexing, keeps the graph aligned with the embedding store, and provides local, global, and mixed retrieval modes with an API and a web interface [87]. SciPhi R2R adds an agent-driven deep research workflow on top of hybrid keyword and semantic search and automatic graph construction, which enables multi-step retrieval that pulls from both the vector index and the graph inside a production-oriented server [192, 191]. These frameworks demonstrate how to synchronize structured and unstructured data and how to route over them. They leave open three concerns that this work addresses. They do not dictate a domain-aware collection process rooted in expert seeds and citation-guided expansion. They do not provide a transparent hierarchical representation that is used by both indexing and routing. They rarely address missing links and uncertainty in the structured layer.

Collection and corpus expansion also differentiate systems. Domain-aware pipelines often begin with subject matter expert seeds, then expand over citation networks by following document-level embeddings. Citation informed encoders such as SPECTER and SciNCL learn representations that reflect inter-paper relatedness, which supports growth that stays on topic [47, 171]. Legal retrieval research also models connectivity in the citation graph, which can strengthen case retrieval as seen in work such as CaseLink [211]. The general-purpose RAG family does not specify a collection strategy. GraphRAG and the open source frameworks above can intake a supplied corpus, and they can extract entities and relations. They do not specify a controlled citation expansion loop or a principled pruning method that keeps the resulting dataset aligned with expert intent. This work fills that gap by starting with expert seeds, expanding along citation and embedding neighborhoods, and using citation-aware encoders for control. The result is a curated domain corpus that is auditable and that is ready for structured learning.

Once a domain corpus is collected, the question is how to expose its semantics in a form that is both inspectable and useful for retrieval. Neural clustering can discover coarse themes, but it does not always produce an interpretable multilevel structure. Hierarchical nonnegative factorizations and tensor models provide transparent structure learning. A generalized hierarchical nonnegative tensor decomposition yields multi-level topic trees that can act as labels for routing and supervision [221]. Scalable NMF and RESCAL implementations such as pyDNMFk and pyDRESCALk add automatic model selection and distributed CPU and GPU execution which makes hierarchical factorization practical at large scale [26, 22]. This work adopts that path so that topics and subtopics can be named, inspected, and versioned. It then aligns the embedding space to that hierarchy so the vector store reflects meaning at multiple levels. HEAL introduces a hierarchical embedding alignment loss that uses labels from hierarchical clustering or factorization and improves retrieval relevance and downstream classification in domain specific retrieval augmentation [21]. The result is a dual data representation where the graph and the vector store share the same semantics and where routing can rely on explicit structure as well as similarity.

Handling missing structure is a further gap in many systems. The research prototypes and the open source frameworks above focus on retrieval and generation over a fixed graph. They often do not include uncertainty-aware completion of missing edges or scores that enable abstention when the structure is unknown. This work addresses that gap by adding tensor and matrix factorization for link prediction in the knowledge graph. That capability supports filling in missing links with calibrated scores. It improves recall for multi-hop questions, and it relaxes strict behavior when the structured layer is sparse. The factorization stage integrates naturally with the interpretable hierarchy and with the synchronized vector store so completion is both grounded and auditable.

Prior work typically pairs a retriever with a reader. For example [79], [128], and [114]. Later

variants introduce generative readers [101] and scale to large collections [102]. Benchmarks such as KILT emphasize evidence grounding and evaluation [180].

More recent systems route between text and graph retrieval. Examples include knowledge graph augmented retrievers [242] and GraphRAG style pipelines [154]. Open implementations extend this direction [153]. Prior graph and text hybrids remain relevant [205] and newer hybrid RAG designs continue this trend [2, 127].

The present approach targets the same objective of selecting the best evidence per query while differing in how evidence is organized and chosen. A tensor structured view of retrieval and reasoning enables an agent to do three things. It can walk a symbolic graph. It can issue dense vector searches. It can invoke link prediction when the structure is missing. An induced topic tree supplies an interpretable routing signal across these modes.

This design treats the collection as a first-class object and unifies structure induction, completion, and retrieval. It adds and synchronizes a human-interpretable hierarchy that organizes the graph and vector store. It completes the missing structure as needed and uses tensor structured RAG to route across modes. The result is effective and transparent and is designed for domains such as legal practice, where domain tuned encoders such as Legal BERT are valuable components but not a complete structured system [35].

# Part II

## Methods

## Chapter 4

# Data Acquisition

This chapter defines the corpus acquisition pipeline that supports every subsequent component of the dissertation. The document collection is assembled through citation-network expansion, expert filtering, and controlled vocabulary alignment, producing a broad yet pertinent input set. This corpus serves as the input to the Binary Bleed decomposition, HNMFk taxonomy induction, and Tensor-Structured RAG modules, which convert the text into provenance-aware knowledge graphs and aligned vector stores. The acquisition workflow meets two objectives: it maximizes recall so that emerging signals are retained, and it maintains the topical coherence and lexical consistency required for the low-rank matrix and tensor operations that drive the retrieval-and-reasoning pipeline.

### 4.1 Citation–Network Expansion and Initial Pruning

The workflow begins by seeding a compact set of publications and expanding outward along citation edges to create a broad but thematically connected corpus. Throughout this process, human expertise and algorithmic screens cooperate to eliminate irrelevant material before it propagates

downstream.

### 4.1.1 Seeding the Core

An SME selects a handful of “core” papers that exemplify the target topic. Each core item is supplied as a unique identifier (e.g., DOI) and resolved via the Semantic Scholar API [117] to collect metadata such as the title, abstract, citations, and references. Because subsequent expansion follows citation links, the citation influence of the seed set governs the eventual corpus size: a single highly cited paper may spawn thousands of descendants, whereas several niche articles may yield a modest network.

### 4.1.2 Traversing the Citation Graph

Let  $X$  denote the current set of papers and  $X^c$  the set of all citations made by papers in  $X$ . One network hop is defined by

$$X \leftarrow X \cup X^c,$$

adding every cited work to the corpus. Repeating the operation performs a breadth-first traversal of the directed citation graph. Users may instead traverse references (i.e., reverse the edge direction) when the core comprises recent or sparsely cited work. The hop count is user-controlled, providing a stop condition that balances recall against computational load.

### 4.1.3 Iterative Pruning Strategy

Citation expansion allows tangential literature, through acknowledgments of foundational but off-topic publications, review articles, and so on. To inhibit topical drift, three complementary pruning mechanisms are applied after each hop.

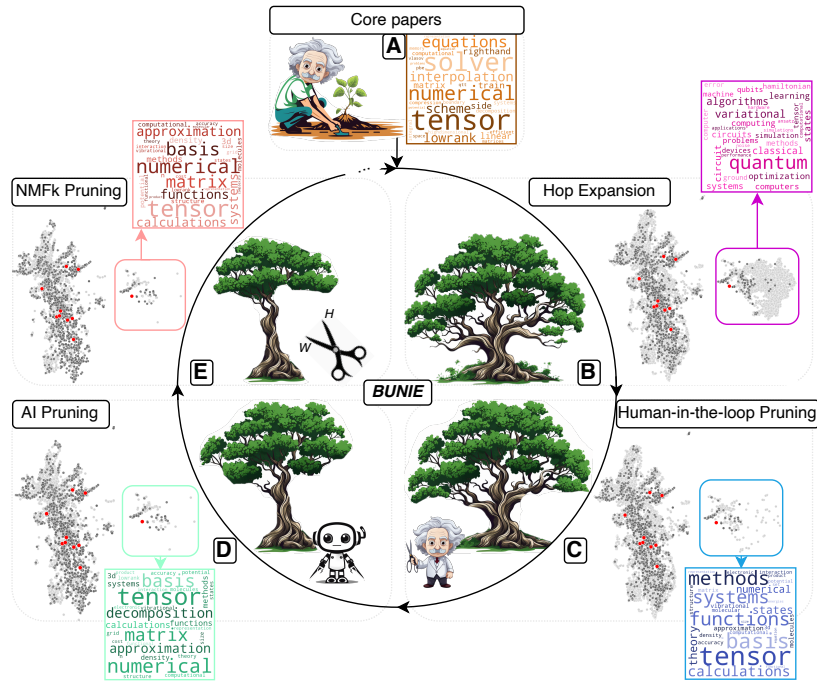


Figure 4.1: BUNIE’s distillation pipeline. **Panel A:** The SME selects specific papers, shown as a bag-of-words wordcloud. **Panel B:** The dataset expands through the citation network. The wordcloud of a selected subset shows how the vocabulary differs from the core. **Panel C:** Human-in-the-loop pruning removes data. The wordcloud begins to resemble the core. **Panel D:** An embedding heuristic removes papers far from the core. The dataset becomes more compact. The wordcloud approaches parity with the original. **Panel E:** NMFk topic modeling produces H-clustering factorization to remove clusters lacking a core paper. A dense, yet substantial set of documents indicates a successful distillation. The wordcloud closely resembles the original. As indicated by the **Ellipse**, iterations can be made for refinement or the data extracted in repose for downstream analysis.

## Human-in-the-Loop Screening

High-dimensional embeddings of concatenated title–abstract text are computed with SciNCL [171] and projected to two dimensions via UMAP [151]. An interactive GUI lets SMEs draw lassos around visible clusters, inspect a bag-of-words word-cloud, and view full metadata. Irrelevant clusters are deselected before the next hop.

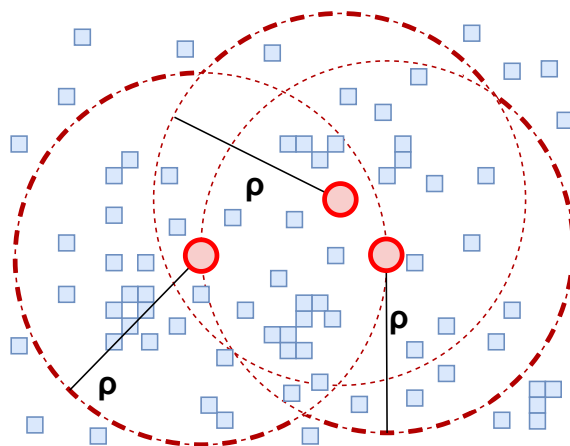


Figure 4.2: 2-dimensional representation of hypersphere pruning. Core papers are red dots, citations/ references are blue squares, and hyperspheres are dotted red circles.

### Hypersphere Pruning in Embedding Space

To retain the geometric structure lost in the UMAP projection, pruning is repeated in the original 768-dimensional space. Let  $E = \{e_1, \dots, e_n\}$  be the embeddings of the  $n$  core papers. Compute all pairwise Euclidean distances, collect them in  $D$ , and set

$$\rho = \text{median}(D).$$

Each core embedding becomes the centre of a hypersphere of radius  $\rho$ ; a candidate paper is accepted if its embedding lies within at least one hypersphere (Figure 4.2). The rule adapts to the intrinsic spread of the seed set and requires no manual tuning.

### Topic-Model Consistency

The remaining documents are subjected to NMFk fuzzy clustering. Factorisation of the TF-IDF matrix  $\mathbf{X} \in \mathbb{R}_{\geq 0}^{m \times n}$  yields

$$\mathbf{X} \approx \mathbf{W}\mathbf{H},$$

where  $\mathbf{W} \in \mathbb{R}_{\geq 0}^{m \times k}$  encodes word-topic affinities and  $\mathbf{H} \in \mathbb{R}_{\geq 0}^{k \times n}$  gives topic intensities per document. Documents dominated by topics that contain no core papers are discarded. A fixed vocabulary, established from the seed set and augmented with SME-defined term consolidations, prevents vocabulary drift and keeps computational cost stable.

#### 4.1.4 Quantifying Cohesion

To monitor the tightening of topical focus, a compactness score

$$C = \frac{1}{n(n-1)} \sum_{i \neq j} |\cos \angle(\mathbf{e}_i, \mathbf{e}_j)|$$

is computed over the cosine similarities of all document embeddings.  $C \in [0, 1]$  rises as the corpus becomes more semantically coherent, providing a quantitative checkpoint before further processing.

Through citation-graph expansion coupled with iterative pruning in embedding and topic space, the procedure yields a large yet thematically faithful corpus ready for the downstream semantic filtering and LLM-assisted pruning stages described in the following sections.

## 4.2 Semantic Query Filtering

Once a broad, citation-expanded corpus has been pruned, the next step is to extract sub-collections aligned with specific research intents (e.g., “laser and deflection, but not green”). This is accomplished through a transparent and auditable semantic filtering mechanism that supports Boolean logic,  $n$ -gram matching, and metadata-aware constraints.

### 4.2.1 Index Construction

For each field within the corpus, such as title, abstract, publication year, authorship, and affiliation, a corresponding inverted index is constructed. Textual fields are normalized through basic preprocessing steps, including lowercasing, punctuation removal, and whitespace normalization. Each normalized token is then mapped to the set of documents in which it appears. Non-textual fields, including categorical and numerical metadata, are indexed using similar mappings. This enables efficient lookups and rapid set-based filtering during query execution.

### 4.2.2 Query Structure

A query consists of an ordered sequence of clauses. Each clause can take one of the following forms:

- A single token, such as `laser`;
- A phrase of multiple tokens in natural word order, such as `angle deflection`;
- A structured constraint, e.g., `"laser": "node", " + blue"` , specifying that documents must contain `laser` and `blue`, but not `node`.

Clauses are combined using intersection (default, AND-search) or union (OR-search). An optional

linked-search mechanism allows reinforcing signals from one clause to override vetoes from another, which is particularly useful in noisy or polysemous contexts.

### 4.2.3 *n*-Gram Matching by Sliding Window

Multi-token phrases are resolved using a sliding window algorithm. Given a query phrase of length  $w$ , the algorithm iterates through all contiguous subsequences of  $w$  tokens in a candidate document. A match is declared if either the tokens appear in exact order or, for unordered matches, the multisets of query and window tokens are equal. This algorithm operates in  $O(mw)$  time per document, where  $m$  is the number of tokens in the document and  $w$  is typically small.

### 4.2.4 Fusion of Metadata and Textual Constraints

At evaluation time, the system constructs two candidate sets:

1.  $S_{\text{meta}}$ , consisting of documents that satisfy all specified metadata constraints;
2.  $S_{\text{text}}$ , consisting of documents matching the lexical clauses of the query, further refined by phrase validation.

The result set is obtained by intersecting  $S_{\text{meta}}$  and  $S_{\text{text}}$ , or by returning  $S_{\text{meta}}$  alone if no textual query is present. Because both sets are derived through precomputed indices, execution scales with the size of the smaller candidate set, making the process efficient and predictable.

This semantic filtering stage enables rapid and context-sensitive extraction of documents that match both lexical content and structured metadata constraints, forming the input to downstream embedding and LLM-based pruning stages.

## 4.3 Embedding and LLM-Based Corpus Pruning

While lexical and metadata filtering effectively remove many irrelevant entries, they remain insufficient for eliminating semantically off-topic or conceptually distant documents. To refine the corpus for downstream semantic analysis, two complementary pruning techniques are employed: one based on geometric proximity in embedding space, and the other on probabilistic voting within the latent space of an LLM. Both techniques are designed to preserve transparency by generating structured, machine-readable justification logs.

### 4.3.1 Geometric Outlier Detection in Embedding Space

Let  $\mathcal{C}$  denote the candidate corpus and  $\mathcal{R} \subset \mathcal{C}$  a trusted reference subset (e.g., documents manually verified or labeled by domain experts). A semantic encoder  $f : \mathcal{C} \rightarrow \mathbb{R}^d$  maps each document to a  $d$ -dimensional vector space, producing embeddings  $\{\mathbf{v}_i\}_{i \in \mathcal{C}}$ .

Define the centroid of the reference set as

$$\boldsymbol{\mu} = \frac{1}{|\mathcal{R}|} \sum_{j \in \mathcal{R}} \mathbf{v}_j,$$

and compute the average radial distance and its variance:

$$\bar{r} = \frac{1}{|\mathcal{R}|} \sum_{j \in \mathcal{R}} \|\mathbf{v}_j - \boldsymbol{\mu}\|_2, \quad \sigma^2 = \frac{1}{|\mathcal{R}|} \sum_{j \in \mathcal{R}} (\|\mathbf{v}_j - \boldsymbol{\mu}\|_2 - \bar{r})^2.$$

A candidate document  $i \in \mathcal{C}$  is retained if it satisfies the threshold:

$$\|\mathbf{v}_i - \boldsymbol{\mu}\|_2 \leq \bar{r} + k\sigma,$$

where  $k$  is a tunable parameter (default  $k = 3$ ). This scale-invariant rule provides a nonparametric and distribution-agnostic method for identifying embedding outliers.

### 4.3.2 Probabilistic LLM Voting

Although the geometric filter is efficient, it cannot capture higher-level semantic cues or domain-specific discourse structures. To address this, each document excluded by the geometric criterion is evaluated by a probabilistic LLM-based voting mechanism, which determines its relevance through contextual reasoning.

#### Prompt Construction

Each vote is cast using a constrained prompt composed of:

- 1) A set of  $m$  reference abstracts  $\mathbf{x}_{1:m}$  drawn at random from  $\mathcal{R}$  (typically  $m = 10$ );
- 2) A candidate abstract  $\mathbf{z}$  drawn from the rejected set  $\mathcal{C} \setminus \mathcal{R}$ ;
- 3) An instruction block enforcing output in a structured JSON format containing keys **answer**  $\in \{\text{yes, no}\}$  and **reason**.

Post-processing ensures that invalid or incomplete responses are discarded and, if necessary, resampled.

#### Monte Carlo Acceptance Test

Each candidate document is submitted to  $n$  independent LLM invocations at a fixed temperature  $\tau$ , producing a sequence of binary votes  $B = \{b_1, \dots, b_n\}$ , where  $b_i = 1$  indicates a positive relevance decision. Define the empirical vote proportion as

$$\hat{p} = \frac{1}{n} \sum_{i=1}^n b_i.$$

A candidate is accepted into the final corpus if  $\hat{p} \geq p_{\text{th}}$ , where  $p_{\text{th}}$  is a predefined threshold (typically 0.6). This ensemble-based approach enables robustness against noisy or ambiguous predictions while leveraging the deep contextual awareness of the LLM.

## 4.4 Vocabulary Standardization: Acronym Resolution and Lexical Consolidation

Even after semantic pruning, domain-specific corpora often exhibit surface-level lexical fragmentation, wherein semantically identical concepts are expressed in multiple forms (e.g., “partial differential equation,” “PDE,” “P.D.E.”). This redundancy inflates the vocabulary, obscures term frequencies, and degrades the performance of downstream matrix- or graph-based learning methods. To address this, we apply a two-stage standardization process: (1) acronym resolution and canonicalization based on syntactic cues, and (2) lexical consolidation of morphologically similar or near-duplicate tokens based on string similarity. Both stages produce traceable substitution mappings that support downstream auditing or expert override.

### 4.4.1 Acronym Detection and Canonicalization

#### Detection Strategy

To identify acronym–definition pairs, all  $(n+1)$ -grams ( $2 \leq n \leq 8$ ) are scanned for patterns in which a contiguous sequence of  $n$  words is immediately followed or preceded by a candidate acronym. Specifically, let  $g = \langle w_1, \dots, w_{n+1} \rangle$  be a candidate phrase. The following conditions are tested:

- (a)  $w_{n+1}$  is an acronym whose letters match the initials of  $w_1$  through  $w_n$ ;

(b)  $w_1$  is an acronym whose letters match the initials of  $w_2$  through  $w_{n+1}$ .

When a valid match is found, the full phrase is mapped to the corresponding acronym, and optionally replaced with a single, joined token to preserve conceptual integrity during tokenization. This replacement ensures that multi-word expressions are treated as atomic lexical units in subsequent analysis.

### Standardization Output

Across the corpus, acronym mappings are aggregated into a unified substitution table of the form *full\_form*  $\mapsto$  *canonical\_token*. This table governs all replacements applied to the corpus and may be reviewed or modified by subject-matter experts to correct overgeneralizations or contextual exceptions.

## 4.4.2 Data-Driven Lexical Consolidation

### Motivation

Technical corpora often contain morphological variants, typographical errors, or inconsistent stylistic conventions that fragment concept representations across tokens (e.g., *tokenize/tokenisation*, *normalisation/normalization*). These redundancies degrade the sparsity structure of lexical matrices and dilute the significance of term frequencies.

### Partitioned Preprocessing

Let  $\mathcal{V} = \{t_1, \dots, t_{|\mathcal{V}|}\}$  denote the full vocabulary. To limit the quadratic cost of pairwise comparisons, preprocessing partitions  $\mathcal{V}$  into buckets based on:

- 1) **Initial character:** Tokens sharing the same first letter are grouped together, reducing the pair space to  $\sum_{\alpha} |B_{\alpha}|^2$ .
- 2) **Length bands:** Within each initial character group, only tokens whose lengths differ by at most  $\Delta\ell$  (typically  $\Delta\ell = 2$ ) are compared, thereby retaining plausible edit sequences.

### Similarity Scoring and Filtering

All unordered token pairs within each group are compared using normalized Levenshtein similarity:

$$s(t_i, t_j) = 1 - \frac{d_L(t_i, t_j)}{\max(|t_i|, |t_j|)},$$

where  $d_L$  denotes the Levenshtein edit distance. Pairs exceeding a similarity threshold  $\tau$  (default  $\tau = 0.90$ ) are flagged for consolidation. To avoid erroneous merges in ambiguous cases ( $\tau - \varepsilon \leq s < \tau + \varepsilon$ ), suffixes such as **-ing**, **-tion**, and **-ness** are stripped before re-evaluation.

### Canonical Form Selection

For each high-similarity pair  $(a, b)$ , the preferred canonical form is chosen based on global corpus frequency. The less frequent token is then mapped to the selected form. The resulting collection of mappings defines a one-to-one substitution function that resolves lexical redundancy across the corpus.

### Impact

This two-stage vocabulary standardization pipeline enforces terminological coherence, reduces vocabulary dimensionality, and improves the quality of input matrices for subsequent factorization and knowledge-graph construction. The use of transparent substitution rules ensures interpretability and allows for controlled overrides in sensitive applications.

## 4.5 Conclusion

The methods described in this chapter expand a small set of expert-identified seed papers into a large-scale corpus comprising millions of documents, with standardized vocabulary, recorded provenance, and complete auditability. This fulfills the role of the chapter within the broader framework of the dissertation: to produce a clean, structured input suitable for downstream processing. The resulting corpus enables efficient decomposition of sparse term–document matrices by Binary Bleed, supports the construction of interpretable topic hierarchies through HNMFk, and provides consistent evidence streams for the event-driven infrastructure that maintains alignment between the KG and VS. With the data acquisition phase complete, the subsequent chapters shift focus to semantic structuring and hybrid inference, advancing the dissertation’s central aim of building scalable, reliable systems for knowledge retrieval and reasoning.

## Chapter 5

# Latent Feature Extraction & Topic Modeling

With the corpus established in the previous chapter, the next step is to uncover the latent structure that supports the remaining components of the system. This chapter presents the semantic structuring process, beginning with Binary Bleed, a recursive rank-selection method that identifies an efficient low-rank representation of the term–document matrix while maintaining scalability for large corpora. This selected rank initializes HNMFk, which separates domain-relevant topics across multiple levels of detail while preserving interpretability for subject-matter experts. These numerical factors are then connected to clear, human-readable labels using a language model alignment pipeline based on vector similarity. Together, these procedures transform raw term–document statistics into a structured and interpretable taxonomy, fulfilling the role of semantic structuring described in the abstract and enabling the synchronized construction of the KG and VS.

## 5.1 Binary Bleed

The method is named Binary Bleed due to its method of pruning less optimal  $k$  values, extends traditional binary search techniques by continuing to search even after a potential optimal  $k$  is found, ensuring the maximization or minimization of the scoring function  $f : \mathcal{K} \times \mathcal{D} \rightarrow \mathbb{R}$ , where  $\mathcal{K}$  is the set of candidate parameters and  $\mathcal{D}$  represents the dataset. Binary Bleed operates recursively to identify the optimal  $k$  that maximizes or minimizes an evaluation score function  $f(k, \mathcal{D})$ . Unlike a conventional binary search that terminates upon finding a target value, Binary Bleed dynamically adjusts the search space based on an evaluation threshold. From an initial range  $[k_{\min}, k_{\max}]$ , it computes the mid-point

$$k_{\text{mid}} = \left\lfloor \frac{k_{\min} + k_{\max}}{2} \right\rfloor$$

and evaluates the score

$$f_{\text{mid}} = f(k_{\text{mid}}, \mathcal{D}).$$

If  $f_{\text{mid}}$  meets the threshold, the search continues in the direction of optimization. For maximization,

the lower bound is updated to  $k_{\min} = k_{\text{mid}} + 1$  when  $f(k_{\text{mid}} + 1, \mathcal{D}) > f_{\text{mid}}$ , and conversely, the upper bound is updated to  $k_{\max} = k_{\text{mid}} - 1$  when  $f(k_{\text{mid}} - 1, \mathcal{D}) < f_{\text{mid}}$ . For minimization, the process is reversed. This recursive update continues until convergence or until a predefined stopping criterion is met, allowing the algorithm to “bleed” into higher or lower  $k$ . This ensures thorough exploration and optimization of the parameter  $k$ , adapting the search space dynamically to achieve the best possible evaluation score within the specified constraints.

This search does not cease upon finding a  $k$  above the threshold; instead, it continues exploring to ensure that no better solutions exist, particularly focusing on larger  $k$  values that may yield higher scores. The runtime complexity of Binary Bleed is bounded by  $\Theta(n^{\log_2(p+1)}) \times (T_{\text{model}} + T_{\text{score}})$

in both the best and worst cases, as shown later in 5.1, where  $p$  is the probability of recursing twice. We apply binary search on the ordered set  $\mathcal{K}$ , the search space, to find its optimal value. Mathematically, this optimization can be expressed as:

$$k_{\text{optimal}} = \max \{k \in \{1, 2, \dots, K\} : S(f(k)) > T\}$$

where  $T$  is the selection threshold,  $f(k)$  represents the model computation, and  $S(f(k))$  is the model's scoring function.

---

**Algorithm 3** Binary Bleed  $k$  Search, Single Rank & Thread

---

**Require:**  $\mathcal{K}$  (list of  $k$ ),  $data$  (dataset),  $model$  (model for calculation),  $scorer$  (function to quantify output),  $T_{\text{select\_k}}$  (minimum score for optimal  $k$ )

**Ensure:** Optimal  $k$  value and its prediction score

```

1:  $rank_{\text{seen}} \leftarrow \emptyset, k_{\text{max}} \leftarrow \infty, k_{\text{min}} \leftarrow -\infty$ 
2: function BINARYBLEEDKSEARCH( $\mathcal{K}, i_{\text{left}}, i_{\text{right}}, scorer, data, model, T_{\text{select\_k}}$ )
3:   if  $i_{\text{left}} \geq i_{\text{right}}$  then return
4:   end if
5:    $middle \leftarrow i_{\text{left}} + \lfloor (i_{\text{right}} - i_{\text{left}}) / 2 \rfloor$ 
6:    $k_{\text{middle}} \leftarrow \mathcal{K}[middle]$ 
7:   if  $k_{\text{min}} < k_{\text{middle}} < k_{\text{max}}$  then
8:      $score \leftarrow scorer(model(data, k_{\text{middle}}))$ 
9:     Append  $(k_{\text{middle}}, score)$  to  $rank_{\text{seen}}$ 
10:    if  $score \geq T_{\text{select\_threshold}}$  then
11:       $k_{\text{min}} \leftarrow k_{\text{middle}}$ 
12:    end if
13:    if  $score \leq k_{\text{stop\_threshold}}$  then

```

```

14:          $k_{\max} \leftarrow k_{\text{middle}}$ 
15:     end if
16: end if
17: if  $middle + 1 \leq k_{\max}$  then
18:     BINARYBLEEDKSEARCH( $\mathcal{K}, k_{\max}, k_{\min}, middle+1, i_{\text{right}}, ranks_{\text{seen}}, scorer, data, model, T_{\text{select\_k}}$ )
19: end if
20: if  $middle - 1 \geq k_{\min}$  then
21:     BINARYBLEEDKSEARCH( $\mathcal{K}, k_{\max}, k_{\min}, i_{\text{left}}, middle-1, ranks_{\text{seen}}, scorer, data, model, T_{\text{select\_k}}$ )
22: end if
23: end function

```

---

The algorithm is presented in Algorithm 3, showcasing its operational structure. To determine the optimal  $k$ , a list of  $k$  ( $\mathcal{K}$ ) is provided with the dataset, model, scorer, and score threshold. Initially, the set of visited  $k$ , the maximum ( $k_{\max}$ ), and minimum ( $k_{\min}$ ) bounds are initialized (lines 1-2). The algorithm checks the base case for recursion: if the left index  $i_{\text{left}}$  is greater than or equal to the right index  $i_{\text{right}}$ , the function returns, terminating the recursive search when no more  $k$  values are left to explore (lines 3-4). The optimal  $k$  is sought by computing the middle index of  $\mathcal{K}$  and its  $k$  (lines 5-6). This value is validated to ensure it is within  $k_{\min}$  and  $k_{\max}$  (line 7). The model is evaluated at this middle  $k$  on the dataset, and the score is added to the set of visited  $k$  values (lines 8-9). If the score at  $k_{\text{middle}}$  meets or exceeds the threshold,  $k_{\min}$  updates to  $k_{\text{middle}}$ , focusing on larger values (lines 10-12). If the score is below the stop threshold,  $k_{\max}$  updates to  $k_{\text{middle}}$ , focusing on lower values (lines 13-15). The BinaryBleedKSearch function is recursively called on the narrowed  $k$  range (lines 17-19). This dynamically recursive search space adjustment allows thorough exploration of  $k$ , ensuring the optimal is found based on the evaluation criteria.

Recursive Behavior Analysis: Let  $T(n)$  denote the runtime of the algorithm for  $\mathcal{K}$  of size  $n$ , and

$T_M$  and  $T_S$  are the runtimes of the model and scorer operating on data,  $\mathcal{D}$ . Table 5.1 asymptotically itemizes the algorithm. Truncation of the search space occurs with a probability  $p$  of two recursions and  $1 - p$  of one recursion. Therefore, the recurrence relation is:

$$T(n) = p \cdot 2T\left(\frac{n}{2}\right) + (1 - p) \cdot T\left(\frac{n}{2}\right) + O(T_M + T_S)$$

The model and scorer operate on  $\mathcal{D}$ , so their runtimes do not scale with  $n$  and thus can be represented as  $O(1)$  for each subproblem. This simplifies the equation to:

$$T(n) = (p + 1)T\left(\frac{n}{2}\right) + O(1)$$

Applying the Master Theorem [48] with  $a = p + 1$  (number of recursive calls),  $b = 2$  (factor by which the problem size is divided), and  $f(n) = O(1)$ :  $\log_b a = \log_2(p + 1)$  For  $\log_2(p + 1) < 0$  to hold,  $p$  would need to be less than 0, probabilistically impossible. Therefore, we have  $c < \log_2(p + 1)$ , and case 1 of the Master Theorem applies:

$$T(n) = \Theta(n^{\log_2(p+1)})$$

Thus, Binary Bleed's asymptotic runtime is  $\Theta(n^{\log_2(p+1)})$ .

Step	Line	Complexity
Initialization	1-2	$O(1)$
Base Case	3-4	$O(1)$
Middle/Check	5-7	$O(1)$
Model/Score	8	$T_M + T_S$
Update Min/Max	10-16	$O(1)$
Bound Check	17,20	$O(1)$
Recursion	18-23	-

Figure 5.1: Primitive operations for asymptotic analysis.

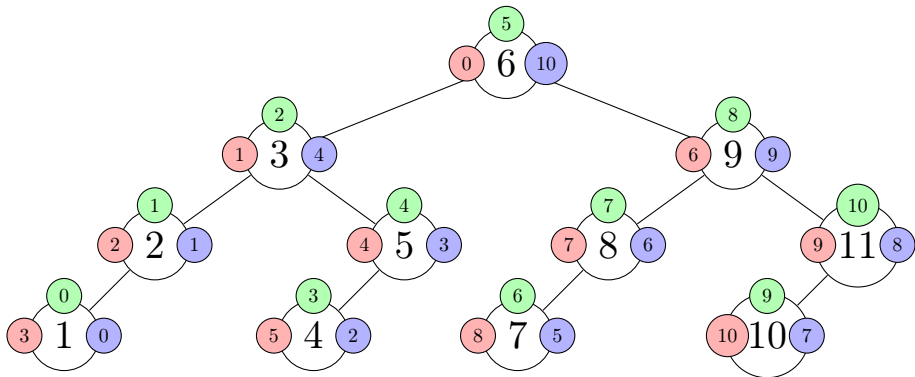


Figure 5.2: Traversal sorts: preorder (red), inorder (green), postorder (blue).

Table 5.1: Chunks Pre/Post Traversal Sorts on two resources

Order	Operation 1	Operation 2
<b>T1</b>	Traversal Order Sort	Chunk Ks by Resource Count
In	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11	[1, 2, 3, 4, 5, 6] [7, 8, 9, 10, 11]
Pre	6, 3, 2, 1, 5, 4, 9, 8, 7, 11, 10	[6, 3, 2, 1, 5, 4] [9, 8, 7, 11, 10]
Post	1, 2, 4, 5, 3, 7, 8, 10, 11, 9, 6	[1, 2, 4, 5, 3, 7] [8, 10, 11, 9, 6]
<b>T2</b>	Traversal Order Sort	Chunk Ks by Alg. 4
In	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11	[1, 3, 5, 7, 9, 11] [2, 4, 6, 8, 10]
Pre	6, 3, 2, 1, 5, 4, 9, 8, 7, 11, 10	[3, 1, 5, 9, 7, 11] [6, 2, 4, 8, 10]
Post	1, 2, 4, 5, 3, 7, 8, 10, 11, 9, 6	[1, 5, 3, 7, 11, 9] [2, 4, 8, 10, 6]
<b>T3</b>	Chunk Ks by Resource Count	Traversal Order Sort
In	[1, 2, 3, 4, 5, 6] [7, 8, 9, 10, 11]	[1, 2, 3, 4, 5, 6] [7, 8, 9, 10, 11]
Pre	[1, 2, 3, 4, 5, 6] [7, 8, 9, 10, 11]	[4, 2, 1, 3, 6, 5] [9, 8, 7, 11, 10]
Post	[1, 2, 3, 4, 5, 6] [7, 8, 9, 10, 11]	[1, 3, 2, 5, 6, 4] [7, 8, 10, 11, 9]
<b>T4</b>	Chunk Ks by Alg. 4	Traversal Order Sort
In	[1, 3, 5, 7, 9, 11] [2, 4, 6, 8, 10]	[1, 3, 5, 7, 9, 11] [2, 4, 6, 8, 10]
Pre	[1, 3, 5, 7, 9, 11] [2, 4, 6, 8, 10]	[7, 3, 1, 5, 11, 9] [6, 4, 2, 10, 8]
Post	[1, 3, 5, 7, 9, 11] [2, 4, 6, 8, 10]	[1, 5, 3, 9, 11, 7] [2, 4, 9, 10, 6]

---

**Algorithm 4** Chunk  $k$  values by Skip Mod Resource Count

---

**Require:**  $\mathcal{K}$  (list of  $k$ ),  $num\_resources$  (resource count to split  $k$ )

**Ensure:** List of chunks where each chunk contains integers assigned to a resource

```
1: function CHUNKKS( $K, num\_resources$ )
2:    $K\_chunks \leftarrow []$ 
3:   for  $i = 0$  to  $num\_resources - 1$  do
4:      $K\_chunks[i] \leftarrow$  empty list
5:   end for
6:   for  $k = 0$  to  $Ks - 1$  do
7:      $resource\_id \leftarrow k \bmod num\_resources$ 
8:     Append  $k$  to  $K\_chunks[resource\_id]$ 
9:   end for
10:  return  $K\_chunks$ 
11: end function
```

---

### 5.1.1 Binary Bleed Multi-threaded, Multi-rank

A parallel implementation of Binary Bleed can be obtained by extending the serial implementation depicted in Algorithm 3 with the following changes: Make  $k_{min}$ ,  $k_{max}$  and the list of visited  $k$   $rank_{seen}$  global using a distributed cache such as redis, and  $i_{left}$  and  $i_{right}$  are now computed by each MPI rank or thread as  $i_{left} = k_{min} + idx \times (k_{max} - k_{min}) / size$  and  $i_{right} = \min(i_{left} + size, k_{max})$  where  $idx$  is the thread index or MPI rank, and  $size$  is the total number of threads or MPI ranks. The resulting parallel implementation will work well, but will not be effective when the list of  $k$  is sparse.

A more robust algorithm can be obtained by replacing recursions in Algorithm 3 by a  $k$ -sort, illustrated in Figure 5.2, where  $k$  values are sorted using in-order, pre-order, or post-order binary tree traversal, indicated by the color on three sides of each rank's  $k$  value.

Further,  $\mathcal{K}$  must be chunked into available resources using Algorithm 4. By appropriately sorting and chunking the  $k$  values, the Binary Bleed algorithm can efficiently operate in a multi-threaded and multi-rank environment, enhancing its scalability and performance.

Logistics of Chunking  $\mathcal{K}$  and Traversal Sort : To illustrate the optimal function sequence, assume  $k = [1 - 10]$ , with two operating resources. The four splits of the data are:

1. Sort, then data chunk by resources (Table 5.1's **T1**)
2. Sort, then data chunk by Algorithm 4 ( Table 5.1's **T2**)
3. Data chunk by resources, then sort (Table 5.1's **T3**)
4. Data chunk by Algorithm 4, then sort (Table 5.1's **T4**)

Table 5.1 distributes  $k$  on two resources, showing that in-order traversal monotonically increases, leading to inadequate ordering for early termination of  $k$ . Truncation removes unvisited smaller values, which is impossible for in-order traversal. Furthermore, **T1** demonstrates the insufficient distribution of  $k$  values across resources. For example, the second resource may find an optimal result in truncation for all  $k$  in the first resource, leading to one resource being idle while the other operates. The remaining splits present **T3** as the least optimal, as the values are similarly partitioned to **T1**, resulting in higher optimal resources. **T2** and **T4** are distinguished by order, so operation logistics are a lower priority than the use of Algorithm 4 as a load-balanced partition. Table 5.1 shows that traversal selections are ambiguous. However, Section 8.1 indicates pre-order

traversals visit fewer  $k$  values. Algorithm 5 coordinates with Algorithm 4 to distribute and process  $k$  values across multiple resources efficiently.

---

**Algorithm 5** Multiple Ranks and Threads Binary Bleed

---

**Require:**  $Ks$  (sorted  $k$  values),  $N$  (ranks in network),  $t$  (threads per rank)

```

1: Initialize all ranks with unique identifiers and communication between ranks
2: procedure INITIALIZERANKKS( $Ks, N, t, data$ )
3:    $K_{\text{chunks}} \leftarrow \text{CHUNKKS}(Ks, N)$ 
4:   for each rank  $n$  in the network do
5:     Initialize  $Ks_{\text{bst}} \leftarrow []$ 
6:     TRAVERSALSORT( $K_{\text{chunks}}[n], \text{length}(Ks), \&Ks_{\text{bst}}$ )
7:     STARTTHREADS( $data, Ks_{\text{bst}}, t$ )
8:   end for
9: end procedure
10: procedure STARTTHREADS( $data, Ks, num_{\text{threads}}$ )
11:    $Ks\_size \leftarrow |Ks|, mutex \leftarrow \text{CreateMutex}()$ 
12:   for  $i \leftarrow 0$  to  $Ks\_size$  do
13:     start thread BINARYBLEEDMULTI(
14:        $Ks_{\text{bst}}[i \% num_{\text{threads}}], data, \&k_{\text{optimal}}, mutex$  )
15:   end for
16:   wait for all threads to finish, then process  $k_{\text{optimal}}$ 
17: end procedure
18: procedure BROADCASTK( $k_{\text{optimal}}, sender$ )
19:   for each rank  $n$  in the network do
20:     if  $n \neq sender$  then Send  $k_{\text{optimal}}$  to rank  $n$ 

```

```

20:     end if
21: end for
22: end procedure
23: procedure RECEIVEKCHECK( $k_{\text{receiver}}$ ,  $receiver$ )
24:   if message from network then Receive  $k_{\text{sender}}$ ,  $sender$ 
25:   else return  $k_{\text{receiver}}$ 
26:   end if
27:   if  $k_{\text{receiver}} < k_{\text{sender}}$  then  $k_{\text{receiver}} \leftarrow k_{\text{sender}}$ 
28:   else Send  $k_{\text{receiver}}$  to rank  $sender$ 
29:   end if
30: end procedure

```

---

First, in Algorithm 5, the  $k$  list, number of ranks, and threads per rank are needed. Ranks are initialized with IDs and communication setup (lines 1-2). Algorithm 4 chunks the  $k$  (line 3). Each rank initializes a binary search tree (BST) (lines 4-5) of sorted  $k$  and starts threads on the data (lines 7-9). The thread initializes variables for  $|\mathcal{K}|$  and a mutex (lines 10-11). It iterates over the threads to start Binary Bleed on each  $k$  value, waits for completion, and processes the optimal value (lines 12-16). The broadcast function iterates the ranks, sending the optimal  $k$  to others (lines 17-22). The function for receiving communication is defined (line 23). On receiving a message, the rank checks if the received  $k$  exceeds the current optimal, updating or returning the higher optimal as needed (lines 24-30). The second change needed for parallel Binary Bleed is the communication of pruned  $k$  values to other resources. Algorithm 5, in conjunction with Algorithm 6, contains the set of calls to complete the communication of pruning in parallel and distributed Binary Bleed.

---

**Algorithm 6** Binary Bleed  $k$  Search, Multi-rank & Thread

---

**Require:**  $k$  (value to compute),  $data$  (dataset),  $model$  (calculator),  $scorer$  (quantifies output),

$T_{\text{select\_k}}$  (minimum  $k$  score),  $\text{rank\_id}$  (id of rank)

**Ensure:** Optimal  $k$  value and its prediction score

```
1:  $k_{\text{optimal}} \leftarrow \text{null}$ 
2: function BINARYBLEEDMULTI(  $k, k_{\text{optimal}}, \text{data}, \text{mutex}, T_{\text{select\_k}}$  )
3:   Initialize  $\text{model}, \text{scorer}, \text{report} \leftarrow 0$ 
4:   Lock( $\text{mutex}$ ) and  $k_{\text{received}} \leftarrow k_{\text{optimal}}$  then Unlock( $\text{mutex}$ )
5:    $k_{\text{received}} \leftarrow \text{RECEIVEKCHECK}( k_{\text{receiver}}, \text{rank\_id} )$ 
6:   Lock( $\text{mutex}$ )
7:   if  $k_{\text{received}} \neq k_{\text{optimal}}$  then
8:     if  $\neg k_{\text{optimal}}$  or  $k_{\text{received}} > k_{\text{optimal}}$  then  $k_{\text{optimal}} \leftarrow k_{\text{received}}$ 
9:     else  $\text{report} \leftarrow 1$ 
10:    end if
11:  end if
12:  Unlock( $\text{mutex}$ )
13:  if  $\text{report}$  then BROADCASTK( $k_{\text{optimal}}, \text{rank\_id}$ ) and  $\text{report} \leftarrow 0$ 
14:  else
15:    if  $k_{\text{optimal}} > k$  then return
16:    end if
17:  end if
18:   $\text{score} \leftarrow \text{scorer}(\text{model}(\text{data}, k))$ 
19:  if  $\text{score} \geq k_{\text{select\_threshold}}$  then Lock( $\text{mutex}$ )
20:    if  $\neg k_{\text{optimal}}$  or  $k > k_{\text{optimal}}$  then  $k_{\text{optimal}} \leftarrow k$  and  $\text{report} \leftarrow 1$ 
21:    end if
```

```

22:     Unlock(mutex)
23:     if report then BROADCASTK( $k_{\text{optimal}}$ , rank_id)
24:     end if
25: end if
26: end function

```

---

Algorithm 6 requires the  $k$  range, data, model, scorer, selection threshold, and rank ID. Lines (1-3) initialize the optimal  $k$ , define the Binary Bleed multi-function, and initialize the model, scorer, and report flag. Line (4) locks the mutex, sets the received  $k$  as optimal, and unlocks the mutex. Line (5) checks network messages and sets the received  $k$ . Lines (6-12) lock the mutex, compare optimal and received  $k$ , update the optimal if different, and possibly set the report flag before unlocking the mutex. Lines (13-17) check if a report is needed for other resources and reset the report flag if necessary. Line (18) operates the model, data, and scorer on  $k$ . Lines (19-26) lock the mutex if the score exceeds the threshold, update the optimal  $k$  if larger, and broadcast if the optimal was updated. In multi-threading, the rank propagates the optimal  $k$  from the finding thread to others, pruning smaller  $k$  values. In multi-rank, any rank propagates the optimal  $k$  to other ranks. In both cases, threads report to ranks, which then report to other ranks and threads. In an HPC system, resources are processes or compute ranks. In multi-threading, the optimal  $k$  a thread finds is propagated by the controlling rank to all threads, which prune smaller  $k$  values. In multi-rank, any rank finding the optimal  $k$  shares it with all ranks. Threads report to ranks, then communicate the optimal  $k$  to other ranks and threads.

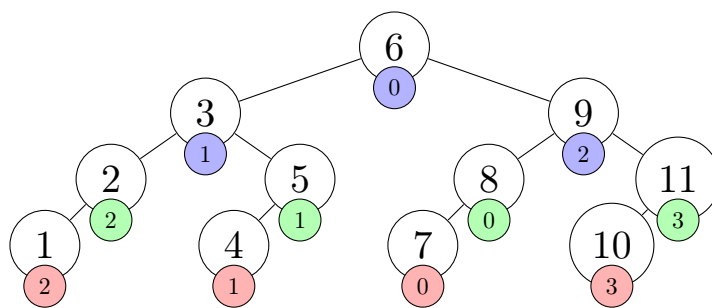


Figure 5.3: Vanilla: Part 1

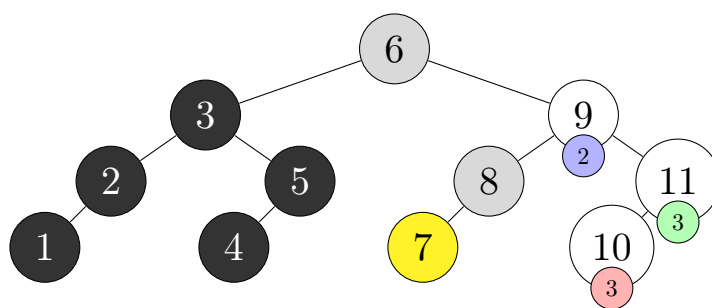


Figure 5.4: Vanilla: Part 2

Operation Dynamics: Figure 5.3 Pre-order sorts  $k = \{1, 2, \dots, 11\}$  on three resources after Algorithm 4 (T4 5.1).  $\forall k$  are the possible optimal. Resource allocation (r,g,b) is by visit order. In part 2, Figure 5.4, the score of  $k = 7$  is greater than the threshold, so it is optimal, and the scores of  $k = \{6, 8\}$  are less than the threshold.  $k = \{1, 2, \dots, 5\}$  are pruned for being less than the optimal. The upper  $k$  range,  $k = \{9, 10, 11\}$  continues. Figure 5.5 schedules  $k$  four resources chunk values, then pre-order sorted. The graph shows the optimal selection threshold as a dashed gold line. Yellow vertical bars indicate truncation is reported in other resources, where a  $k$  score passed the threshold. The gold threshold is crossed at four values:  $k = \{7, 8, 10, 24\}$ . When the first  $k = 8$  passes the threshold, all  $k < 8$  are pruned. However, the implementation shown does not prune  $k$

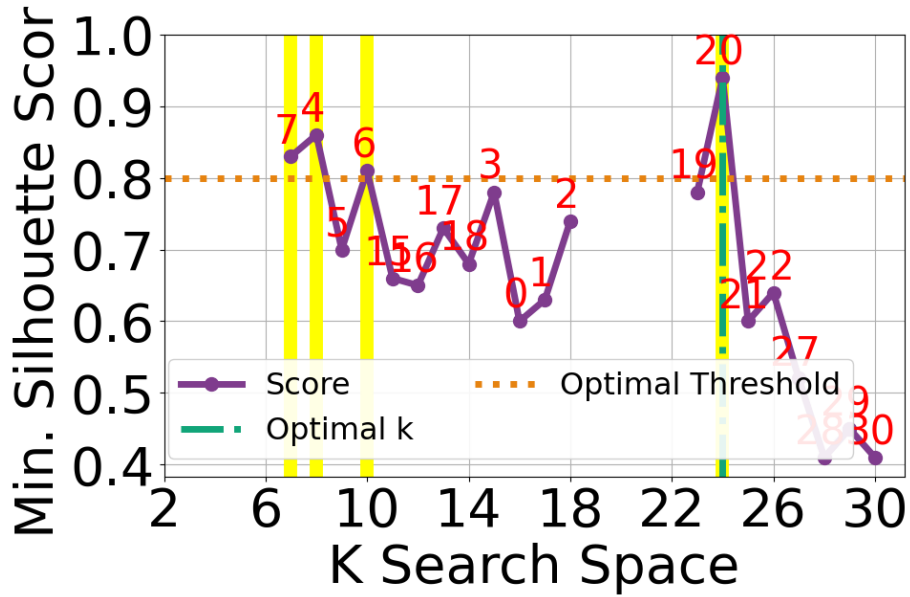


Figure 5.5: Binary Bleed Vanilla.

values after the model begins execution; therefore,  $k = 7$  is complete. The  $k$  values  $[7, 10]$  have no remaining  $k$  values to prune. Due to pre-order sorting,  $K = 24$  is run before  $k = \{18, 19, \dots, 22\}$ . Since  $k = 24$  is above the selection threshold, those lower priority  $k$  values are pruned. The upper  $k$  range continues without crossing the selection threshold, so  $K = 24$  remains optimal.

### 5.1.2 Binary Bleed Early Stop

Specific domains allow for an additional heuristic. Rather than pruning the lower  $k$  values, setting a bound on the other extreme of the data will allow for truncation of the upper values. This heuristic is based on domain knowledge, where if any scores cross the stop threshold, they will never rise to cross the selection threshold, and subsequent  $k$  values can be ignored. Mathematically, it can be represented as:

$$k_{\text{optimal}} = \max \{k \in \{1, 2, \dots, K\} : \forall i \leq k, S(k_i) > U\}$$

where  $S(k_i)$  is the score of the  $i^{\text{th}}$   $k$ , and  $U$  is the stop bound.

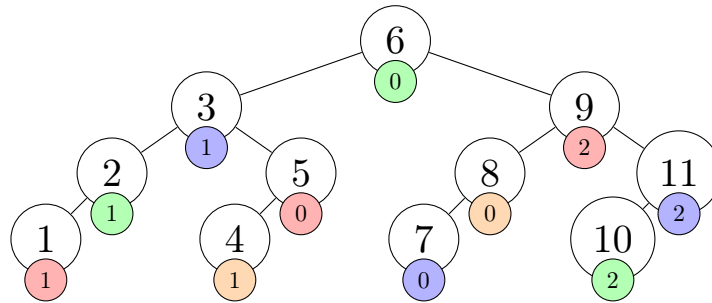


Figure 5.6: Early Stop: Part 1

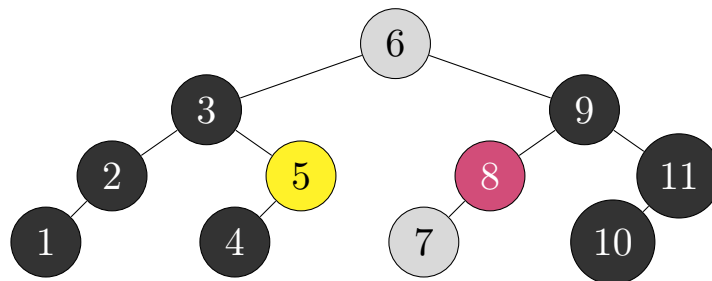


Figure 5.7: Early Stop: Part 2

Figure 5.6 Pre-order sorts  $k = [1 - 11]$  after Algorithm 4 (**T4** 5.1), on four resources. In part 2, Figure 5.7,  $k = 5$  exceeds the selection threshold, so the optimal is set and  $k = [1 - 4]$  (dark nodes) are pruned.  $k = 8$  crosses the stop threshold, so  $k = [9 - 11]$  are pruned. Therefore, the optimal remains 5.

### 5.1.3 Additional Considerations

In multi-resource computing, an optimal  $k$  may be larger than an executing  $k$ . For long computations, checks can be pushed into the model to terminate such  $k$  early. Consider the scoring

distribution, like silhouettes, which can speed up the process. True optimal values are found faster when scores above the threshold resemble a square wave. Mathematically:

$$S(k_i) = \frac{\text{sgn}(k_0 - k_i) + 1}{2}$$

where  $S(k_i)$  is the evaluation score at the  $i^{\text{th}}$  index of  $k$  values,  $k_0$  is the optimal  $k$ , and  $\text{sgn}$  is the signum function, which evaluates to  $+1$  for  $k_i < k_0$  and  $-1$  for  $k_i \geq k_0$ . In the worst case, a Laplacian score distribution will peak at the optimal  $k$  while other  $k$  scores will be below the score selection threshold. Lower  $k$  values may be pruned if the peak is visited before lower values. Otherwise, all values will be visited in the order of the sort. Despite the score distribution, Binary Bleed will not visit more  $k$  values than a linear search.

#### 5.1.4 HNMFk implementation details

Building on the theoretical foundation in Section 2.5, we deploy HNMFk as the workhorse of the topic–concept decomposition layer. The input is the TF-IDF-weighted document–term matrix  $\mathbf{X} \in \mathbb{R}_+^{n \times m}$  whose vocabulary has been pre-filtered for corpus-specific stop-words and rare terms (occurrence  $< 5$ ). An adaptive candidate range  $K^{(0)} = \{2, 3, \dots, 32\}$  is selected via the rank–heuristic  $K^{(0)} = \{2, \dots, \lceil \sqrt{\min(n, m)} \rceil\}$  and refined on-the-fly by the NEXTRANGE policy in Algorithm 2. Each NMFk fit uses  $\beta$ -divergence ( $\beta = 1$ ) minimisation with 200 multiplicative-update iterations, followed by 10 HALS2 refinement sweeps to accelerate convergence [46]. The rank-selection criterion is the minimum description length (MDL) of the residual spectrum, evaluated over a held-out validation fold to avoid optimistic bias [204]. All factors are re-normalised to unit  $\ell_1$  length before the maximum–loading assignment to stabilise clustering against scale drift.

Distributed execution: HNMFk is embedded in an `mpi4py`-based breadth-first scheduler that

maps each active node of the hierarchy to a distinct Slurm task. The checkpointed factor pairs  $(\widetilde{\mathbf{W}}, \widetilde{\mathbf{H}})$  and index sets  $\mathcal{I}_c$ , so that failed tasks can be restarted without re-factoring ancestors. For corpora with  $> 10^6$  documents, we slice  $\mathbf{X}$  row-wise into vocabulary shards of at most  $10^5$  terms, factorize each shard independently, and merge the shard-specific hierarchies through HEAL matching [21].

Post-processing and export: Every non-leaf node is enriched with (i) the top-20 loading terms in  $\widetilde{\mathbf{W}}_c$ , (ii) a centroid embedding obtained by averaging sentence-BERT vectors of member documents, and (iii) a list of citation identifiers propagated from the raw corpus. These artefacts seed the graph-reasoning layer (Section 7.2) and the dense vector store. The complete hierarchy is serialised in a compact JSON-LD dialect (6–8× smaller than plain CSV) and pushed to the RabbitMQ message bus, enabling asynchronous downstream consumers without blocking the factorisation frontier.

## 5.2 Automatic Topic Labelling with Embedding–Aligned LLM

### Prompts

Hierarchical factorisation yields a set of coherent but unnamed clusters. To replace ad-hoc manual naming, we introduce a two-stage label-selection pipeline that unites distributional semantics with LLM generation. The key idea is to sample candidate phrases from an LLM conditioned on the cluster’s most salient terms and then rank those phrases by vector proximity to the cluster centroid in the same embedding space used for downstream retrieval.

**Notation.** Let  $\mathcal{D} = \{d_1, \dots, d_N\}$  be a corpus segmented into  $K$  latent topics by HNMF $k$ , and let  $f : \mathcal{D} \cup \mathcal{L} \rightarrow \mathbb{R}^d$  denote a sentence-level encoder (SciNCL or SPECTER) that maps both document texts and candidate label phrases  $\ell \in \mathcal{L}$  into the same  $d$ -dimensional space.

### Step 1: Cluster Centroid Estimation

For each topic  $c$ , compute the centroid

$$\mu_c = \frac{1}{|\mathcal{D}_c|} \sum_{d \in \mathcal{D}_c} f(d),$$

or alternatively a medoid when `center_metric` is set to "closest". The index of the document closest to  $\mu_c$  is cached for explanatory displays.

### Step 2: Candidate Generation via LLM Prompting

Given the  $m$  highest-probability tokens  $\langle w_1, \dots, w_m \rangle$  from the factor-loading column  $\mathbf{W}_{\cdot,c}$ , we build a few-shot prompt

`Create a concise label for the following words (order matters):  $w_1, \dots, w_m$ .`

Optional criteria: minimum/maximum word count, must-contain or must-not-contain lists, and auxiliary metadata (e.g., venue, year span) may be embedded in the prompt. The framework supports both local Ollama models (`llama3.2:3b-instruct-fp16`) and remote OpenAI models; here we use the former for reproducibility. We draw  $n_{\text{trial}} = 10$  independent samples at temperature  $\tau = 0.4$  and discard any sample that violates the criteria (Algorithm 1).

---

**Algorithm 7** Sampling valid candidate labels

---

```
1: function GENERATEVALIDLABELS( $LLM, p, \kappa, n$ )  
2:    $\mathcal{L} \leftarrow \emptyset$   
3:   while  $|\mathcal{L}| < n$  do  
4:      $\ell \leftarrow LLM.invoke(p)$   
5:     if VALIDATE( $\ell, \kappa$ ) then  $\mathcal{L} \leftarrow \mathcal{L} \cup \{\ell\}$   
6:     end if  
7:   end while  
8:   return  $\mathcal{L}$   
9: end function
```

---

**Step 3: Embedding-Consistent Selection**

Each candidate  $\ell$  is embedded by  $f$  on CPU or GPU, depending on `embedds_use_gpu`. The cosine (or user-supplied) distance to the centroid is computed, and the *closest* label is selected:

$$\hat{\ell}_c = \arg \min_{\ell \in \mathcal{L}_c} \text{dist}(f(\ell), \boldsymbol{\mu}_c).$$

This embedding-aligned LLM labelling scheme closes the loop between distributional topic discovery and human-readable semantics, providing interpretable, high-fidelity labels without manual curation.

## 5.3 Conclusion

The procedures outlined in this chapter complete the middle layer of the dissertation’s architecture by organizing statistical patterns into a hierarchical topic structure. Binary Bleed determines

a compact, data-driven rank that ensures the decomposition remains tractable at scale. Hierarchical NMFk then constructs a layered factorization that captures domain concepts at varying levels of specificity, while maintaining interpretability for subject-matter experts. These components are further supported by an automated labeling pipeline that assigns concise, readable names to each factor based on embedding similarity. Together, they form a structured representation that enables the two core capabilities defined in the abstract: accurate retrieval, addressed in the following chapter through Tensor-Structured RAG, and evidence-based reasoning, developed later through tensor-based link prediction. This chapter establishes the system’s semantic foundation, transforming unstructured term–document statistics into a stable and interpretable framework for downstream tasks.

## Chapter 6

# Knowledge-Graph & Vector Store

## Retrieval Augmented Generation

The previous chapter identified the latent structure of the corpus; this chapter builds on that foundation by constructing a knowledge representation that supports querying, versioning, and inference. It begins by defining a typed, schema-aware knowledge graph in which edges reflect the topic structures produced by the earlier decomposition methods. After examining the limitations of language model-generated graphs, including issues such as hallucination, semantic drift, and schema violations, the focus shifts to a matrix and tensor factorization approach that grounds all nodes and relations in observed co-occurrence statistics. In parallel, a topic-partitioned vector store is constructed using aligned embeddings that preserve the same hierarchical semantics. Together, these two representations form a consistent foundation for downstream retrieval and reasoning tasks.

## 6.1 Unified Data-Infrastructure for Retrieval-Augmented Generation

A scalable retrieval-augmented generation (RAG) system emerges from coupled layers that may be re-used, with minimal parameter changes, across domains as diverse as biomedical literature, legal corpora, or cyber-security reports.

- 1. Hierarchical topic discovery.** HNMF $k$  extracts a multi-level latent taxonomy whose outputs seed the subsequent layers.
- 2. Knowledge-graph (KG) construction.** A labelled-property graph unifies entities, provenance, and ontological types.
- 3. Vector-store (VS) construction.** Topic-sharded indices provide sub-second dense retrieval.
- 4. Hybrid retrieval orchestration.** An agent loop routes queries between symbolic and sub-symbolic stores.

### 6.1.1 Schema Design for a Versioned Knowledge Graph

The KG is formalised as

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{L}, \mathcal{P}),$$

where  $\mathcal{V}$  is the node set,  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$  the directed edge set,  $\mathcal{L}$  the labelling function, and  $\mathcal{P}$  a key-value map (e.g. `doi`, `version`, `confidence`). Table 6.1 summarises the principal node labels and edge types.

Node labels	
Document	{doi,title,year,version}
Author	{author_id,name}
Affiliation	{aff_id,name,country}
Venue	{venue_id,name}
Topic	{topic_id,level,coherence}
Keyword	{term}
Edge types	
WRITTEN_BY	Document → Author
AFFILIATED_WITH	Author → Affiliation
PUBLISHED_IN	Document → Venue
CITES	Document → Document
MENTIONS	Document → Keyword
IN_TOPIC [w]	Document → Topic (weight $w$ )

Table 6.1: Core KG schema with provenance and monotone **version** properties.

**Entity extraction & resolution.** Named-entity recognition (spaCy) yields raw mentions  $E = \text{NER}(T)$  from the corpus  $T = \{t_1, \dots, t_n\}$ . Ambiguity is resolved by a hybrid distance

$$\text{dist}(e_i, e_j) = \alpha \text{operatorname{SimHash}} + (1 - \alpha) \text{Lev}, \quad 0 \leq \alpha \leq 1,$$

merging pairs below a threshold  $\tau$ . Canonicalised entities are then batch-inserted via UNWIND, each edge recording a *provenance-confidence* pair  $(v, c)$  to guarantee auditability.

### 6.1.2 Data Sources

Our pipeline ingests heterogeneous artefacts:

- **Structured metadata** from SCOPUS and Semantic Scholar APIs, including citations and affiliation hierarchies.
- **Born-digital full texts** (PDF or XML) harvested through publisher APIs compliant with Crossref’s *TDM* licences.
- **Grey literature** (e.g. policy briefs) fetched via focused web-crawls and subsequently filtered by a domain-specific classifier.

All sources are normalised into a canonical `Document` record whose fields feed both KG nodes and VS payloads.

### 6.1.3 Full-Text Acquisition and Segmentation

Full texts are parsed with GROBID; boiler-plate sections (references, appendices) are removed by a rule-based filter. Each document’s token sequence  $\langle w_1, \dots, w_L \rangle$  is split into windows  $S_i$  of  $B = 384$  tokens with an overlap of  $s = 48$  tokens,

$$S_i = \langle w_{iB+1}, \dots, w_{(i+1)B} \rangle, \quad S_{i+1} \leftarrow S_{i+1} \cup \text{tail}(S_i).$$

The pair  $(doc\_id, par\_id)$  serves as a globally unique key shared by the KG and VS, thereby enabling  $O(1)$  cross-store dereferencing.

### 6.1.4 Special Graph Constructions

Beyond the base KG, two derived graphs support downstream tasks:

**Co-authorship graph.** An undirected, weighted projection of the `WRITTEN_BY` relation; edge weights are TF-normalised collaboration counts.

**Citation-topic graph.** A bipartite graph linking `Topic` nodes to cited `Document` nodes, capturing topic-specific influence patterns.

### 6.1.5 Vector Store Construction

Each text segment  $S_i$  is embedded once with a domain-tuned transformer encoder, yielding a single vector representation  $\mathbf{v}_{S_i} \in \mathbb{R}^d$ . After `HNMFk` assigns a dominant topic  $T_k$  to every segment, all vectors belonging to  $T_k$  are written directly into a dedicated Milvus collection that uses an HNSW index. Partitioning by topic keeps each query inside a semantically coherent subset and reduces latency; index parameters are selected empirically to balance recall against throughput.

New vectors are inserted straight into the persistent Milvus layer—no intermediate caching tier is required. Vector-store identifiers are aligned with those in the knowledge graph so that

$$\text{KGVersion}(d, p) = \text{VSVersion}(d, p)$$

for every document  $d$  and paragraph  $p$ , ensuring consistent cross-store look-ups.

### 6.1.6 Knowledge Graph Population

The topics produced by `HNMFk` are materialised as `:Topic` nodes annotated with their hierarchy depth and coherence score. For each topic, the most salient keywords are instantiated as `:Keyword` nodes and connected to the topic by `MENTIONS` edges weighted by keyword importance. A document  $d$  links to its predominant topic via an `IN_TOPIC` edge whenever the corresponding entry  $H_{dk}$  in the loading matrix exceeds a threshold chosen to balance coverage and precision.

All nodes and edges carry **provenance**, **confidence**, and **version** properties, enabling complete audit trails and time-travel queries over historical graph states.

### 6.1.7 Hybrid Retrieval and Agent Orchestration

**Query routing.** An instruction-tuned classifier predicts an execution plan

$$Q(q) = \{\text{VS} : \ell, \text{KG} : m\},$$

where  $\ell$  is the number of vector hops and  $m$  the number of graph traversals, guided by lexical cues and topic priors.

**Agent loop.** Adopting the ReAct paradigm, the agent alternates reasoning  $R_t$  and action  $A_t$ ,

$$(R_t, A_t) = \text{Agent}(q, \text{history}_{<t}),$$

until its self-assessed faithfulness  $s \geq 0.7$ . Available tools are

$$\text{VS-Search}(q, k), \text{VS-ReRank}(q, \mathcal{C}), \text{KG- Traverse}(\psi),$$

enabling dense retrieval, cross-encoder re-ranking, and symbolic Cypher execution. The final answer concatenates ranked passages, structured graph facts, and citations.

### 6.1.8 Contribution

By unifying topic-aware KG edges, topic-sharded VS indices, and an agent-mediated hybrid retrieval loop, we deliver a reproducible blueprint for low-latency, high-accuracy RAG systems. The design supports strict versioning, full provenance, and plug-and-play domain adaptation, thereby lowering the barrier to production-grade knowledge-centred language models.

## 6.2 Conclusion

By integrating a typed knowledge graph, provenance-aware metadata, and a topic-partitioned vector store within a unified, versioned pipeline, this chapter completes the central layer of the dissertation’s architecture. The knowledge graph provides a structured and interpretable representation grounded in co-occurrence statistics, while the vector store enables efficient semantic retrieval through aligned embeddings. Shared identifiers across both components ensure consistency and traceability, allowing for reliable interaction between symbolic and embedding-based views of the corpus. This alignment sets the stage for the next phase of the system, the Tensor-Structured Retrieval-Augmented Generation module, and provides the structured basis for tensor-based link prediction. In summary, this chapter translates abstract statistical patterns into a coherent, extensible knowledge representation, supporting the dissertation’s broader objective of building AI systems that retrieve and reason over domain-specific information with transparency and scale.

## Chapter 7

# AI Reasoning

Having established a consistent alignment between the symbolic graph and the vector store in the previous chapter, this chapter addresses the gaps that remain in the knowledge representation. It introduces a family of tensor factorization models, WNMFk, RNMFk, and BNMFk, that frame missing links as low-rank completion problems. Each model selects its rank automatically, estimates its confidence, and combines complementary perspectives such as Boolean, real-valued, and probabilistic to produce calibrated link predictions across large-scale graphs. The focus remains on achieving high link prediction accuracy and quantifying uncertainty, while more complex forms of graph reasoning are intentionally deferred until the graph has been made denser and more complete through this stage of inference.

### 7.1 Link Prediction

This section discusses the methodologies and techniques underlying the approaches. Sections 7.1.1, 7.1.2, and 7.1.3 present the WNMFk, RNMFk, and BNMFk methods, respectively. The ensem-

Table 7.1: Summary of the notation styles used in the paper.

Notation	Description	Notation	Description
$x$	Scalar	$\mathbf{x}_i$	$i$ th element in the vector
$\mathbf{x}$	Vector	$\mathbf{X}_{ij}$	Entry at row $i$ , column $j$
$\mathbf{X}$	Matrix	$\mathbf{X}_{i:}$	$i$ th row
$\mathfrak{X}$	Tensor	$\mathbf{X}_{:j}$	$j$ th column
$\mathfrak{X}_{::i}$	$i$ th slice (3rd dim)	$\mathfrak{X}_{:::i}$	$i$ th slice (4th dim)
$\mathfrak{X}^{name}$	Superscript identifier	$*$	Dot product
$\odot$	Element-wise product	$\otimes_B$	Boolean matrix multiplication

ble approaches are described with these methods by integrating LMF in Section 7.1.4. In Section 7.1.5, the Boolean perturbation technique is introduced, and in Section 7.1.6, the Boolean clustering scheme employed in the methods when performing decomposition under Boolean settings is examined. Additionally, Section 7.1.7 introduces the adoption of Otsu thresholding and k-means clustering for thresholding latent factors in Boolean decomposition, along with a description of the coordinate descent-based approach. Note that the Boolean operations discussed in Sections 7.1.5, 7.1.6, and 7.1.7 are integrated into BNMFk and are utilized to enable its functionality in Boolean settings. In the experiments, these operations are also applied to WNMFk and RNMFk to evaluate their performance under Boolean conditions. Conversely, the Boolean settings in BNMFk can be turned off to assess the method’s performance under non-Boolean optimization, providing a comprehensive analysis of its adaptability across different data environments. Finally, the integration of the UQ framework into the system is presented in Section 7.1.8. For ease of reference, a summary of the notations used throughout the paper is provided in Table 7.1.

### 7.1.1 Weighted Non-Negative Matrix Factorization with Automatic Model Determination (WNMFk)

While NMF, as summarized in Section 2.2, is an effective method for extracting meaningful latent features, it minimizes the distance between  $\mathbf{X}$  and the approximation  $\mathbf{WH}$  as defined in Equation 2.1. This minimization includes zero entries in  $\mathbf{X}$ , causing the approximation  $\mathbf{WH}$  to be forced toward zero for those entries. However, in link prediction tasks, the network nodes interact with only a small subset of other nodes, resulting in a highly sparse matrix  $\mathbf{X}$ . Standard NMF minimization fails because the zeros in  $\mathbf{X}$ , representing missing values, are precisely the links to predict rather than those to minimize. Instead, the optimization must consider only the known entries in  $\mathbf{X}$ . Estimating the missing values in this way, as recommendations or predictions of missing links, constitutes a matrix completion problem [234].

WNMFk extends the NMF framework by incorporating element-wise weights during factorization. This modification enables WNMFk to account for varying confidence levels in the observed data, providing robustness in scenarios where some matrix elements are less reliable than others. Additionally, WNMFk extends standard WNMF [116] by integrating the NMFk framework for automatic rank determination, addressing the challenge of selecting the optimal number of latent features,  $k$ .

Given a matrix  $\mathbf{X} \in \mathbb{R}_+^{n \times m}$  and an associated weight matrix  $\mathbf{M} \in \{0, 1\}^{n \times m}$ , where  $\mathbf{M}_{ij} > 0$  indicates that the corresponding entry  $\mathbf{X}_{ij}$  is observed or a known link and  $\mathbf{M}_{ij} = 0$  indicates that the entry is unobserved or a missing link, WNMFk seeks to approximate  $\mathbf{X}$  as the product of two non-negative matrices such that  $\mathbf{X} \approx \mathbf{WH}$ , where  $\mathbf{W} \in \mathbb{R}_+^{n \times k}$  and  $\mathbf{H} \in \mathbb{R}_+^{k \times m}$ . The final prediction

matrix is obtained as:

$$\hat{\mathbf{X}} = \mathbf{W}\mathbf{H}. \quad (7.1)$$

The weight matrix  $\mathbf{M}$  prioritizes reconstructing observed entries while ignoring the missing ones. When the mask  $\mathbf{M}$  is binary, with values 0 and 1, instead of continuous weights, WNMFK naturally performs missing link prediction by focusing solely on reconstructing the observed entries ( $\mathbf{M}_{ij} = 1$ ) while ignoring the unobserved entries ( $\mathbf{M}_{ij} = 0$ ). The objective function for WNMFK is defined as:

$$\underset{\mathbf{W}, \mathbf{H} \geq 0}{\text{minimize}} \|\mathbf{M} \odot (\mathbf{X} - \mathbf{W}\mathbf{H})\|_F^2 + \lambda(\|\mathbf{W}\|_F^2 + \|\mathbf{H}\|_F^2), \quad (7.2)$$

where  $\lambda$  is a regularization parameter to prevent overfitting. The optimization problem is solved iteratively using multiplicative update rules for  $\mathbf{W}$  and  $\mathbf{H}$ . At each iteration, the residual matrix  $\mathbf{R}$  is computed as  $\mathbf{R} = \mathbf{X} - \mathbf{W}\mathbf{H}$ . For each latent factor  $k$ , the updates for  $\mathbf{W}$  and  $\mathbf{H}$  are as follows, and  $\mathbf{W}$  and  $\mathbf{H}$  are constrained to remain non-negative:

$$W_{:,k} \leftarrow \frac{(\mathbf{M} \odot \mathbf{R})\mathbf{H}_{k,:}^\top}{\mathbf{M}\mathbf{H}_{k,:}^\top\mathbf{H}_{k,:} + \lambda}, \quad (7.3)$$

$$H_{k,:} \leftarrow \frac{(\mathbf{M} \odot \mathbf{R})^\top W_{:,k}}{\mathbf{M}^\top W_{:,k}W_{:,k} + \lambda}. \quad (7.4)$$

To automatically determine the optimal rank  $k$ , WNMFK leverages the NMFk framework, as described in Section 2.3. More specifically, line 4 in Algorithm 1, where  $\mathcal{W}_{::kq}, \mathcal{H}_{::kq} = \text{NMF}(\mathcal{X}_{::q}, k)$ , is replaced by the factorization procedure as described in this section. WNMFK identifies the rank that balances reconstruction accuracy and stability across perturbations by applying the perturbation-based ensemble approach and clustering stability metrics. Furthermore, with WNMFK, the mask matrix  $\mathbf{M}$  can contain non-Boolean entries, allowing continuous values to be used as weights during approximation. For instance, if specific entries in the observed matrix  $\mathbf{X}$  are less reliable or should

contribute less to the optimization process, their corresponding weights in  $\mathbf{M}$ , such as  $\mathbf{M}_{ij}$ , can be assigned lower values. For example, in a recommendation system, if a user’s rating for a specific item is suspected to be influenced by noise (e.g., a randomly assigned rating rather than a genuine preference), the corresponding entry in  $\mathbf{M}$  could be assigned a lower weight to reduce its influence on the factorization. This flexibility makes WNMfK particularly useful in scenarios with varying confidence levels across data points.

### 7.1.2 Recommender NMF with Automatic Model Determination and Biases (RNMfK)

RNMfK incorporates biases to account for systematic user and item tendencies. Like WNMfK, RNMfK integrates the NMFk framework to automatically determine the optimal rank  $k$ , addressing the challenge of model selection. While WNMfK focuses on reconstructing observed entries in the data matrix using a weight mask, RNMfK introduces additional bias terms to improve performance in scenarios where user- and item-specific effects must be explicitly modeled. The mask  $\mathbf{M}$  contains boolean values (0s or 1s) to signify the missing and known entries. The biases help capture variations such as individual user preferences or global item popularity, which cannot be represented solely by the latent factors  $\mathbf{W}$  and  $\mathbf{H}$ . In this work, the Collaborative NMF algorithm presented in the *Surprise* package [94] is extended and modified, and the code is adopted to leverage vector multiplication in the Python package T-ELF, and integrated with the NMFk framework.

Given a data matrix  $\mathbf{X} \in \mathbb{R}_+^{n \times m}$ , where  $n$  represents users or number of rows and  $m$  represents items or number of columns, RNMfK aims to approximate  $\mathbf{X}$  by factoring it into:

$$\hat{\mathbf{X}}_{i,j} = \mathbf{W}_{i,:} \mathbf{H}_{:,j} + b_{W_i} + b_{H_j} + \mu, \quad (7.5)$$

where:  $\mathbf{W} \in \mathbb{R}_+^{n \times k}$  is the user latent factor matrix,  $\mathbf{H} \in \mathbb{R}_+^{k \times m}$  is the item latent factor matrix,  $b_{W_i}$

and  $b_{H_j}$  are the biases for user  $i$  and item  $j$ , respectively,  $\mu$  is the global average rating, representing the group bias. This formulation extends the standard NMF objective by explicitly modeling biases, which account for systematic user- or item-level variations. The optimization objective in RNMFk is given by:

$$\underset{\mathbf{W}, \mathbf{H} \geq 0, b_W, b_H}{\text{minimize}} \quad \|\mathbf{X}_{i,j} - (\mathbf{W}_{i,:} \mathbf{H}_{:,j} + b_{W_i} + b_{H_j} + \mu)\|_F^2 + \alpha \|\mathbf{W}\|_F^2 + \beta \|\mathbf{H}\|_F^2 + \gamma \|b_W\|_F^2 + \delta \|b_H\|_F^2, \quad (7.6)$$

where  $\alpha$ ,  $\beta$ ,  $\gamma$ , and  $\delta$  are regularization parameters for  $\mathbf{W}$ ,  $\mathbf{H}$ ,  $b_W$ , and  $b_H$ , respectively. The minimization is performed only over the observed entries in  $\mathbf{X}$ , making RNMFk suitable for matrix completion tasks. The parameters  $\mathbf{W}$ ,  $\mathbf{H}$ ,  $b_W$ , and  $b_H$  are estimated iteratively using the following updates:

$$b_W \leftarrow b_W + \eta_W \sum_{j=1}^m (\mathbf{err}_{:,j} - \gamma b_W), \quad (7.7) \quad b_H \leftarrow b_H + \eta_H \sum_{i=1}^n (\mathbf{err}_{i,:} - \delta b_H), \quad (7.8)$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\mathbf{X} \mathbf{H}^\top}{\hat{\mathbf{X}} \mathbf{H}^\top + \alpha \mathbf{W}}, \quad (7.9) \quad \mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^\top \mathbf{X}}{\mathbf{W}^\top \hat{\mathbf{X}} + \beta \mathbf{H}}, \quad (7.10)$$

where  $\hat{\mathbf{X}}_{i,j} = \mathbf{W}_{i,:} \mathbf{H}_{:,j} + b_{W_i} + b_{H_j} + \mu$  is the predicted matrix, and  $\mathbf{err}_{i,j} = \mathbf{X}_{i,j} - \hat{\mathbf{X}}_{i,j}$ . To integrate automatic model selection to RNMFk, the factorization procedure of NMFk is replaced, line 4 in Algorithm 1, where  $\mathbf{W}_{:,kq}, \mathbf{H}_{:,kq} = \text{NMF}(\mathbf{X}_{:,q}, k)$ , with the factorization procedure described in this section.

### 7.1.3 Boolean Non-Negative Matrix Factorization with Automatic Model Determination (BNMFk)

BNMFk extends the NMF framework to Boolean settings, where the data matrix  $\mathbf{X} \in \{0, 1\}^{n \times m}$  consists of binary values and integrates NMFk for automatic rank determination. Unlike WNMFk and RNMFk, which use weighted and biased approaches, BNMFk applies constraints to maintain the Boolean structure in both the data and the latent factor matrices during factorization. Given a binary matrix  $\mathbf{X} \in \{0, 1\}^{n \times m}$ , BNMFk approximates  $\mathbf{X}$  as  $\mathbf{X} \approx \mathbf{W} \otimes_B \mathbf{H}$ , where  $\mathbf{W} \in \{0, 1\}^{n \times k}$  is the Boolean row latent factor matrix, and  $\mathbf{H} \in \{0, 1\}^{k \times m}$  is the Boolean column latent factor matrix. The final prediction matrix was also obtained using the Equation 7.1. BNMFk aims to minimize the reconstruction error while maintaining the binary constraints on  $\mathbf{W}$  and  $\mathbf{H}$ . Unlike traditional NMF, the factorization involves thresholding operations to ensure Boolean values. Boolean-specific clustering and thresholding operations are introduced in Sections 7.1.5, 7.1.6, and 7.1.7. In the experiments, these Boolean settings for BNMFk are turned off and turned on for WNMFk and RNMFk. The optimization problem for BNMFk is defined as:

$$\underset{\mathbf{W}, \mathbf{H} \geq 0}{\text{minimize}} \|\mathbf{X} - (\mathbf{W} \otimes_B \mathbf{H})\|_F^2 + \alpha \|\mathbf{W}\|_F^2 + \beta \|\mathbf{H}\|_F^2, \quad (7.11)$$

where  $\alpha$  and  $\beta$  are regularization parameters that penalize large values in  $\mathbf{W}$  and  $\mathbf{H}$ . The Boolean constraints are enforced through adaptive thresholding during updates. BNMFk employs multiplicative updates with adaptive thresholding to ensure the Boolean structure of the latent factor matrices:

$$\mathbf{W} \leftarrow \text{threshold} \left( \mathbf{W} \odot \frac{\mathbf{X}\mathbf{H}^\top}{\mathbf{W}\mathbf{H}^\top\mathbf{H} + \alpha}, \tau_{\text{low}}, \tau_{\text{high}} \right), \quad (7.12)$$

$$\mathbf{H} \leftarrow \text{threshold} \left( \mathbf{H} \odot \frac{\mathbf{W}^\top \mathbf{X}}{\mathbf{W}^\top \mathbf{W} \mathbf{H} + \beta}, \tau_{\text{low}}, \tau_{\text{high}} \right), \quad (7.13)$$

where the `threshold` function, described in Section 7.1.7, for restricting  $\mathbf{W}$  and  $\mathbf{H}$  to Boolean values. Here,  $\tau_{\text{low}}$  and  $\tau_{\text{high}}$  represent the lower and upper thresholds used in the thresholding operation for binarizing the matrices  $\mathbf{W}$  and  $\mathbf{H}$ . BNMFk integrates the NMFk framework to determine the optimal rank  $k$  automatically. Like WNMfK and RNMFk, the NMF procedure in NMFk is replaced, line 4 in Algorithm 1, to integrate an automatic model determination system.

#### 7.1.4 LMF Extensions for Ensemble BNMFk, RNMFk, and WNMfK

An ensemble approach for BNMFk, RNMFk, and WNMfK is introduced by integrating LMF from Section 2.4 into these methods (WNMFk<sub>lmf</sub>, BNMFk<sub>lmf</sub>, and RNMFk<sub>lmf</sub>). This extension leverages the predicted rank from the automatic model determination process in plain BNMFk, RNMFk, and WNMfK, solves for LMF with the same rank, and combines the reconstructed matrix  $\hat{\mathbf{X}}$  from the original decomposition with the biases learned from LMF. The combined output is then passed through a sigmoid function to produce probabilistic predictions.

The ensemble approach works as follows:

1.  $\mathbf{X}$  is first factorized with one of the models introduced above, BNMFk, RNMFk, or WNMfK, to obtain the predicted rank  $k$  and  $\hat{\mathbf{X}}$  (Equation 7.1 for BNMFk and WNMfK, and Equation 7.5 for RNMFk) from this decomposition.
2. The predicted rank  $k$  from the automatic model determination step is used to initialize LMF, where the same  $\mathbf{X}$  is decomposed at the predicted rank  $k$  using LMF.
3. LMF solves the optimization problem to learn row and column biases ( $b_r$  and  $b_c$  from Equation 2.5).

4. Then the final prediction matrix  $\tilde{\mathbf{X}}_{\text{final}}$  is calculated as:

$$\tilde{\mathbf{X}}_{\text{final}} = \sigma \left( \hat{\mathbf{X}} + b_{\text{row}} + b_{\text{col}} \right). \quad (7.14)$$

This ensemble approach aims to combine the strengths of the original matrix factorization method, BNMFk, RNMFk, or WNMFk, with LMF, effectively combining the structural insights and adding the probabilistic modeling capabilities of LMF.

### 7.1.5 Boolean Perturbations for Boolean Factorization

In the context of Boolean matrix factorization, Boolean perturbations are applied to generate an ensemble of slightly modified versions of the binary input matrix  $\mathbf{X}$ . These perturbations introduce controlled noise by flipping randomly selected entries in  $\mathbf{X}$ , ensuring diversity in the perturbed matrices while maintaining the Boolean structure. The perturbation process can be described as follows:

- **Positive Noise (Additive):** A fraction of entries with value 0 in  $\mathbf{X}$  are randomly flipped to 1.
- **Negative Noise (Subtractive):** A fraction of entries with value 1 in  $\mathbf{X}$  are randomly flipped to 0.

The perturbed matrix  $\mathbf{Y}$  is generated by applying these noise components  $\mathbf{Y} = \text{boolean}(\mathbf{X}, \epsilon)$  where  $\epsilon = [\epsilon_{\text{pos}}, \epsilon_{\text{neg}}]$  represents the proportion of positive and negative noise to be added. For each noise type:

- $\epsilon_{\text{pos}}$ : Proportion of 0s in  $\mathbf{X}$  flipped to 1s.
- $\epsilon_{\text{neg}}$ : Proportion of 1s in  $\mathbf{X}$  flipped to 0s.

Line three on Algorithm 1 of NMFk is replaced, where  $\mathbf{X}_{::q} = \text{Perturb}(\mathbf{X})$ , with Boolean perturbation when applying Boolean factorization.

### 7.1.6 Boolean Clustering for Boolean Factorization

In the context of Boolean factorization, the clustering procedure used in the standard NMFk framework, specifically line 7 of Algorithm 1 where  $\hat{\mathbf{W}}, \hat{\mathcal{H}} = \text{customCluster}(\mathbf{W}^{all}, \mathcal{H}^{all})$ , is replaced with a Boolean clustering approach described in this section. This adjustment ensures that the clustering step respects the Boolean nature of the data, making it more suitable for binary datasets. Boolean clustering operates on the latent factor matrices  $\mathbf{W}$  generated from multiple perturbed instances of the input matrix  $\mathbf{X}$ . Let  $\mathbf{W}_{all} \in \{0, 1\}^{n \times k \times M}$  represent a three-dimensional tensor containing the perturbed  $\mathbf{W}$  matrices, where:  $n$  is the number of rows in  $\mathbf{W}$ ,  $k$  is the number of latent features,  $M$  is the number of perturbations, and  $\mathbf{W}$  and  $\mathbf{H}$  are Boolean following the thresholding techniques that will be introduced in Section 7.1.7. Boolean clustering aims to iteratively align and compute the Boolean centroids for the perturbed  $\mathbf{W}$  matrices using a distance metric tailored for Boolean data, such as Hamming distance [131]. The Boolean clustering algorithm alternates between two main steps: distance calculation and centroid computation, repeated until convergence or a maximum number of iterations is reached.

**1. Distance Calculation:** For each perturbed  $\mathbf{W}$ , the algorithm computes the distance between the current centroids and the columns of  $\mathbf{W}$  using a Boolean distance metric, such as Hamming distance. The columns of each perturbed  $\mathbf{W}$  are then reordered to minimize the total distance to the centroids.

**2. Centroid Computation:** The centroids are updated based on the reordered  $\mathbf{W}$  matrices. Boolean centroids are computed by aggregating the binary values of each column across all perturbations, ensuring that the centroids reflect the majority consensus of the binary data.

Let  $\mathbf{W}_{\text{all}}$  represent the tensor of perturbed  $\mathbf{W}$  matrices and  $\mathbf{C}$  represent the centroids. Then the steps are:

1. **Initialization:** The centroids are initialized using the first perturbed  $\mathbf{W}$  matrix in  $\mathbf{W}_{\text{all}}$ .
2. **Distance Calculation:** For each perturbed  $\mathbf{W}$ , compute the distance between the centroids and the columns of  $\mathbf{W}$  and reorder the columns to minimize the distance.
3. **Centroid Update:** Compute the Boolean centroids by aggregating the reordered columns of  $\mathbf{W}$  across all perturbations.
4. **Convergence Check:** If the column ordering stabilizes across all iterations, terminate the algorithm; otherwise, repeat the process.

Boolean clustering replaces the standard custom clustering step in NMFk, enabling BNMFk (or WNMFk and RNMFk when Boolean settings are used) to effectively analyze Boolean datasets while maintaining consistency with the underlying data structure.

### 7.1.7 Boolean Latent Factor Thresholding

Boolean latent factor thresholding ensures that the latent factor matrices  $\mathbf{W}$  and  $\mathbf{H}$  retain their Boolean structure while minimizing the reconstruction error. This section presents three thresholding techniques—Otsu’s method, k-means clustering, and coordinate descent—each of which determines thresholds to binarize  $\mathbf{W}$  and  $\mathbf{H}$  while preserving a close approximation to the observed matrix  $\mathbf{X}$ .

Binarization converts continuous or multi-valued data into binary values (0s and 1s) based on a predefined threshold.

### Otsu's Method

Otsu's method [172] determines the optimal threshold for binarizing the latent factor matrices  $\mathbf{W}$  and  $\mathbf{H}$  by maximizing the between-class variance in their values. Specifically, for each column  $\mathbf{W}_{:,i}$  or row  $\mathbf{H}_{i,:}$ , Otsu's method computes a threshold  $t^*$  that maximizes the separability between binary classes, ensuring that  $\mathbf{W}, \mathbf{H} \in \{0, 1\}$ . The threshold  $t^*$  is defined as:

$$t^* = \arg \max_t \left[ \pi_0(t) \pi_1(t) (\mu_0(t) - \mu_1(t))^2 \right], \quad (7.15)$$

where  $\pi_0(t)$  and  $\pi_1(t)$  are the probabilities (normalized counts) of values in  $\mathbf{W}_{:,i}$  or  $\mathbf{H}_{i,:}$  below and above the threshold  $t$ , respectively,  $\mu_0(t)$  and  $\mu_1(t)$  are the means of the values below and above the threshold  $t$ , respectively. Otsu's method finds the optimal threshold by exhaustively evaluating all possible threshold values and selecting the one that maximizes the between-class variance  $\sigma_B^2(t)$ , defined below. For a given threshold  $t$ , the method partitions the data into two classes: values below  $t$  and values above  $t$ . It then calculates the probabilities  $\pi_0(t)$  and  $\pi_1(t)$  and the means  $\mu_0(t)$  and  $\mu_1(t)$  of the two classes. The threshold that maximizes the separability of these two classes, as measured by  $\sigma_B^2(t)$ , is chosen as the optimal threshold. This ensures that the binary partitioning captures the most significant difference between the two groups. For each component  $i$  of  $\mathbf{W}$  or  $\mathbf{H}$ :

- Compute the histogram of the values in  $\mathbf{W}_{:,i}$  or  $\mathbf{H}_{i,:}$ .
- Calculate  $\pi_0(t)$ ,  $\pi_1(t)$ ,  $\mu_0(t)$ , and  $\mu_1(t)$  for each potential threshold  $t$ .
- Select  $t^*$  that maximizes the between-class variance:

$$\sigma_B^2(t) = \pi_0(t) \pi_1(t) (\mu_0(t) - \mu_1(t))^2.$$

After determining  $t^*$ , binarization is applied as follows:

$$\mathbf{W}_{ij} = \begin{cases} 1, & \text{if } \mathbf{W}_{ij} \geq t^* \\ 0, & \text{if } \mathbf{W}_{ij} < t^* \end{cases}, \mathbf{H}_{ij} = \begin{cases} 1, & \text{if } \mathbf{H}_{ij} \geq t^* \\ 0, & \text{if } \mathbf{H}_{ij} < t^*. \end{cases} \quad (7.16)$$

This approach ensures that the thresholds for  $\mathbf{W}$  and  $\mathbf{H}$  effectively partition the data into binary groups, maximizing the separability between classes and preserving the Boolean structure of the latent factors.

### K-Means Clustering

K-means clustering [176] thresholds each component of  $\mathbf{W}$  and  $\mathbf{H}$  by clustering their values into two groups corresponding to 0 and 1. For a vector  $\mathbf{z}$  (e.g.,  $\mathbf{W}_{:,i}$  or  $\mathbf{H}_{i,:}$ ), k-means clustering identifies two cluster centers  $c_0$  and  $c_1$ . The threshold is then defined as:

$$t^* = \frac{c_0 + c_1}{2}. \quad (7.17)$$

The binarization is performed by assigning:

$$z_j = \begin{cases} 1, & \text{if } z_j \geq t^* \\ 0, & \text{if } z_j < t^*. \end{cases} \quad (7.18)$$

### Coordinate Descent Thresholding (search)

Coordinate descent thresholding iteratively adjusts the thresholds for  $\mathbf{W}$  and  $\mathbf{H}$  to minimize the reconstruction error  $\mathbf{X} \approx \mathbf{W} \odot \mathbf{H}$ . For each component  $i$ , the reconstruction error is computed as:

$$\text{Error} = \|\mathbf{X} - (\mathbf{W}_{:,i}\mathbf{H}_{i,:})\|_F^2. \quad (7.19)$$

The thresholds  $t_W[i]$  and  $t_H[i]$  for  $\mathbf{W}_{:,i}$  and  $\mathbf{H}_{i,:}$  are optimized iteratively:

- Fix  $\mathbf{H}$  and optimize  $\mathbf{W}_{:,i}$  by selecting  $t_W[i]$  to minimize the reconstruction error.
- Fix  $\mathbf{W}$  and optimize  $\mathbf{H}_{i,:}$  by selecting  $t_H[i]$  to minimize the reconstruction error.

The process repeats until the thresholds converge or the maximum number of iterations is reached.

The final binarization is applied as:

$$\mathbf{W}_{:,i} = \begin{cases} 1, & \text{if } \mathbf{W}_{:,i} \geq t_W[i] \\ 0, & \text{if } \mathbf{W}_{:,i} < t_W[i], \end{cases} \quad (7.20)$$

and similarly for  $\mathbf{H}_{i,:}$ . In the experiments, *search* is used as a term for this thresholding technique.

### 7.1.8 Uncertainty Quantification (UQ)

UQ is a critical component of robust predictive modeling, providing insights into the reliability of model predictions. In the context of NMFk and its variants WNMfK, RNMFk, and BNMFk presented in this paper, UQ evaluates the stability of the reconstructed matrix  $\hat{\mathbf{X}}$  (Equation 7.1 for BNMFk and WNMfK, and Equation 7.5 for RNMFk) across perturbations, using data augmentation to define the confidence, enabling the identification of confident and uncertain predictions. The UQ framework leverages the idea that truly confident predictions are stable under perturbation or data augmentation [246, 12], which hypothesizes that stable predictions across multiple perturbations are more likely to be accurate. During the NMFk process, perturbations of the input matrix  $\mathbf{X}$  generate an ensemble of latent factor matrices  $\mathbf{W}_{\text{all}}$  and  $\mathbf{H}_{\text{all}}$ , corresponding to different realizations of  $\mathbf{W}$  and  $\mathbf{H}$ .

Under non-Boolean settings, perturbations are generated by uniformly sampling from  $\mathbf{X}$  to create modified versions that are a controlled distance away from the original matrix  $\mathbf{X}$ . This is achieved by scaling the entries of  $\mathbf{X}$  with random noise drawn uniformly from the range  $[1 - \epsilon, 1 + \epsilon]$ ,

where  $\epsilon$  controls the magnitude of the perturbation. Then, the perturbed matrix  $\mathbf{Y}$  is defined as:

$$\mathbf{Y} = \mathbf{X} \odot (1 - \epsilon + 2\epsilon \cdot \text{rand}(\text{shape}(\mathbf{X}))), \quad (7.21)$$

where:  $\text{rand}(\text{shape}(\mathbf{X}))$  generates random values uniformly distributed in  $[0, 1]$ , and  $\epsilon$  is the perturbation parameter. In the experiments,  $\epsilon = 0.015$ . Here, the  $\epsilon = 0.015$  hyperparameter is selected as it is shown to give a stable region for  $k$  selection [160]. The perturbation method described in Section 7.1.5 is used for Boolean settings. This approach modifies the Boolean structure of  $\mathbf{X}$  by flipping selected entries (from 0 to 1 or 1 to 0) based on the specified noise proportions  $\epsilon_{\text{pos}}$  and  $\epsilon_{\text{neg}}$  ( $(\epsilon_{\text{pos}}, \epsilon_{\text{neg}}) = (0.015, 0.015)$  in the experiments). For each perturbation  $p$ , the reconstructed matrix  $\hat{\mathbf{X}}^{(p)}$  is computed as:

$$\begin{aligned} \hat{\mathbf{X}}^{(p)} &= \mathbf{W}^{(p)} \mathbf{H}^{(p)}, \\ \hat{\mathbf{X}}_{::p} &= \mathbf{W}_{::p} \mathcal{H}_{::p}, \end{aligned} \quad (7.22)$$

where  $\mathbf{W}^{(p)} \in \mathbb{R}_+^{n \times k}$  and  $\mathbf{H}^{(p)} \in \mathbb{R}_+^{k \times m}$  are the latent factors for perturbation  $p$ . For  $P$  perturbations, there are  $\mathbf{W}^{all} = [\mathbf{W}_{::1}, \dots, \mathbf{W}_{::p}, \dots, \mathbf{W}_{::P}]$  and  $\mathcal{H}^{all} = [\mathcal{H}_{::1}, \dots, \mathcal{H}_{::p}, \dots, \mathcal{H}_{::P}]$ , where  $\mathbf{W}^{all}$  and  $\mathcal{H}^{all}$  are three dimensional tensors of size  $n \times k \times P$  and  $k \times m \times P$ . Across  $P$  perturbations, the ensemble of reconstructed matrices  $\{\hat{\mathbf{X}}^{(p)}\}_{p=1}^P$  reflects the variability in the model's predictions, where  $\hat{\mathbf{X}}^{all} = [\hat{\mathbf{X}}_{::1}, \dots, \hat{\mathbf{X}}_{::p}, \dots, \hat{\mathbf{X}}_{::P}]$ , and where  $\hat{\mathbf{X}}^{all}$  is a three dimensional tensor size of  $n \times m \times P$ . To quantify uncertainty, the standard deviation of the reconstructed values for each entry  $(i, j)$  in  $\hat{\mathbf{X}}$  is computed:

$$\mathbf{U}_{ij} = \sqrt{\frac{1}{P} \sum_{p=1}^P \left( \hat{\mathbf{X}}_{ij}^{(p)} - \bar{\hat{\mathbf{X}}}_{ij} \right)^2}, \quad (7.23)$$

where:  $\hat{\mathbf{X}}_{ij}^{(p)}$  is the  $(i, j)$  entry of  $\hat{\mathbf{X}}^{(p)}$ ,  $\bar{\hat{\mathbf{X}}}_{ij} = \frac{1}{P} \sum_{p=1}^P \hat{\mathbf{X}}_{ij}^{(p)}$  is the mean of the reconstructed values at  $(i, j)$  across all  $P$  perturbations. The resulting uncertainty matrix  $\mathbf{U}$  captures the variability of

predictions at each entry of  $\hat{\mathbf{X}}$ . Low uncertainty values or standard deviation indicate consistent predictions across perturbations, suggesting high confidence in the model’s output at those entries. Conversely, high uncertainty values signal variability and reduced confidence in predictions. Based on the hypothesis that truly confident predictions are stable under error [246], predictions with low uncertainty are more likely to be accurate and reliable.

### 7.1.9 Evaluation

Given  $y$  total samples, we define the training set size as a proportion  $train\_size \in [0.1, 0.2, \dots, 0.9]$ . The number of samples in the training set is  $y_{train} = y \times train\_size$ . The test set consists of the remaining samples, with size  $y_{test} = y \times (1 - train\_size)$ . For missing link prediction, we define the observation matrix  $\mathbf{X}$ , where known links are represented by nonzero values ( $\mathbf{X}_{ij} \neq 0$ ), and known negative links (i.e., known absence of a connection) are represented by  $\mathbf{X}_{ij} = 0$ . The locations of known links are determined by the mask matrix  $\mathbf{M}$ , where  $\mathbf{M}_{ij} = 1$  indicates an observed link in the training set, and  $\mathbf{M}_{ij} = 0$  represents a missing link. To construct the training and test sets, we define the index sets:

1. Let  $\mathcal{I}_{pos}$  be the set of indices where  $\mathbf{X}_{ij} \neq 0$  (known positive links).
2. Let  $\mathcal{I}_{neg}$  be the set of indices where  $\mathbf{X}_{ij} = 0$  (known negative links).
3. The test set index set,  $\mathcal{I}_{test}$ , is chosen such that the number of indices satisfies  $|\mathcal{I}_{test}| = y_{test}$ .

The test set is constructed by stratified sampling from both known positive and known negative links:

1. **Positive known links:** A subset of  $\mathcal{I}_{pos}$  is randomly selected and placed in  $\mathcal{I}_{test}$ .
2. **Negative known links:** A subset of  $\mathcal{I}_{neg}$  is randomly selected and placed in  $\mathcal{I}_{test}$ .

3. **Training set:** The remaining indices belong to the training index set  $\mathcal{I}_{\text{train}} = (\mathcal{I}_{\text{pos}} \cup \mathcal{I}_{\text{neg}}) \setminus \mathcal{I}_{\text{test}}$ , where  $\setminus$  represents the set difference.

The mask matrix  $\mathbf{M}$  is defined such that test set locations are masked out:

$$\mathbf{M}_{ij} = \begin{cases} 0, & \text{if } (i, j) \in \mathcal{I}_{\text{test}} \\ 1, & \text{otherwise.} \end{cases} \quad (7.24)$$

The training matrix is then:

$$\mathbf{X}^{\text{train}} = \mathbf{X} \odot \mathbf{M}, \quad (7.25)$$

where  $\odot$  represents the element-wise (Hadamard) product. The test matrix consists of the missing edges:

$$\mathbf{X}^{\text{test}} = \mathbf{X} \odot (1 - \mathbf{M}). \quad (7.26)$$

A prediction matrix  $\hat{\mathbf{X}}$  is computed using a model trained on  $\mathbf{X}^{\text{train}}$ . The performance is evaluated by comparing the predicted values at test locations:

$$\hat{\mathbf{X}}_{\text{test}} = \hat{\mathbf{X}} \odot (1 - \mathbf{M}), \quad (7.27)$$

where we extract only the entries corresponding to the test set to measure prediction performance.

On the Swimmer dataset, we keep the test-set size fixed at 10% and do not test for data sparsity, as the computational time required for this dataset is significantly longer due to the larger matrix. We set the test set size for the PPI dataset to 20%, the same as in [177]. The missing links are randomly sampled 10 times to ensure robust evaluation. All results are reported with an average over the cross-validations with coverage intervals (CIs).

### 7.1.10 Metrics

Performance metrics include rank  $k$  predictions, Root Mean Squared Error (RMSE), abstained sample fraction, RMSE on non-abstained samples, and Pearson Correlation between UQ entries and reconstruction error. We also report ROC AUC and PR AUC, including UQ-based extensions, where UQ values from  $\mathbf{U}$  at test set points  $\mathcal{I}_{\text{test}}$  serve as weights, simulating confidence-based predictions.

#### Rank $k$ Predictions

We assess the accuracy of the model’s automatic determination of the true rank  $k$ . Correct rank prediction is critical for capturing the underlying structure of the data. The results use a Violin plot [85] to show how the rank predictions are distributed.

#### Root Mean Squared Error (RMSE)

RMSE measures the reconstruction error, quantifying how closely the reconstructed matrix  $\hat{\mathbf{X}}$  approximates the observed matrix  $\mathbf{X}$  at the test set points  $\mathcal{I}_{\text{test}}$ :

$$\text{RMSE} = \sqrt{\frac{1}{|\mathcal{I}_{\text{test}}|} \sum_{(i,j) \in \mathcal{I}_{\text{test}}} (\mathbf{X}_{ij} - \hat{\mathbf{X}}_{ij})^2}. \quad (7.28)$$

A lower RMSE indicates better reconstruction accuracy. We use RMSE to report the performance of predicting the missing links.

#### Fraction of Abstained or Rejected Samples

This metric evaluates the proportion of predictions the model abstains from due to high uncertainty, focusing on confident predictions while disregarding less certain ones. The fraction of abstained

samples is calculated as follows:

$$f_{\text{abstain}} = \frac{|\mathcal{I}_{\text{abstain}}|}{|\mathcal{I}_{\text{test}}|}, \quad (7.29)$$

where  $f_{\text{abstain}}$  is the fraction of abstained predictions,  $|\mathcal{I}_{\text{abstain}}|$  is the number of abstained predictions, and  $|\mathcal{I}_{\text{test}}|$  is the total number of test set samples. The coverage rate can be calculated with  $1 - f_{\text{abstain}}$ , referring to the fraction of non-abstained samples. The threshold for rejecting predictions in missing links is based on the uncertainty values in  $\mathbf{U}$ . For a given uncertainty matrix  $\mathbf{U}$ , the threshold for abstaining (reject-option)  $\tau$  is defined as:

$$\tau = \frac{1}{|\mathcal{I}_{\text{train}}|} \sum_{(i,j) \in \mathcal{I}_{\text{train}}} \mathbf{U}_{ij}, \quad (7.30)$$

where  $\mathcal{I}_{\text{train}}$  is the set of all training-set indices (known links), and  $\mathbf{U}_{ij}$  is the uncertainty value at location  $(i, j)$ . Predictions at test-set indices  $(i, j) \in \mathcal{I}_{\text{test}}$  are abstained if:

$$\mathbf{U}_{ij} > \tau, \quad (7.31)$$

where  $\mathcal{I}_{\text{test}}$  represents the set of test-set indices. This ensures that the model avoids making predictions in locations with higher uncertainty than the average certainty observed in the training set (known links).

### RMSE on Non-Rejected or Non-Abstained Samples

This metric calculates the RMSE only on the predictions that are not abstained:

$$\text{RMSE}_{\text{Non-Abstained}} = \sqrt{\frac{1}{|\mathcal{I}'_{\text{test}}|} \sum_{(i,j) \in \mathcal{I}'_{\text{test}}} (\mathbf{x}_{ij} - \hat{\mathbf{x}}_{ij})^2}, \quad (7.32)$$

where  $\mathcal{I}'_{\text{test}}$  is the subset of  $\mathcal{I}_{\text{test}}$  containing non-abstained predictions. This highlights the accuracy of the most confident predictions.

## ROC, AUC, and PR AUC

Receiver Operating Characteristic (ROC) AUC and Precision-Recall (PR) AUC are standard metrics for evaluating classification performance. ROC AUC measures the ability to distinguish between positive and negative classes, while PR AUC evaluates precision and recall trade-offs:

$$\text{ROC AUC} = \int_0^1 \text{TPR}(FPR) d(FPR), \quad \text{PR AUC} = \int_0^1 \text{Precision}(\text{Recall}) d(\text{Recall}). \quad (7.33)$$

The results include their UQ-based extensions incorporating UQ values  $\mathbf{U}$  as weights, simulating a scenario where confidence is assigned to predictions. The weights are calculated based on the UQ values and normalized to account for overall uncertainty. Specifically, the weight for each prediction is defined as:

$$w_{ij} = \frac{\mathbf{U}_{ij}}{1 + \text{median}(\mathbf{U}_{kl} \text{ for } (k, l) \in \mathcal{I}_{\text{train}})}. \quad (7.34)$$

The normalized weight is given by:

$$w_{ij}^{\text{norm}} = \frac{1}{1 + w_{ij}}. \quad (7.35)$$

Using these normalized weights, the UQ-based ROC AUC and PR AUC metrics are computed as:

$$\text{UQ - ROC AUC} = \frac{\sum_{(i,j) \in \mathcal{I}_{\text{test}}} w_{ij}^{\text{norm}} \cdot \text{TPR}(FPR)}{\sum_{(i,j) \in \mathcal{I}_{\text{test}}} w_{ij}^{\text{norm}}}, \quad (7.36)$$

$$\text{UQ - PR AUC} = \frac{\sum_{(i,j) \in \mathcal{I}_{\text{test}}} w_{ij}^{\text{norm}} \cdot \text{Precision}(\text{Recall})}{\sum_{(i,j) \in \mathcal{I}_{\text{test}}} w_{ij}^{\text{norm}}}. \quad (7.37)$$

We incorporate these weights into the ROC AUC and PR AUC calculations by utilizing the `sample_weight` parameter provided in Scikit-learn [176]. This allows us to account for the uncertainty-based weights during the evaluation, ensuring that higher-confidence predictions contribute more

significantly to the metrics. In contrast, less confident predictions have a reduced impact, reflecting the effect of uncertainty on classification performance.

## 7.2 Graph reasoning over latent knowledge graphs

While the vector store supports rapid semantic matching, many queries require multi-hop logical inference that is opaque to pure embedding similarity. We therefore integrate a graph reasoning module that operates on the heterogeneous knowledge graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$  produced by the extraction pipeline. Each document cluster from HNMFk contributes a topic node, connected via edges to its constituent Document nodes and via `has_keyword` edges to the topics recorded above.

**Symbolic path enumeration.** Given an entity-centric query  $q$ , we identify a start set  $S \subset \mathcal{V}$  using deterministic matchers (regular expressions for statute citations, Neo4j full-text for free text). Paths of length  $\leq L=3$  are enumerated in Cypher with edge-type constraints that privilege explanatory relations (`cites`, `is_case_of`, `has_method`) over associative ones (`co_occur`). Each path  $p = v_0 \xrightarrow{r_1} v_1 \dots v_\ell$  receives an initial score  $\sigma_{\text{sym}}(p) = \prod_{i=1}^{\ell} w(r_i)$  where  $w(r)$  is a learnt edge reliability weight.

**Neuro-symbolic re-ranking.** The candidate subgraph is further re-ranked by an R-GCN encoder [190] that projects every node to a  $d$ -dimensional vector, optimised with a margin-ranking loss against ground-truth relevance triples obtained from expert annotations in the cybersecurity and legal domains. The final path score is a convex combination  $\sigma(p) = \lambda \sigma_{\text{sym}}(p) + (1 - \lambda) \frac{1}{\ell} \sum_{i=0}^{\ell} \cos(\mathbf{h}_q, \mathbf{h}_{v_i})$ , where  $\mathbf{h}_q$  is the query embedding from LegalBERT and  $\lambda = 0.4$  tuned on a validation set. Paths above a threshold  $\tau = 0.15$  supply supporting documents whose passages are forwarded to the RAG composer.

**Incremental materialisation.** To keep latency bounded, the reasoning engine maintains an `openCypher` materialised view cache of the  $10^5$  most frequently traversed patterns, refreshed nightly, and invalidated when the upstream KG ingest reports a structural delta  $>0.5\%$ . This policy yields a  $2\times$  reduction in average query time on the jurisprudence workload (median 210 ms vs. 440 ms) without sacrificing recall.

**Alignment with the vector store.** Before retrieval, the top-ranked subgraph is converted into dense prompts by concatenating relation labels and node names, encoded with the same encoder that built the VS. De-duplication against the  $k$ -NN neighbours ensures that symbolic and dense channels contribute disjoint evidence, realising the hybrid RAG architecture.

## 7.3 Conclusion

The techniques presented in this chapter transform the knowledge graph from a static structure into an evidence-augmented representation, where newly inferred links are accompanied by explicit confidence estimates. These enriched connections improve recall in downstream retrieval tasks, as developed in the next chapter, while providing a carefully expanded foundation for subsequent reasoning. By focusing on accurate link completion without introducing premature complexity, this chapter fulfills the dissertation’s objective of identifying and completing missing links in a way that preserves interpretability and supports reliable inference.

## Part III

# Applications & Results

## Chapter 8

# SMART-SLIC Framework

The previous chapters established the architectural foundation of the system. This chapter turns to its practical significance by demonstrating how the full pipeline operates on real-world data. It traces the workflow from Binary Bleed parameter selection through HNMfK taxonomy construction and the synchronization of the knowledge graph and vector store. Rather than emphasizing numerical benchmarks, the focus is on qualitative outcomes. Binary Bleed reduces the search space, making large-scale factorization computationally routine. HNMfK generates topic hierarchies that are interpretable by subject-matter experts. The combined knowledge graph and vector store remain consistent even when applied to diverse corpora, including scientific articles, legal texts, and cybersecurity reports. Together, these demonstrations show the transition from a conceptual architecture to a functioning system, advancing the dissertation’s central claim that a unified, interpretable pipeline for domain-specific knowledge discovery can replace fragmented and brittle workflows.

## 8.1 Binary Bleed

We gauge the efficacy of unsupervised learning methods in single-node and distributed settings. We evaluate NMF and K-means in the former, then NMF and RESCAL in the latter.

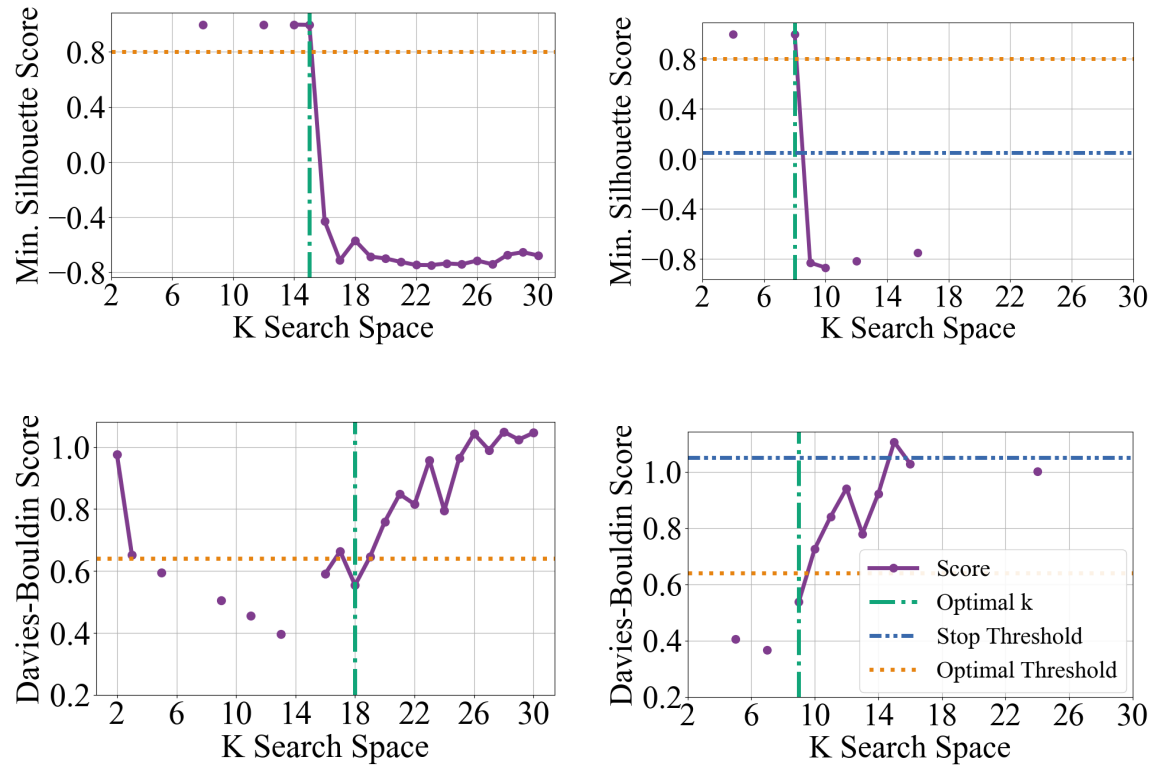


Figure 8.1: NMFk (top row) and K-means (bottom row), Vanilla (Left) and Early stopping (Right).

$$\forall k_{optimal} = k_{true}.$$

### 8.1.1 Single-node Setting

**NMFk:** Data was generated using a synthetic data generator with random Gaussian features for a predetermined  $k$ , where  $k_{true} = \{2, 3, \dots, 30\}$ . The shape of 30 matrices was sized 1000 by 1100,

resulting in 1.1 million non-negative entries. The  $k_{true}$  predetermined the number of clusters in the data, typically showing a drastic drop-off in the scoring metric for subsequent  $k$  values. The evaluation criterion was the silhouette scores of the proposed clusters for the visited  $k$ . For the NMFk Binary Bleed trials, three separate instances were operated over the same data: Standard NMFk, Binary Bleed Vanilla, and Binary Bleed Early Stopping. Each instance was evaluated on all 30 matrices with  $k$  search space  $\mathcal{K} = \{2, 3, \dots, 30\}$ . The results corresponding to NMFk Vanilla and NMFk Early Stopping for  $k = 15$  and  $k = 8$ , respectively, are Figure 8.1's top row. It can be seen that Binary Bleed prunes multiple  $k$  values, whereas the standard method must visit all  $\mathcal{K}$ . In the overview, Figure 8.2, orange and blue lines show the Binary Bleed Vanilla algorithm and the downward trend of  $k$  visits relative to the standard, where Pre-order finds the optimal  $k$  in fewer overall visits. Similarly, Early Stop in pink and green have lower overall visits than Vanilla, with Pre-order benefiting more despite a slightly increasing trend for both Early Stop lines over  $k_{true}$ . Interestingly, the post-order Early Stop is as fast as the Pre-order and visits more  $k$  than the Binary Bleed Vanilla, which can be attributed to the number of compute resources paired with which values are  $k_{true}$  and the ordering of  $\mathcal{K}$ . Overall, the algorithms visit the following percentages of  $\mathcal{K}$ : Pre-order Vanilla: 56%, Post-order Vanilla: 76%, Pre-order Early Stop: 27%, Post-order Early Stop: 44%, while Standard NMFk visits 100% of  $\mathcal{K}$ . This shows pre-ordering  $\mathcal{K}$  with Early Stop executes fastest.

**K-means:** The data was generated by simulating Gaussian-distributed clusters with a standard deviation of .5 and the true  $k$  cluster count. Overlaid random noise introduces variability and ensures robustness.  $k_{true} = \{2, 3, \dots, 30\}$  and  $\mathcal{K} = \{2, 3, \dots, 30\}$ . Given the stochastic scoring, we cluster fifty times for each  $k_{true}$  on every method-ordering pair (Vanilla, K-means Standard, Early Stop) in pre- and Post-order configurations. Davies-Bouldin scoring is used to evaluate the cluster

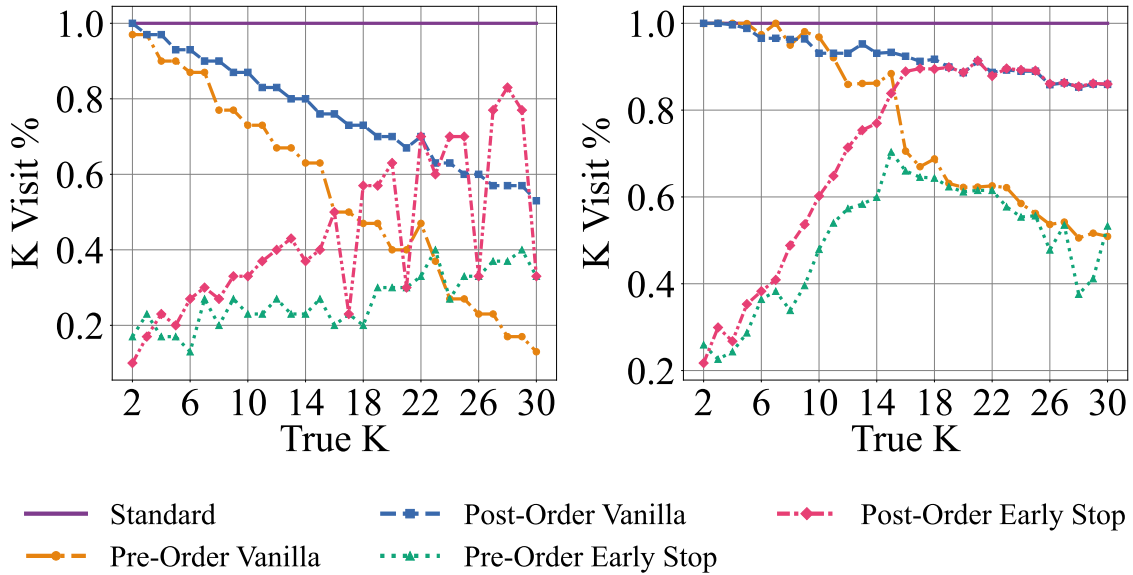


Figure 8.2: Standard NMFk (left) and K-means (right), Vanilla, and Early Stop over Pre-order, and Post-order traversal sorting.

quality and to determine the correct  $k$ . The Root Mean Square Error (RMSE) scores of correct  $k$  were as follows: Post-Order Early Stop: 1.08, Pre-order Early Stop: 2.11, Post-Order Vanilla: 1.08, Pre-order Vanilla: 1.72, and baseline Standard k-means: 1.32. These results indicate the accuracy in identifying  $k_{true}$ , with all Binary Bleed iterations having RMSEs of 0.79 or less, close to the standard RMSE of 1.32.

K-means Vanilla and K-means Early Stopping results for  $k = 18$  and  $k = 9$ , respectively, are shown in Figure 8.1. In the overview, Figure 8.2, Early Stop, pink and green, dominate the speed-up in lower  $k$ , but the dominating factor transitions to Pre-order sorting after  $k = 15$ . Average  $k$  visit percentages are— Pre-order Vanilla: 77%, Post-order Vanilla: 92%, Pre-order Early Stop: 50%, Post-order Early Stop: 71%. As reported, the percent  $k$  visited shows the reduced number of visits

$k$  needed to complete the optimization process.

### 8.1.2 Multi-node Setting

We demonstrate a reduction in  $k$  visits for topic modeling of over 2 million scientific abstracts from arXiv with NMFk from [13]. We used the LANL Chicoma super-computer cluster on the GPU partition and allocated ten nodes, with four NVIDIA A100s per node. NMFk with Binary Bleed Early Stop and standard run on  $\mathcal{K} = \{2, 3, \dots, 100\}$ . Early stop visited 60% of the total compared to Standard NMFk. Both agreed on the  $k_{optimal} = 71$  for the vocabulary size 10,280.

### 8.1.3 Distributed Setting

To showcase the efficacy of the approach on the largest dataset, we utilized the distributed NMF framework, pyDNMFk, results, and the distributed RESCAL framework, pyDRESCAL, as referenced in [23] and [22], respectively. Large datasets need substantial computational resources to factorize. For instance, without Binary Bleed, pyDNMFk required 2 hours with 52,000 cores to estimate  $k$  using standard NMF for a 50TB dataset, averaging 17.14 minutes per  $k$  for  $\mathcal{K} = \{2, 3, \dots, 8\}$ . Similarly, pyDRESCALk required 3 hours with 4,096 cores to factorize 11.5TB of synthetic data using standard RESCAL, averaging 18 minutes per  $k$  for  $\mathcal{K} = \{2, 3, \dots, 11\}$ . Binary Bleed Vanilla and Early Stop results on this distributed data were identical, so only the former’s results were considered, given that the stop thresholds were crossed on the last  $k$ . For both RESCAL and NMFk, the selected  $k$  matched the standard.

**RESCAL:** Binary Bleed was applied to the silhouette and relative error metrics with pre- and Post-order traversal.

For Pre-order traversal, 30% of the total  $k$  values were visited, resulting in an average runtime

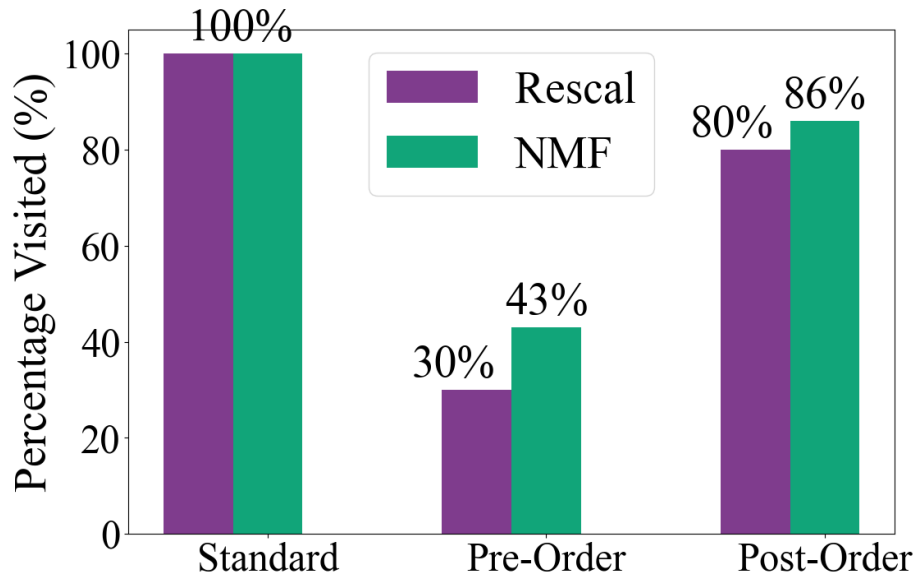


Figure 8.3: Binary Bleed reduction on distributed Rescal (Purple), and distributed NMF (Green).

of 54 minutes, compared to the 180 minutes required for the standard RESCAL. In contrast, Post-Order traversal visited 80% of the total  $k$  values, with an average runtime of 144 minutes, as in Figure 8.3.

**NMF:** Similarly, the results from [23] were analyzed for  $\mathcal{K} = \{2, 3, \dots, 8\}$ . Using Pre-order and post-order traversal, we evaluated the Binary Bleed Vanilla algorithm for the silhouette score. For Pre-Order traversal, 43% of the total  $k$  values were visited, resulting in an average runtime of 51.43 minutes, compared to the 120 minutes required for the standard NMF. Post-order traversal visited 86% of the total  $k$  values, with an average runtime of 102.86 minutes, as shown in Figure 8.3. These results further emphasize the efficiency of the proposed approach in reducing computational time while ensuring effective factorization.

## 8.2 Cyber-Security KG via HNMFk

Results are shown in Figure 8.4 with the word clouds (most prominent words corresponding to each column of  $\mathbf{W}$ ), and with the distribution of categories in each topic and sub-topic in Figure 8.5 (the top ten categories are shown for the given topic). While word clouds highlight the interpretability and quality of the extracted topics, the categories' distribution shows the documents' specificity, and the sub-categories are also examined. The method was hierarchically applied three times, each level selecting one topic to expand further and identify its subtopics. Since the initial  $\mathbf{X}$  is large, chunk the documents into 20 separate matrices and apply semantic joint factorization to determine the number of latent topics automatically. Individually, the 20 matrices revealed between 30 and 80 latent topics. After joint factorization, 24 total super-topics were identified in 2 million+ documents.

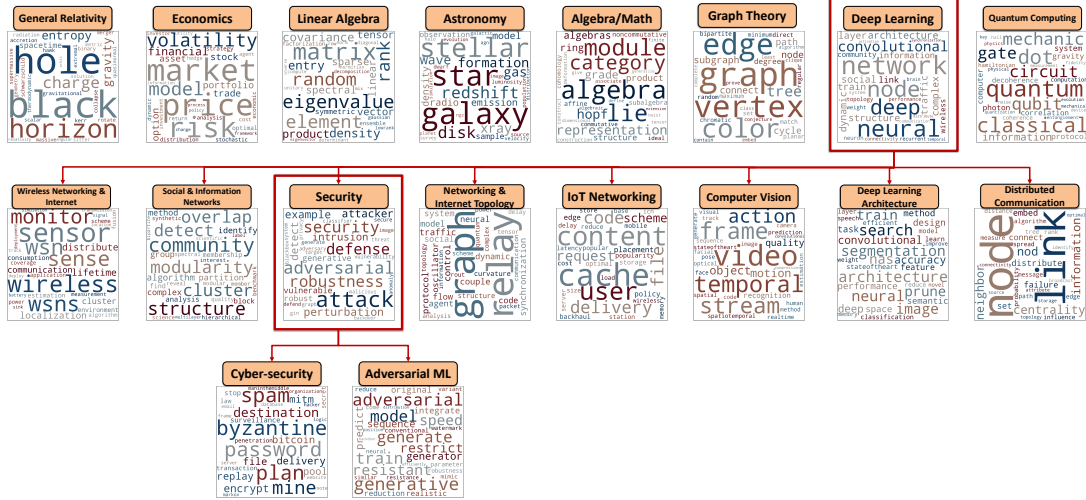


Figure 8.4: Word cloud of selected topics and their interpreted labels. The method is hierarchically applied three times, selecting a topic to extract its subtopics at each level. Selected topics from each level are shown in each row of the figure.

The topic related to Deep Learning was selected. The first row of Figure 8.4 shows the word cloud for this topic, and the first row of Figure 8.5 displays the distribution of categories in this topic. The categories are mainly represented in diverse sub-fields of Computer Science, ranging from Cyber (*cs.CR*) to Computer Vision (*cs.CV*). At the second level, the sub-topic of Deep Learning produced 29 total subtopics, which include Computer Vision and Security. The sub-topic that can be interpreted as Computer Vision based on the tokens in the word cloud is now narrowed down to the *cs.CV*. On the other hand, the Security sub-topic is mainly distributed between the categories *cs.CR* (Cyber), *cs.CV* (Computer Vision), and *cs.LG* (Language); therefore, the Security sub-topic was expanded to the third level, further separating it into three sub-sub-topics, two of which can be interpreted as Cyber-security and Adversarial ML based on the word clouds. Similarly, at the third level, the Cyber-security topic category distribution is narrowed to *cs.CR*. The Adversarial ML topic contains documents from three main categories of *cs.CR*, *cs.CV*, and *cs.LG*, which is reasonable based on this field and how research focuses on adversarial computer vision and language models.

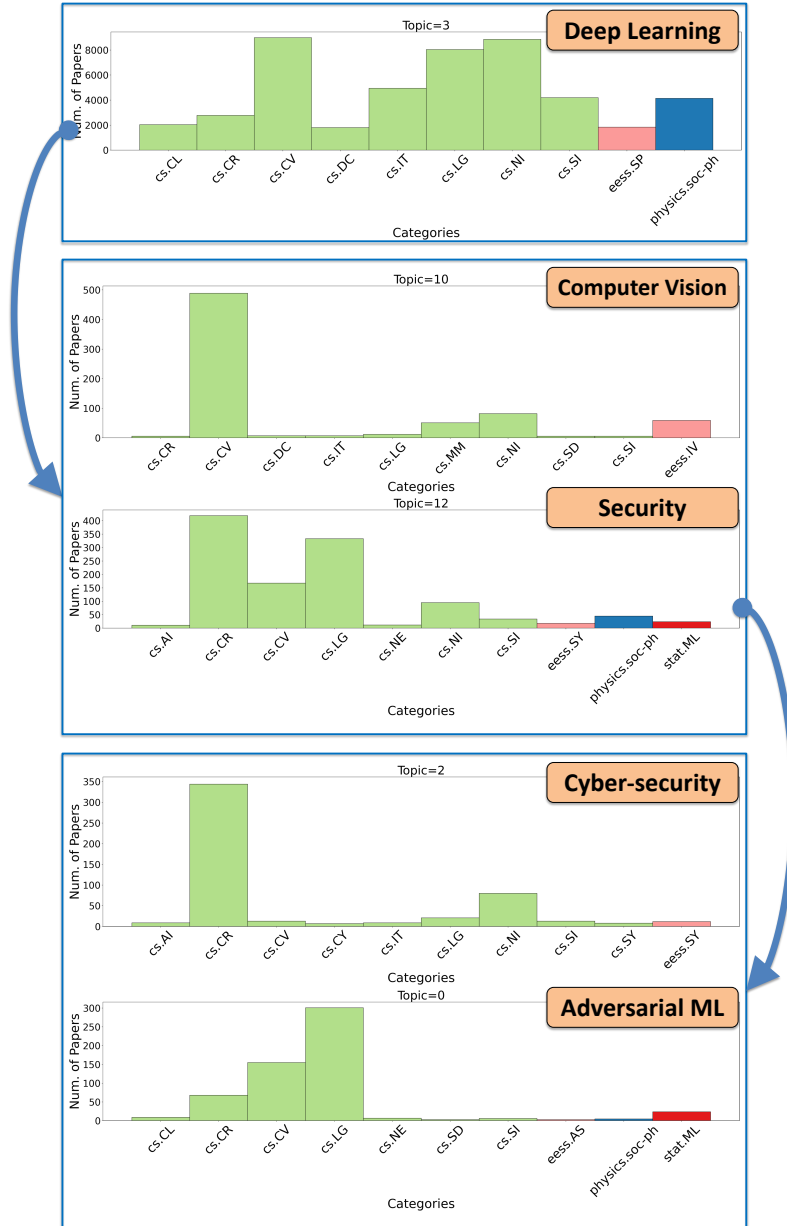


Figure 8.5: Distribution of the top ten document categories, based on the arXiv author assignments. Each topic corresponds to the word clouds from Figure 8.4. Selected topics from each level are shown in each row of the figure.

In the domain-specific leaf decomposition, 1383 vocabulary words were used to discover three latent topics: the counts of the entities in the graph. These three topics contain their own unique, non-overlapping set of documents. Specifically, topic 0 had 597 documents, topic 1 had 114 documents, and topic 2 had 565 documents. These topics comprised 1383 vocabulary words, but the probability of the words occurring in each topic changes according to the documents. The knowledge graph contained NER entities such as 245 organizations, seven events, 64 persons, 3 locations, 679 products, and 22 geopolitical entities. Overall, the graph produced 3758 node entities and 9428 edge relationships. Queries could be asked of the knowledge graph, which must first be translated from natural language into Cypher. For example, 'Which papers mention MNIST?' returns all of the documents with MNIST as an NER, a count of 50 documents. Similarly, 'Which documents in the security decomposition are in the 'cs.SE' (software engineering) arXiv category' finds seven documents, which are further linked to 2 organizations, an event, topics 0 and 2, a product, and seven additional categories.

## 8.3 Law

This section presents the resulting legal texts and their hierarchical decomposition, offering a detailed breakdown of the sections and cases within each document type. The following results illustrate the scope and depth of the collected data, providing a foundation for further exploration of trends and patterns across the legal corpus.

### 8.3.1 Dataset

After collecting and structuring the data, the legal documents were organized into 265 Constitutional provisions, 28,251 Statutory sections, and Case Law, comprising 10,072 Court of Appeals

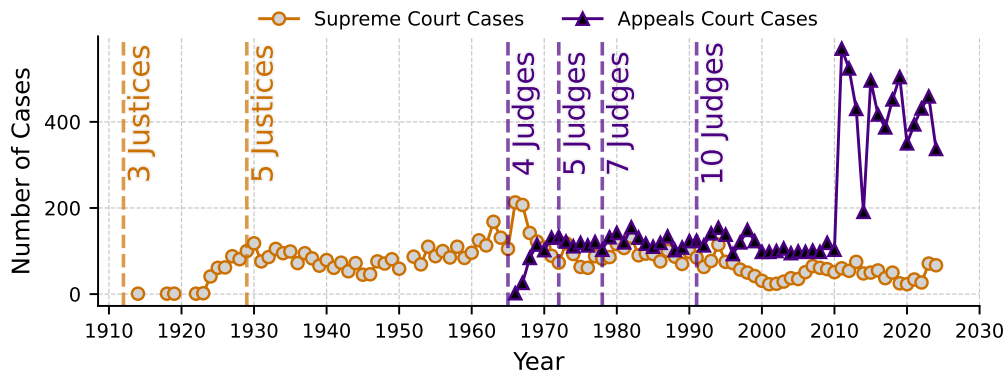


Figure 8.6: New Mexico Supreme/Appeals case counts per year.

cases and 5,727 Supreme Court cases. Figure 8.6 shows the trends in these Supreme Court and Court of Appeals cases over the years, as available from Justia [112], which also includes the expansions and creations of the courts themselves.

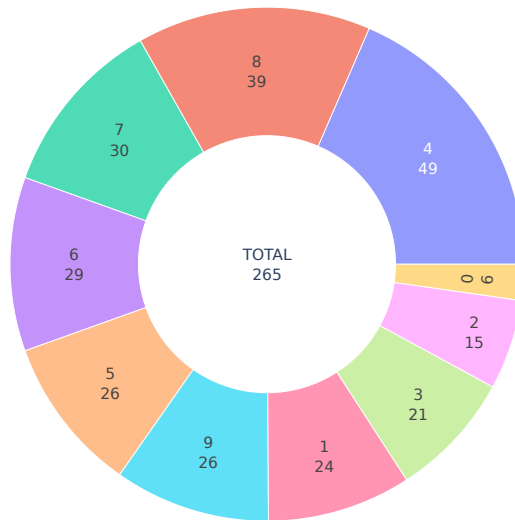


Figure 8.7: 10 latent topics from **Constitutional Provisions**.

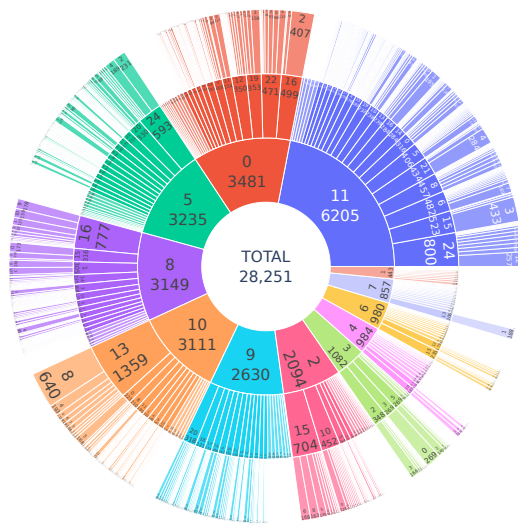


Figure 8.8: 985 latent topics from **Statutory Sections**.

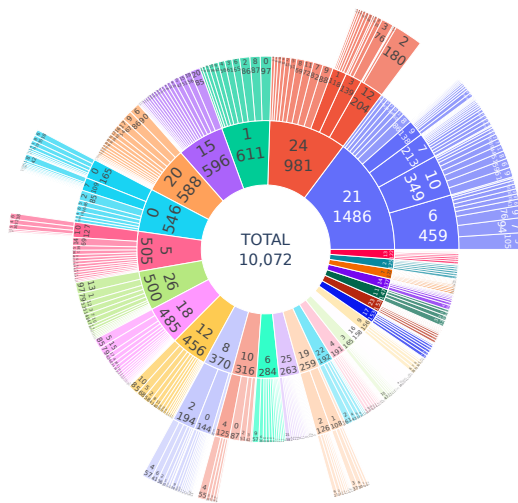


Figure 8.9: 420 latent topics from **Court of Appeals Cases**.

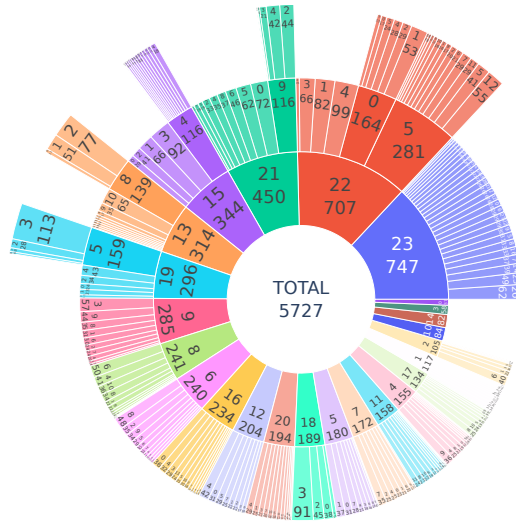


Figure 8.10: 248 latent leaf topics from **Supreme Court Cases**.

### 8.3.2 Decomposition

The four component data types were decomposed hierarchically with NMFk. The vocabulary for building the TF-IDF matrix was collected using specific parameters for each part. For the Constitution vocabulary, the minimum token document frequency (DF) was set to 5 documents, with a maximum token DF of 80% of the corpus, resulting in a final size of 416 tokens. For the Statutes vocabulary, the minimum token DF was set to 30 documents, with a maximum token DF of 70% of the corpus, yielding 7,508 tokens. The Court of Appeals vocabulary used a minimum token DF of 50 documents (cases) and a maximum token DF of 70% of the corpus, resulting in 10,189 tokens. Last, the Supreme vocabulary employed a minimum token DF of 50 papers and a maximum token DF of 70% of the corpus, with a final size of 8,425 tokens. The maximum depth was set to 2, of which only the constitution sections did not reach due to the limited number of sections. The limiting factor of further decompositions was 100 documents, so if the preceding

cluster had  $\geq 100$ , the cluster would decompose or otherwise stop. As seen in Figure 8.7, the largest H-cluster is from cluster 4 in dark blue with 49 sections, with the fewest documents in yellow with six sections in H-cluster 0. The other three decompositions can be examined in Figures 8.7, 8.8, 8.9, and 8.10, where the court cases, there are 10 leaf H-clusters in the constitution decomposition, 985 leaf H-clusters in the statutes decomposition, 420 leaf H-clusters in the Court of Appeals cases, and 248 leaf H-clusters in the Supreme Court cases. From the methods in [229], each H-cluster throughout the decomposition hierarchies has LLM-generated labels for ease of reference and quick insight. Labels for the first decomposition depth for the constitution can be observed in Table 8.1, the statutes in Table 8.2, the Supreme Court in Table 8.4, and the appeals in Table 8.3. The depth-0 H-clustering corresponds to the first ring radial from the center totals in the several Figures 8.7, 8.8, 8.9, and 8.10.

Table 8.1: NM Constitutional Depth-0 H-Clusters

#	Constitution Label
0	Irrigation and Water Resource Management Principles
1	Regulation of Private Sector Influence on Public Schools and Education Services
2	Branches of Government Structure and Functionality
3	Education Funding for New Mexico’s Educational Institutions
4	Legislative Proceedings and Lawmaking Activities Enacted During Sessions
5	Territorial Transition: Constitutional Ratification and Statehood Provisions
6	Governance, Land, and Taxation Framework
7	Municipal and County Financial Obligations and Liabilities Management

*Continued on next page*

---

# **Constitution Label**

---

- 8 Judicial Power and Jurisdictional Frameworks
  - 9 Public Service Election Governance Structure and Processes
- 

Table 8.2: NM Statutory Depth-0 H-Clusters

---

# **Statute Label**

---

- 0 Municipal Court Civil Cases Involving Children’s Rights
  - 1 Public Education Infrastructure Management Systems
  - 2 Criminal Codes, Local Governance Boundaries, Licensing Rules
  - 3 Comprehensive Emergency Health and Human Services Response Framework
  - 4 Taxation and Revenue Collection Oversight
  - 5 Military Decorations, Licensing Procedures, Governance Boards
  - 6 Regulatory Insurance Contract Law and Policy Analysis
  - 7 Public Obligations Financing and Project Bonds Issuance
  - 8 State Government Investment Grants for Education and Economic Development
  - 9 Irrigation and Water Rights Regulations
  - 10 Military Honors, Discrimination Penalties, and Trust Authority
  - 11 Corporation Governance Framework and Regulatory Compliance
-

Table 8.3: NM Appeals Court Case Law Depth-0 H-Clusters

---

#	Appeals Court Label
0	Real Property Rights and Interests
1	Civil Liability and Injury Issues Arising from Healthcare Services
2	Parental Rights and Custody Proceedings Involving Disputed Parental Fitness
3	Motor Vehicle Insurance and Liability Claims Processing Procedures
4	Mortgage Foreclosure and Secured Lending Frameworks
5	Arraignments, Tribal Jurisdiction, Divorce, Bond Conditions, Motor Licensing
6	Courtroom Advocacy and Representation Strategies
7	Petitioner's Right to Parental Decision Making
8	Fourth Amendment Protections Against Unlawful Searches and Seizures
9	Taxation of Gross Receipts and Sales Transactions in a Business Context
10	Key Legal Concepts Related to Sexual Offenses
11	Child Protective Services Laws and Regulations
12	Work-Related Injury and Disability Compensation Process
13	Healthcare Contract Dispute Resolution Processes
14	Marital Property and Financial Disposition
15	Legal Proceedings Involving Jury Determination of Guilt
16	Probation Supervision and Monitoring Oversight Process
17	Juvenile Protection, Miranda Rights, Evidence Collection
18	Firearms, Substances, DWI, and Sentencing

---

*Continued on next page*

---

**# Appeals Court Label**

---

- 19 Sentencing Enhancements for Serious Repeat Offenders
  - 20 Traffic Stop Under Suspicion with Mandatory Blood Alcohol Testing
  - 21 Criminal Proceedings Trial Litigation Documentation and Record-Keeping Procedures
  - 22 Workers' Rights and Insurance Benefits
  - 23 Children's Welfare and Family Reunification Efforts
  - 24 Public Municipal Legal Frameworks and Governance Structure
  - 25 Business Disputes, Wrongful Injury, Taxation Appeals, Workers' Compensation
  - 26 Administrative Disputes, Native American Legal Memorandums
- 

Table 8.4: NM Supreme Court Case Law Depth-0 H-Clusters

---

**# Supreme Court Label**

---

- 0 Arbitration of Contract Disputes and Judicial Decision-Making in Motor Vehicle Cases
  - 1 Revenue and Taxation Frameworks in Governance and Administration
  - 2 Mineral Rights Leases
  - 3 Native American Self-Governance and Tribal Jurisdictional Frameworks
  - 4 Municipal Zoning Ordinances and Regulations of Local Governance Areas
  - 5 Damages Award for Wrongful Conduct Against Business Partners
  - 6 Jurisdictional Appeals, Juvenile Sentencing, Felony Enhancements, Parole Terms
  - 7 Post-Divorce Asset Distribution and Management Strategies
  - 8 Constitutional Challenges to Public Education Governance
- 

*Continued on next page*

---

**# Supreme Court Label**

---

- 9 Secured Financial Instruments and Property Rights
  - 10 Water Resource Allocation and Management
  - 11 Electric Utility Rate Regulation Oversight Authority
  - 12 Law Enforcement Procedures and Rights Protection under Fourth Amendment Protections
  - 13 State Legislative Review Process Decisions
  - 14 Denial of Petition for Habeas Corpus
  - 15 Workers' Compensation Process for Work-Related Injuries and Disabilities
  - 16 Real Estate Boundary Disputes and Conveyance Matters
  - 17 Parental Rights and Legal Guardianship Proceedings
  - 18 Appellant's Argument Against the Ruling of a Trial Court
  - 19 Document Terms and Conditions Regarding Business Transactions
  - 20 Mutual Insurance Policies for Vehicle and Individual Coverage
  - 21 Accident resulting from driver error on public roadways leads to legal consequences
  - 22 Court proceedings involving criminal trials and testimonial evidence
  - 23 Civil Litigation Proceedings and Trials
- 

### 8.3.3 Knowledge Graph

The four data parts, 265 constitutional provisions, 28,251 Statute sections, 5,727 Supreme Court cases, and 10,072 Court of Appeals cases, were inserted into the neo4j [162] knowledge graph. The graph's number of nodes and edges can be seen in Table 8.5, where edge counts are where the

triplet's tail originates with the row's node. The legal citations were collected by iterating the text of each case or section into chat-gpt-3.5-turbo with the following prompt: "You are an expert legal document analyzer. Your job is to find all references to the Constitution, Case Law, or Statutes in the text." The result was that the LLM acted like a named entity extractor, such that any citations in the text were pulled out in an enumerated list. The citations mainly included the cases, statutes, and constitution of New Mexico, but also had references to the United States Constitution, New Mexico Administrative Code, and New Mexico Rules Annotated (NMRA). The NMRA had many references to Uniform Jury instructions and the rules of criminal and civil proceedings.

In figure 8.11, the NMFk topic keyword and a bag-of-words vocabulary were both queried for 'estoppel.' The NMFk keyword is red, the topics show in a light blue node, and the BOW node is purple. Three node types occur for this keyword: the statutes are green nodes, the Court of Appeals cases are yellow, and the Supreme Court cases are orange. The constitution neither clustered over the term nor mentions it, which is not represented in Figure 8.11. There is a difference in the data that have topics associated with estoppel vs all of the documents that mention 'estoppel'. Still, not all documents mentioning 'estoppel' were clustered with the word, which means that other terms and concepts from those terms had more importance for the documents on the left side of the image than 'estoppel'. There are 14 topics, 441 Court of Appeals cases, 276 Supreme Court Cases, and 136 Statutes in Figure 8.11.

These topics have 'estoppel' in their top keywords, whereas if every topic that contained 'estoppel' in BOW words were called, there would be 328 topics. Of the 14 topics, one was connected to the Statutes, "Collection and recovery of liabilities made to board members with errors and omissions". Three of the 14 topics were connected to Supreme Court cases: "Employment Rights and Property Disputes in New Mexico Municipal Affairs", "Public Corporation Property Taxation Matters and Disputes with Licensing Authorities", "Drilling and

Table 8.5: Neo4j Node and Edge Overview

Node Type	Nodes	Out Edges	Legal Cites
NMFk Topics	2,469	92,634	–
NMFk Keyword	11,076	8,281,843	–
BOW Vocabulary	132,423	–	–
Constitution	265	9,067	41
Statute	28,251	1,930,707	81,353
Supreme Court Case	5,727	2,437,161	76,478
Court of Appeals Case	10,072	4,176,288	131,230
<b>Total Unique</b>	<b>190,283</b>	<b>16,927,700</b>	<b>289,102</b>

Gas Agreement Terms Regarding Oil Wells". Finally, the remain 10 of the 14 topics with 'estoppel' in its top words were connected to cases from the Court of Appeals: "Litigation outcomes and jurisdictional limitations", "Administrative License Revocation Proceedings by Division Officers", "Legal Proceedings and Litigation Issues in a Medical Context", "Corporate Governance and Financial Management Matters", "Dispute Resolution Process for Agricultural Property Transactions", "Motor Vehicle Administrative License Actions", "Insanity Defense Expert Witness Testimony", "Malpractice claim within time constraints", "Employer Liability for Federal Disability Claims Against Administration Agencies", "Criminal offenses and doctrine involve multiple types of larceny charges", "Collection and recovery of liabilities made to board members with errors and omissions", "Public Corporation Property Taxation Matters and Disputes with Licensing Authorities", "Employment Rights and Property Disputes in New Mexico Municipal Affairs", "Drilling and Gas Agreement Terms Regarding Oil Wells".

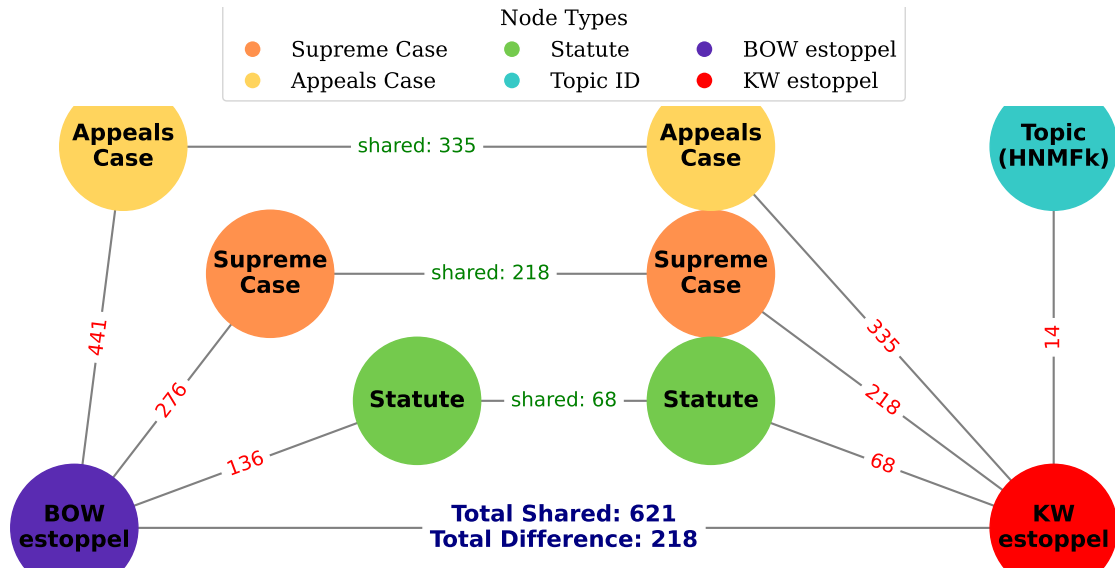


Figure 8.11: Examination of ‘Estoppel’ relating to being a keyword in topics, vs bag of words vocabulary.

### 8.3.4 Domain-Specific Evaluation

Questions were formulated to query information about legal concepts from different LLM channels. In one experiment, five questions from each data source, namely, the constitutional provisions, the statutes, the Court of Appeals opinions, and the Supreme Court cases, were followed by five additional quantity-based questions.

To assess our domain-specific RAG system’s accuracy in retrieving quantitative data without hallucinations from our KG and to verify the system’s integrity, we generated the first group of 25 questions using ChatGPT-3.5. These questions were then verified through the SME lawyer. This method reflects established practices where LLMs create evaluation questions. Additionally, our evaluation targeted the system’s ability to answer highly specific queries assessable through non-

expert term searches, highlighting the limitations of general models in addressing such inquiries.

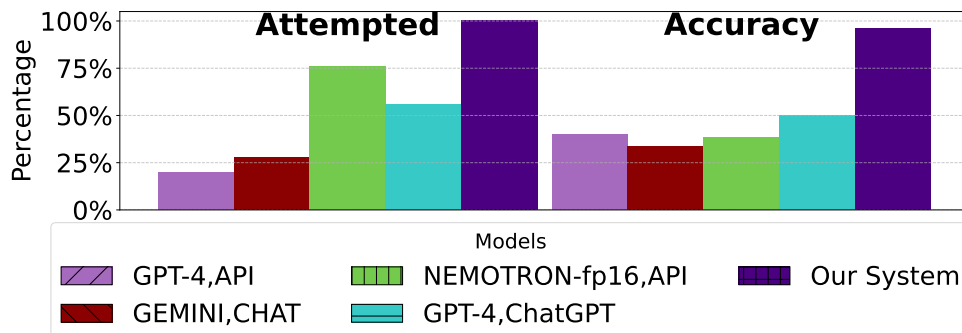


Figure 8.12: 25 questions queried across five different LLM channels. “Attempts” indicate the percentage of responses that tried to answer the question, and “accuracy” represents the percentage of correct attempts.

The first 25 questions were used to generate the attempted accuracy results shown in Figure 8.12. In another experiment, a total of 60 questions were formulated by a legal subject matter expert.

The models evaluated differ slightly between the two experiments. For the 25-question study (see Figure 8.12), the systems compared were: OpenAI’s GPT-4 accessed via the API, Google’s Gemini via web chat, Nvidia’s nemotron:70B-Instruct, OpenAI’s GPT-4 accessed via a web chat, and our system.

Answering legal questions becomes increasingly challenging with larger datasets. For example, while statutes have a natural hierarchical organization that enables LLMs to train on and summarize them internally, case law consists of unstructured, lengthy texts that are more difficult to process. Although constitutional questions were more manageable, many models mentioned only articles (rather than specific sections) when finer detail was requested, citing frequent changes in how sections are enumerated.

Our evaluation procedure marks an attempt as zero if a model states it cannot answer the question, and as one for any non-empty attempt. Accuracy is then measured on a scale from 0 to 3:

- 3 is awarded for an entirely correct or nearly correct answer (allowing for some uncertainty in absolute quantities),
- 2 for a response that is primarily correct but with minor misunderstandings,
- 1 when some truth is present despite significant error,
- and 0 for completely wrong or hallucinated responses.

Occasionally, models were given points for close numeric estimates even if those were based on flawed “database lookups.” In contrast, our system provides traceable reasoning by referencing a KG and by analyzing the decomposed hierarchical structure of the documents.

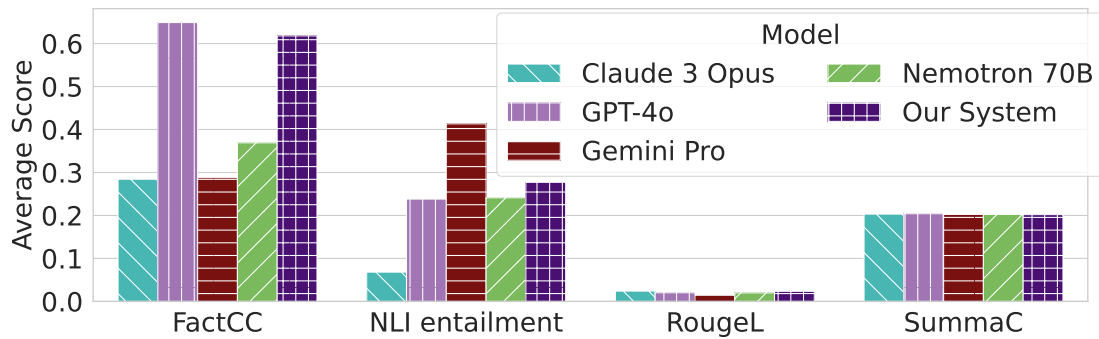


Figure 8.13: Comparison of metrics (ROUGEL, NLI entailment, SummaC, and FactCC) across models: Nemotron-70B-Instruct, Claude 3 Opus, GPT-4o, Gemini Pro, and our system.

## 8.4 Malware Analysis Research

This section discusses the identification of optimal clusters for tensor decomposition, vectorization of the dataset, construction of KG, and compares the system with GPT-4-instruct [169] as the operating model of **SMART-SLIC** to answer research questions. The same model was used to answer without RAG as well. The findings highlight the accuracy and reliability of the **SMART-SLIC**'s RAG.

Table 8.6: Labels for Topic Clusters

#	Label	# Docs.	Percent
0	Malware Behavioral Analysis	158	1.80
1	Cybersecurity Challenges	305	3.47
2	Cybersecurity Research	114	1.30
3	Botnet Detection Techniques	142	1.62
4	Malware Feature Selection And Extraction	353	4.02
5	Network Intrusion Detection	134	1.52
6	Evaluation of Malware Classifiers	301	3.42
7	Malicious Code Analysis	827	9.41
8	Artificial Intelligence for Malware	888	10.10
9	Nonnegative Matrix Decomposition	520	5.92
10	Security Threat Mitigation	180	2.05
11	Deep Learning for Malware	113	1.29
12	Machine Learning Techniques	275	3.13

*Continued on next page*

#	Label	# Docs.	Percent
13	Education Technology	447	5.09
14	Unsupervised Anomaly Detection	372	4.23
15	Ransomware Prevention	147	1.67
16	Temporal Graph Forecast	307	3.49
17	Mobile Malware Detection	230	2.62
18	Adversarial Defense Strategy	358	4.07
19	IoT Security	238	2.71
20	Privacy Protection Challenge	628	7.14
21	Sparse Tensor Decomposition	212	2.41
22	Backdoor Detection	350	3.98
23	Neural Network Architecture	581	6.61
24	Malware Analysis Techniques	610	6.94

#### 8.4.1 Dataset

The SME initially selected 30 documents specializing in large-scale malware analysis and anomaly detection with tensor decomposition fields as the core documents to construct the data. These documents were expanded along the citation/reference network two times. The final dataset included 8,790 scientific publications. From the cleaned corpus, the tensor object was generated.

#### 8.4.2 Extraction of Latent Features

After setting up the tensor, the most coherent grouping is determined by iterating through a range of  $k = \{1, 2, 3, \dots, 45\}$  clusters to decompose. The analysis determined that 25 topic clusters

represented the optimal division across all evaluated  $k$  values. The decomposition was executed using **T-ELF** on high-performance computing resources, specifically two AMD EPYC 9454 48-Core Processors. This setup provided 192 logical CPUs, enabling us to complete the entire decomposition process in approximately 2 hours. After the decomposition, post-processing refined and defined clusters for the topics listed in Table 8.6.

### 8.4.3 Vector Store

The 8,790 documents were vectorized and ingested into the Milvus vector store. When questions are posed to the framework, they are also vectorized using this model. Of the total documents, 22% had full-texts available, vectorized into the Milvus. Each document and full-text had a DOI, with the full-texts also including paragraph identifiers.

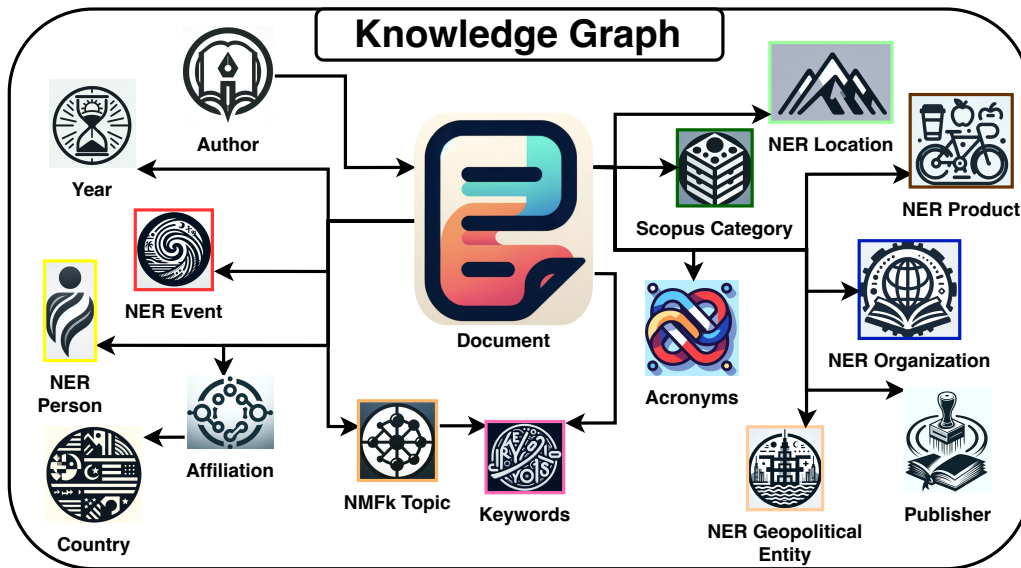


Figure 8.14: The KG schema. Images generated with DALL·E [170].

#### 8.4.4 Knowledge Graph

From the 25 clusters output from **T-ELF**, the data was formatted into 1,457,534 triplets. Once injected into the knowledge graph, there were 321,122 nodes and 1,136,412 edge relationships. The nodes injected into the graph are represented in Figure 8.14, where they are organized into 16 base categories, referred to as labels, that define the foundational classes for the injection process. Once the graph was built, it was directly queried for information as Structured Query Language (SQL) is directly queryable outside of an application. In Figure 8.15, the knowledge graph for the SME keyword related to cybercrime is queried. The query is structured as:

```
MATCH (k:Keyword)-[r1]-(d:Document)-[r2]
-(aff:Affiliation)-[r3]-(c:Country )
WHERE k.term CONTAINS 'cybercrime'
RETURN k,r1,d,r2,aff,r3,c
```

Several relationships were navigated to retrieve the country nodes from a keyword. First, from the keyword to documents, then from documents to affiliations, and finally from the affiliations to the countries. In the cypher query, these links are denoted as an *r* with an integer following, where *r* is the relationship identifier. The syntax is ()-[]-()-[]-()-[], where brackets are relations and parentheses are nodes. In the first part of the “where” clause, the keyword label is further tailored to the keyword node, such that it must contain “cybercrime.” Overall, this can answer the question, “Which countries have published papers that mention cybercrime?” The question’s retrieved nodes in Figure 8.15 have 29 countries in red, 99 affiliated institutions in yellow, and 65 published documents in blue.

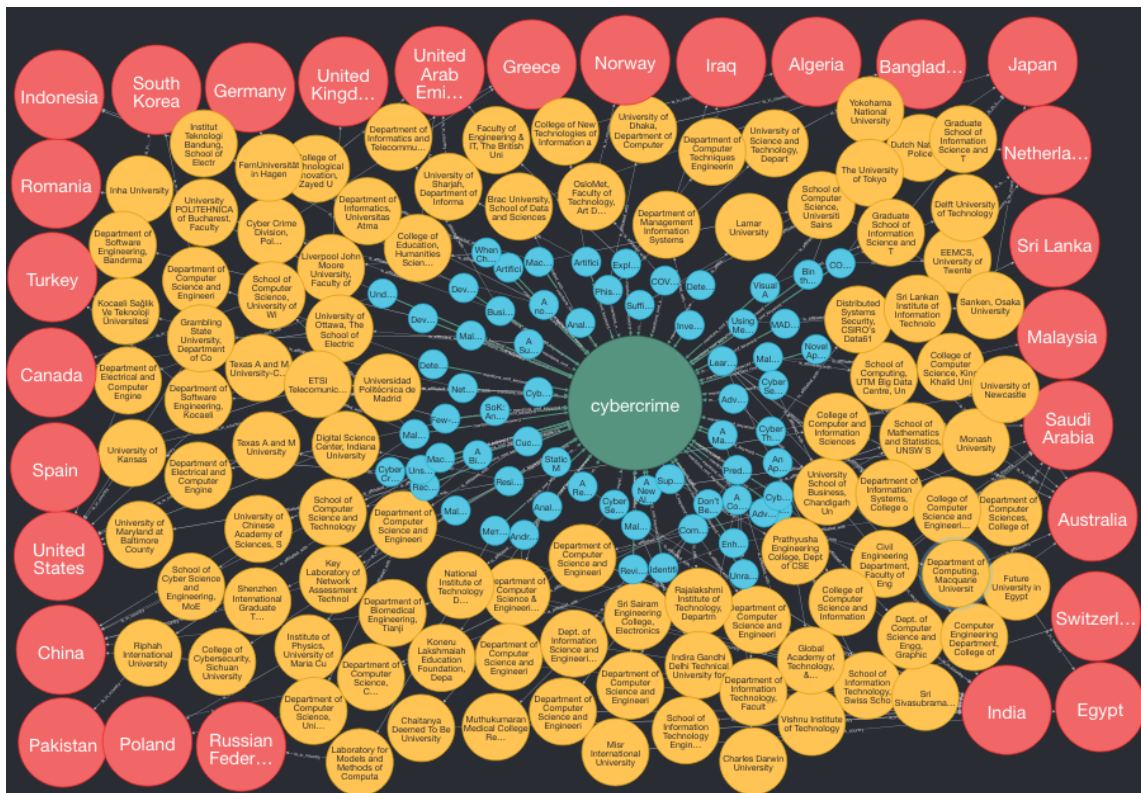


Figure 8.15: Keyword 'cybercrime' graph search. A single keyword (green) and linked documents (light blue) are returned. The documents also link affiliated institutions (yellow) and the country of the institutions (red).

## 8.5 Conclusion

Viewed in its entirety, the set of case studies presented in this chapter affirms the dissertation's core contributions: scalability, interpretability, and applicability across distinct domains. The same methods that support large-scale factorization on high-performance computing resources also enable usable systems for legal practitioners and materials scientists, without relying on opaque or ad

hoc techniques. By emphasizing detailed evidence over isolated metrics, the discussion highlights the broader value of the approach, a unified architecture capable of ingesting, structuring, and presenting knowledge in a form that users can understand and trust. With this empirical foundation established, the dissertation now turns to the final component, showing how Tensor-Structured RAG builds on the same graph and vector infrastructure to support reliable, context-aware question answering. This completes the transition from document retrieval to reasoned response that defines the overarching goal of the system.

## Chapter 9

# AI-Reasoning: Missing Links

This chapter shifts the emphasis from demonstrating functionality to evaluating reliability. Using a spectrum of graphs from controlled synthetic constructs (Dog, Swimmer, Gaussian) through biologically rich protein–protein-interaction (PPI) networks to a newly curated corpus of TMDs for superconductivity link prediction, we probe how the WNMfK, RNMfK, and BNMfK families behave under varying graph size, sparsity, and noise. Three themes consistently emerge. First, Binary Bleed continues to identify a compact rank that preserves signal while containing computational cost. Second, Boolean and real-valued decompositions reveal complementary latent factors that remain interpretable across domains. Third, ensemble-based uncertainty estimates stay well-calibrated, enabling the system to defer judgment when evidence is thin. This is an ability confirmed not just on synthetic data and PPIs, but also on the TMD link-prediction task, where masked superconducting links are rediscovered with near-perfect top-3 precision. Together, these findings strengthen the dissertation’s claim that a unified, principled architecture generalizes across domains without manual hyperparameter tuning.

The results are evaluated on the synthetic datasets (Dog, Swimmer, and Gaussian Data) by randomly sampling the missing links 10 times for cross-validation to ensure statistical significance. For the Boolean datasets, Dog Data and Swimmer Data, missing links are randomly sampled by stratifying on 0s and 1s. This enables us to test the methods’ predictive capabilities on negative and positive interactions or known links. Additionally, we test the methods against data sparsity on the Dog and Gaussian datasets by systematically increasing the test-set size from 10% to 90% in increments of 10%.

### 9.0.1 System Configuration

The experiments were conducted on an HPC cluster named Dracarys at the Los Alamos National Laboratory (LANL). Dracarys uses the AMD EPYC 9454 48-core processor at a clock speed of 3.81GHz. There are 192 virtual processors and a total physical RAM of 1.97 TeraBytes (TBs). The system also comprises 8 NVIDIA H100 GPUs with VRAM memory of 82 GigaBytes (GBs) each.

This section presents method performances, starting with synthetic datasets under Boolean and non-Boolean settings. We evaluate rank prediction, link prediction, performance under increasing sparsity, and UQ utility in controlled settings with pre-determined ranks. Next, we showcase results on PPI networks, demonstrating real-world effectiveness.

## 9.1 PPI Datasets

This paper uses five PPI network datasets from [177]. Specifically, the PPI networks include Brain, Disease of Metabolism, *H. sapiens*-extended, Liver, and Neurodegenerative Disease. The statistics of these five PPI networks are summarized in Table 9.1. Pre-processing of these datasets involves removing conflicting or inconsistent interactions. For example, cases where protein  $i$  is shown to

Table 9.1: The statistics of the five PPI datasets include the number of proteins and positive and negative interaction pairs of each PPI network when represented as a matrix.

Properties/Dataset	H.sapiens-extended	Brain	Disease of Metabolism	Liver	Neurodegenerative Disease
Number of proteins	14,407	11,167	1,036	10,627	820
Number of positive pairs	157,950	225,200	5,131	218,239	5,881
Number of negative pairs	157,300	223,130	5,123	215,984	5,879

interact positively with protein  $j$ , but the dataset also includes a second entry indicating a negative interaction between the same proteins, are excluded. Additionally, proteins that interact with fewer than five other proteins are removed, serving as a pruning approach to ensure the inclusion of proteins with sufficient interactions to allow meaningful predictions [65].

The experiments benchmark against the symNMF method results reported in [177]. At the same time, it is unclear whether [177] applied similar pre-processing considerations, notably removing conflicting protein interactions. Therefore, the analysis also includes a comparison with LMF (symLMF without the symmetry constraint) on the pre-processed datasets used in this paper.

Next, benchmarks are presented for the methods against LMF and symLMF [177]. Table 9.2 presents the benchmark results on the PPI networks: *H. sapiens*-extended, Brain, Disease of Metabolism, Liver, and Neurodegenerative Disease. For BNMFk, we employ k-means clustering for Boolean thresholding, as described in Section 7.1.7. At the same time, RNMFk and WNMFk are evaluated without Boolean settings, following their original implementations as described in Sections 7.1.1 and 7.1.2, respectively. Additionally, we include the LMF-extended results for WNMFk<sub>lmf</sub>, BNMFk<sub>lmf</sub>, and RNMFk<sub>lmf</sub>, as described in Section 7.1.4. Performance metrics include ROC AUC, PR AUC, and UQ-based scores, detailed in Section 7.1.10.

Table 9.2 shows BNMFk has the lowest performance, while RNMFk and WNMFk consistently

match or outperform LMF and symLMF across datasets. For example, on H.sapiens-extended, RNMFk achieves an ROC AUC of 0.941, compared to 0.949 for LMF and 0.944 for symLMF, with a higher PR AUC of 0.957 (vs. 0.934 for LMF and 0.955 for symLMF). Ensemble models yield higher performance, with BNMFk<sub>lmf</sub> achieving the highest scores across datasets, showing that combining models with LMF improves performance over standalone versions. On the Brain PPI dataset, BNMFk attains a PR AUC of 0.849, RNMFk 0.953, and WNMFk 0.948, while their LMF extensions reach 0.962, 0.960, and 0.959, respectively. These scores surpass LMF alone, indicating complementary model strengths under the ensemble framework. UQ-based weighting slightly improves scores in some cases, as shown by underlined ROC AUC and PR AUC values, though the gains are small for these datasets. These results on five real-world PPI networks validate the efficacy of the methods beyond synthetic datasets. The following section summarizes the technical capabilities and software development considerations behind the Python library.

## 9.2 Synthetic Data

The first dataset used in the experiments, referred to as the Dog dataset [51], is constructed by mixing four binary images. This dataset includes four distinct images—sun, person, dog, and cloud, each with dimensions  $20 \times 20$  pixels. The Dog dataset is illustrated in Figure 9.1. To generate the dataset, the columns of each image are stacked along the first axis. Then they create 16 unique samples by combining these stacked representations using Boolean addition, considering all possible combinations of the four images. This results in a matrix  $\mathbf{X} \in \{0, 1\}^{400 \times 16}$ . Since the dataset comprises four distinct images, the true Boolean rank of the Dog dataset is  $k = 4$ .

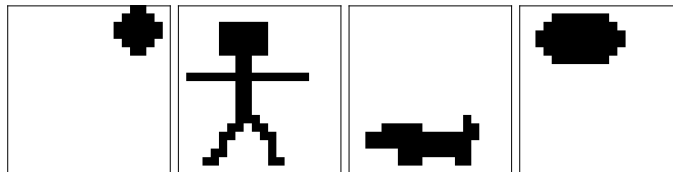


Figure 9.1: **Dog Dataset** - Four binary images are used as Boolean latent features to generate the synthetic data of shape  $400 \times 16$ .

### 9.2.1 Dog Data

The results on the Dog Data are presented in Figures 9.2, 9.3, and 9.4, where each column displays the performance of the methods under different settings.

Figure 9.2 compares  $\text{BNMF}_{k_{\text{kmeans}}}$ ,  $\text{NMF}_k$ , and  $\text{WNMF}_k$ , where the  $k$ -means subscript denotes Boolean thresholding via  $k$ -means clustering (Section 7.1.7). In the first row,  $\text{BNMF}_{k_{\text{kmeans}}}$  predicts the true rank  $k = 4$  more consistently than  $\text{NMF}_k$  and  $\text{WNMF}_k$ , as expected for a Boolean decomposition method on Boolean Dog Data. However, for test-set sizes above 0.4,  $\text{BNMF}_{k_{\text{kmeans}}}$  shifts toward  $k = 1$ , while  $\text{NMF}_k$  and  $\text{WNMF}_k$  frequently predict ranks above  $k = 4$ , capturing non-Boolean structures. Both  $\text{BNMF}_{k_{\text{kmeans}}}$  and  $\text{WNMF}_k$  predict lower ranks as sparsity increases. The second row shows RMSE scores, where lower values indicate better link prediction.  $\text{BNMF}_{k_{\text{kmeans}}}$  achieves the lowest RMSE at smaller test sets, with RMSE increasing as test-set size grows, limiting pattern learning. RMSE is minimized near  $k = 4$ , emphasizing the importance of correct rank selection.

Figure 9.3 expands on these results, comparing Boolean thresholding techniques for  $\text{BNMF}_k$ ,  $\text{NMF}_k$ , and  $\text{WNMF}_k$  (Section 7.1.7). Each column represents a method using a different thresholding approach, denoted by subscripts **kmeans**, **otsu**, and **search**, with **search** referring to coordinate descent-based thresholding. For instance,  $\text{BNMF}_{k_{\text{otsu}}}$  applies Otsu thresholding. The first

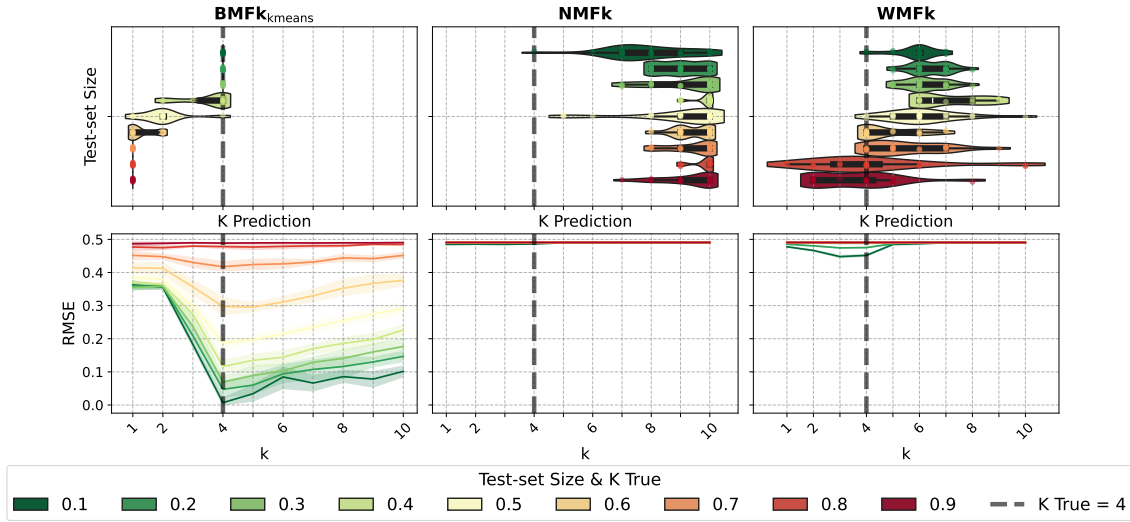


Figure 9.2: Results for Dog Data across methods including  $\text{BNMFk}_{\text{kmeans}}$ ,  $\text{NMFk}$ , and  $\text{WNMFk}$ . Boolean thresholding is not used for  $\text{NMFk}$  and  $\text{WNMFk}$ . The first row presents violin plots visualizing the rank  $k$  predictions at different test-set size levels. The second row displays RMSE scores for the test set, demonstrating the missing link prediction performance. The results are reported for each rank  $k$  on the x-axis, with the dark/dashed vertical line across columns being the true rank  $k = 4$ .

row presents rank prediction violin plots, while the second row reports RMSE scores. Compared to Figure 9.2, Boolean thresholding helps  $\text{NMFk}$  and  $\text{WNMFk}$  predict  $k = 4$  more frequently, though they still tend to overestimate ranks at smaller test sizes, whereas  $\text{BNMFk}$  predicts  $k = 4$  or lower. RMSE results confirm Boolean settings improve link prediction, especially for  $\text{WNMFk}_{\text{kmeans}}$  and  $\text{WNMFk}_{\text{search}}$ .  $\text{BNMFk}$  remains the best for rank prediction and lower RMSE, except with Otsu thresholding, which yields higher RMSE values. Across all methods, a trend of improved RMSE scores at or around the true rank can be observed in this figure, further highlighting the importance of rank selection.

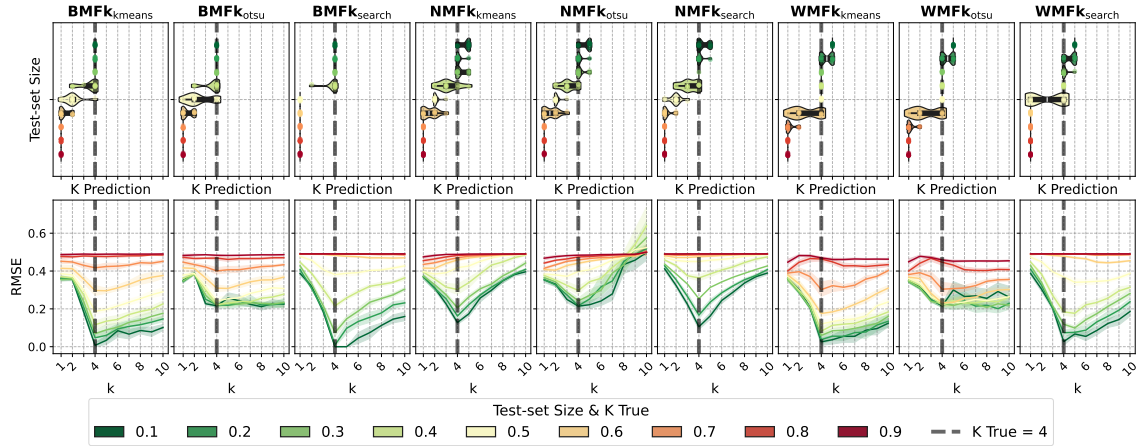


Figure 9.3: Results for Dog Data across methods, including BNMFk, NMFk, and WNMfK, evaluated under different Boolean thresholding techniques. The Boolean thresholding techniques are denoted with the subscripts of **kmeans**, **otsu**, and **search** (coordinate descent). The first row presents violin plots visualizing the rank  $k$  predictions at different test-set sizes. The second row displays RMSE scores for the test set, demonstrating the missing link prediction performance. The results are reported for each rank  $k$  on the x-axis, with the dark and dashed vertical line across each column representing the true rank  $k = 4$ .

Figure 9.4 highlights UQ's role in improving predictions. The first row shows the RMSE change after applying UQ, where negative values indicate improved performance (reduced RMSE value) by abstaining from uncertain predictions. The second row presents the fraction of abstained predictions at each test-set size, with higher fractions indicating lower coverage. UQ effectively filters uncertain predictions, lowering RMSE in several cases compared to Figure 9.3 (without UQ). Methods like  $\text{BNMFk}_{\text{search}}$  and  $\text{WNMFk}_{\text{kmeans}}$  show notable RMSE reduction. Abstention rates generally increase with test-set size, reflecting greater uncertainty in sparse data, until around a test-set size of 0.8. Beyond this point, the training set becomes too small, resulting in no meaningful abstentions.

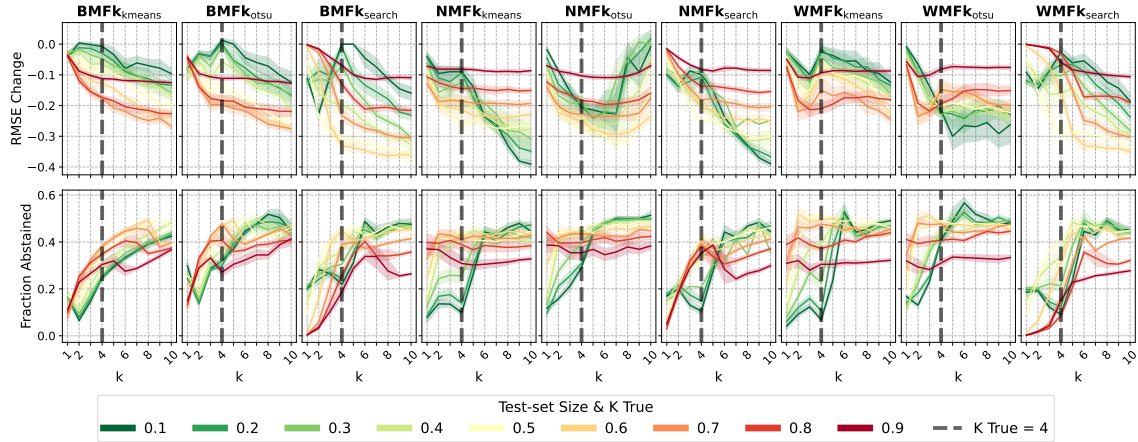


Figure 9.4: Results for Dog Data across methods, including BMFk, NMFk, and WMFk, evaluated under different Boolean thresholding techniques. The Boolean thresholding techniques are denoted with the subscripts of **kmeans**, **otsu**, and **search** (coordinate descent). The first row presents the change in RMSE after making predictions on the non-abstained samples, where a negative change indicates improved performance (lower RMSE) for the given fraction of abstained samples. The second row illustrates the fraction of abstained predictions, representing the proportion of cases where the model opted not to make a prediction due to uncertainty. The results are reported for each rank  $k$  on the x-axis, with the dark and dashed vertical line across each column indicating the true rank  $k = 4$ .

However, at the true rank  $k = 4$ , abstention rates decline for several methods, which, combined with observations in Figure 9.3 showing improved performance at true rank, indicate higher confidence in predictions at the correct rank.

Figure 9.5 shows the computational time for each method in seconds. While Figure 9.2 demonstrated that coordinate descent-based Boolean thresholding yields better results, its improvement comes at significantly higher computational time. As shown, methods using the search setting

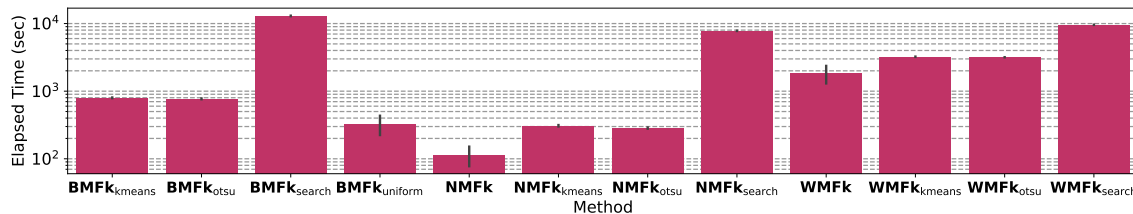


Figure 9.5: Computation times for each method are shown in the log scale on the Dog Data. The results indicate that methods leveraging coordinate descent (search) for Boolean thresholding require significantly longer computation times than other approaches.

have substantially longer run times. Given this, we exclude the search method from subsequent experiments.

## 9.2.2 Swimmer Data

The second dataset used in the experiments is the Swimmer dataset [141], which contains 16 synthetic images of a "stick figure" swimming. Each image has dimensions of  $32 \times 32$  pixels. Examples from this dataset are displayed in Figure 9.6. As shown in the first and third rows of Figure 9.6, the images contain "splashes" appearing as high-intensity values away from the swimmer or locations with low values of the swimmer figure, representing that location being under the water. Otsu thresholding is applied to convert the dataset into a Boolean format, with the resulting Boolean images displayed in rows 2 and 4 of Figure 9.6. The locations of the "splashes" and the places where the swimmer is under the water are visible in the Boolean version of the dataset. To construct the Boolean dataset, each Boolean image is stacked along the first axis and combined using Boolean addition into 256 different samples, resulting in a matrix  $\mathbf{X} \in \{0, 1\}^{1024 \times 256}$ . The true rank of the dataset is  $k = 16$ , corresponding to the 16 distinct swimmer figures.

The Swimmer dataset is analyzed with results in Figure 9.7. The test-set size is fixed at 0.1 (10%)

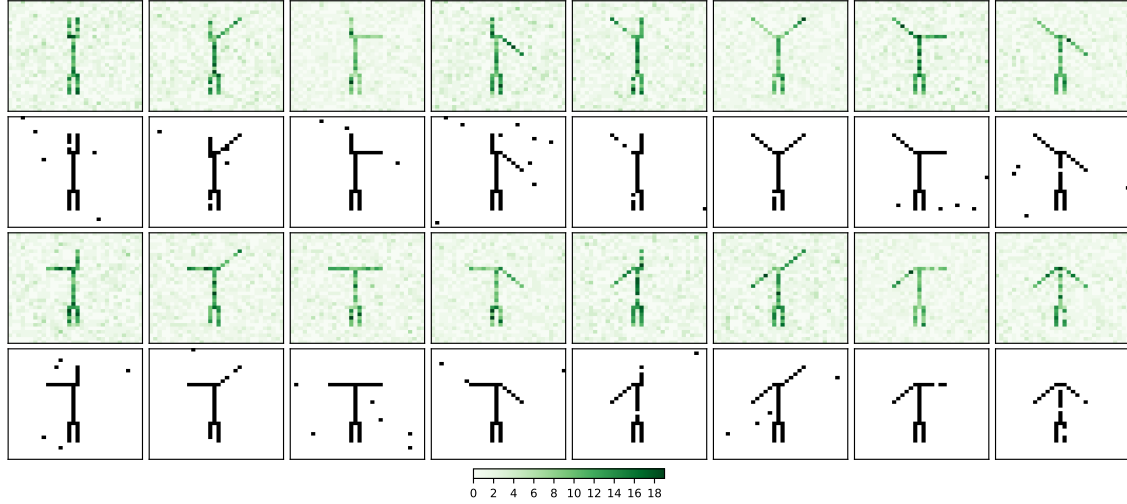


Figure 9.6: **Swimmer Dataset** - Dataset of 16 swimmer images. The first and third rows are the images with real-valued intensities ranging from 0 to 19. The second and fourth rows display the Boolean versions obtained after applying Otsu thresholding. For analysis, the Boolean versions of the dataset are represented as a matrix of size  $1024 \times 256$ .

without sparsity analysis. Figure 9.7 shows  $k$ -predictions (first column), RMSE scores (second), RMSE with abstention (third), and the fraction of abstained samples (fourth). The true rank  $k = 16$  is marked by a dark dashed vertical line, with the x-axis representing  $k$  values. Each method is color-coded, while line styles distinguish Boolean settings in columns 2-4. Figure 9.7 shows that BNMFk with Boolean settings consistently identified  $k = 16$ , as did WNMfK without Boolean settings. WNMfK with k-means thresholding predicted both  $k = 16$  and  $k = 17$ , while BNMFk<sub>uniform</sub> (without Boolean settings) always predicted  $k = 1$ . Other methods produced  $k$  values between  $k = 1$  and  $k = 17$ . Some non-Boolean methods predicted  $k = 16$  because the Swimmer dataset's non-negative and Boolean ranks are equal.

In the second column of Figure 9.7, NMFk<sub>kmeans</sub>, WNMfK<sub>otsu</sub>, WNMfK<sub>kmeans</sub>, and BNMFk<sub>kmeans</sub>

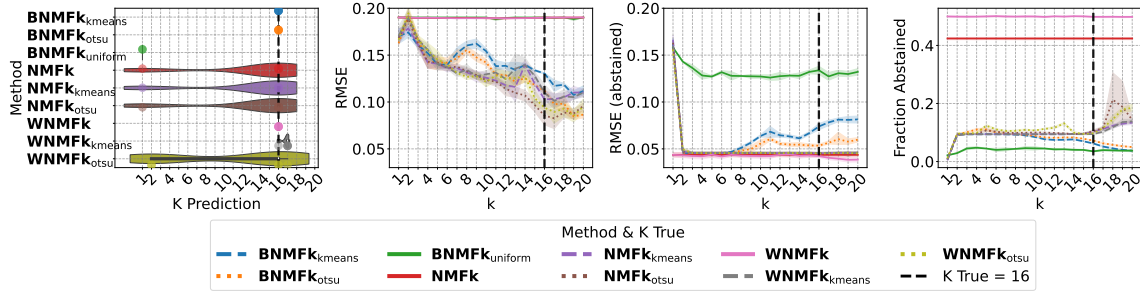


Figure 9.7: The performance of each method on the Swimmer Data is shown. The first column features a violin plot displaying the distribution of  $k$  predictions for each method, with the y-axis representing the methods and the x-axis representing the predicted  $k$  values. The second through final columns present results at each  $k$  decomposition value, with the x-axis corresponding to the rank  $k$ . The dark dashed vertical lines indicate this dataset’s true rank of  $k = 16$ . The Boolean thresholding techniques are denoted with the subscripts of **kmeans**, **otsu**, and **search** (coordinate descent), while the **uniform** subscript refers to running BNMfK without Boolean thresholding on the latent factors. NMFk and WNMfK, without subscripts, are the results of not using Boolean techniques with these methods.

improve as  $k$  nears 16, after which RMSE increases. Notably,  $\text{BNMFk}_{\text{kmeans}}$  continues improving until  $k = 18$  before RMSE rises, mirroring Dog Data trends. Non-Boolean decompositions maintain higher RMSE (lower performance) across all  $k$ , confirming Boolean methods perform better on Boolean data. The third column shows UQ significantly improves RMSE, even for non-Boolean methods, though less so for  $\text{BNMFk}_{\text{uniform}}$ . The RMSE remains stable across ranks as models abstain from uncertain predictions. However, as a trade-off, non-Boolean methods like WNMfK and NMFk exhibit a higher reduction in coverage, shown by a greater fraction of abstained samples in the last column.

The third synthetic dataset is randomly sampled matrices  $\mathbf{X}$  from a normal distribution, leveraging the framework introduced in [160]. This dataset is referred to as the Gaussian Dataset. The generated matrices  $\mathbf{X} \in \mathbb{R}_+^{50 \times m}$ , where  $m \in \{50, 100, 150, 200, 250, 300, 350, 400\}$  (8 distinct shapes), correspond to an increasing number of features. For each matrix size, the true rank is set as  $k \in \{2, 3, 4, 5, 6\}$  (5 distinct true rank values). Each matrix size and rank combination is randomly sampled ten times with different random seeds, resulting in  $8 \times 5 \times 10 = 400$  matrices. While the two Boolean datasets mentioned above and one non-Boolean synthetic dataset are used to assess the performance of the methods under controlled conditions, larger real-world datasets are used to evaluate further and validate the methods in practical scenarios.

### 9.2.3 Gaussian Data

The final synthetic dataset, Gaussian Data, is analyzed in Figure 9.8 for WNMf $k$  and RNMF $k$ , assessing their rank prediction at different test-set sizes. No Boolean settings are used. Color and line style differentiate true ranks; for example, an orange line with  $x$  markers represents  $k = 3$ . WNMf $k$  predicts the correct rank until a test-set size of 0.7, after which predictions skew toward  $k = 1$ . RNMF $k$  follows a similar trend but loses rank prediction accuracy earlier at higher true ranks. For instance, at  $k = 6$  (purple line with diamond markers), lower rank predictions appear after a test-set size of 0.5.

We examine link prediction performance using RMSE on a log scale for test-set sizes 0.1 and 0.5 in Figure 9.9, where blue lines represent 0.5 and red lines 0.1. Shapes differentiate true ranks; for example, square markers indicate  $k = 4$ , with blue squares for  $k = 4$  at 0.5. For WNMf $k$ , RMSE increases beyond the true rank, reflecting reduced performance, with the best results at the correct rank. This trend holds at higher sparsity (0.5 test-set size) but is less pronounced due to overall

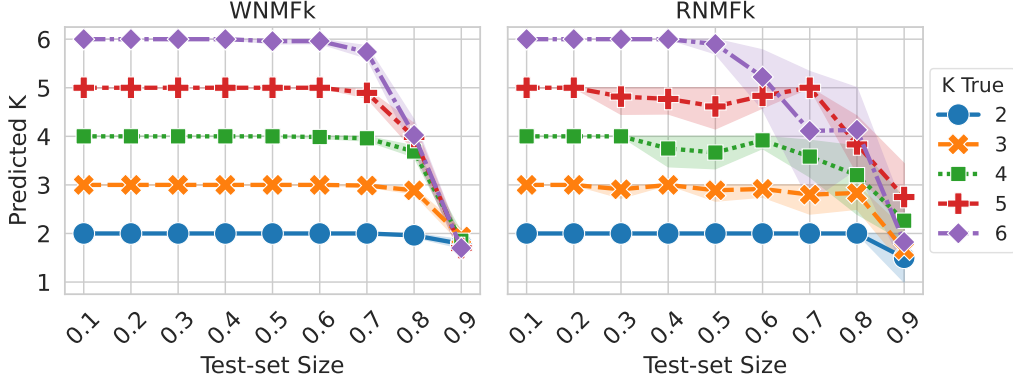


Figure 9.8: Results for  $k$  predictions on Gaussian Data are shown for WNMfK and RNMFk methods across different data sparsity levels, represented by the increasing test-set size on the x-axis. The y-axis indicates the predicted  $k$  values. The results demonstrate that  $k$  predictions remain accurate for both methods until high levels of sparsity are reached, while WNMfK results in more accurate  $k$  predictions. Line color and style differentiate between different true  $k$  values. For instance, a solid line with circle markers represents decompositions where the matrix rank was  $k = 2$ .

performance decline. RNMFk follows a similar pattern but shows more stable link prediction, as indicated by flatter RMSE lines, suggesting more consistent performance across ranks.

For Gaussian Data, we analyze the correlation between UQ and errors in Figure 9.10 to assess whether models assign higher uncertainty to higher-error points. Figure 9.10 shows Pearson Correlation Coefficients and CI between UQ and relative reconstruction errors (Equation 2.4). Columns represent test-set sizes, rows correspond to true ranks  $k = 2$  and  $k = 3$ , and the y-axis denotes correlation coefficients. Hue differentiates WNMfK and RNMFk, with color bars indicating decomposition results for each  $k$ .

Figure 9.10 shows that correlations between certainty and error remain high for RNMFk and WNMfK, confirming that UQ effectively assigns higher uncertainty to high-error points. Correlation

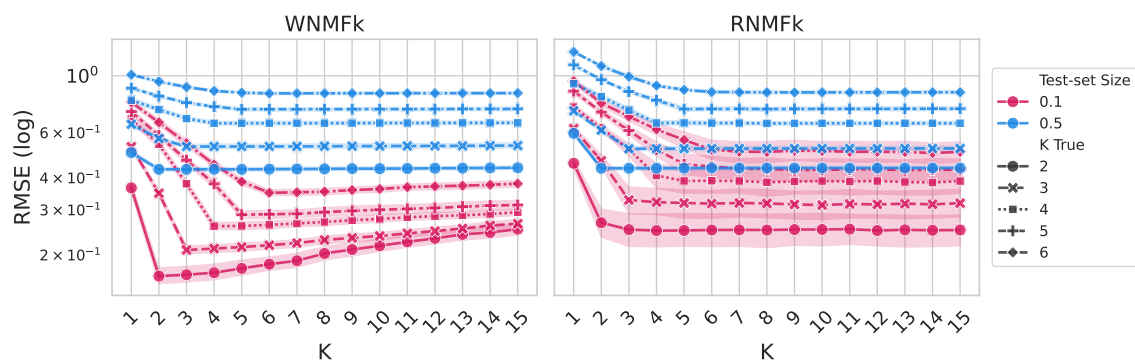


Figure 9.9: Link prediction performance on test-set sizes of 10% (red lines) and 50% (blue lines) is shown for WNMfK and RNMfK, with RMSE scores plotted on a log scale. The x-axis represents the  $k$  values, displaying results at different  $k$  decompositions. Line markers are used to differentiate between the true  $k$  values. For example, solid red lines with circle markers represent matrices with a true rank of  $k = 2$  and a test-set size of 10%.

is lower before the true rank and peaks at the correct rank, especially for WNMfK and smaller test-set sizes, indicating reliable certainty estimates. In the second row ( $k = 3$ ), the orange bars for WNMfK are notably higher, highlighting improved performance at this rank. Another key trend in Figure 9.10 is the reduction in correlation between certainty and error at the largest test-set size of 0.9. The hypothesis for this behavior is that at higher sparsity levels, where the training set is significantly reduced, the uncertainty estimates may saturate at consistently high values across all test points, thereby reducing variability. We have also observed a relevant and similar result in Figure 9.4, where the fraction of abstained samples reduced at higher test set sizes. When uncertainty does not vary meaningfully, its correlation with actual errors naturally diminishes. This saturation reflects a breakdown in the model's ability to provide meaningful uncertainty estimates, signaling that the model is operating in a regime of extreme sparsity. Under such conditions,

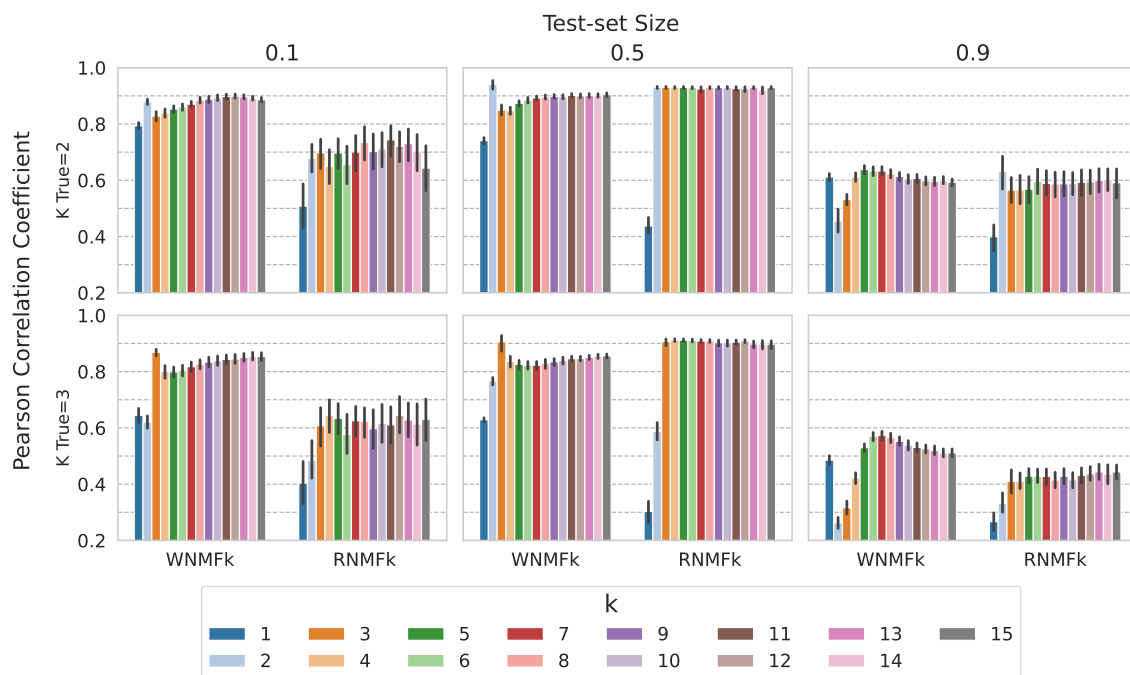


Figure 9.10: Pearson Correlation Coefficients with CI between UQ and errors at the test set are reported across different sparsity levels or test-set sizes that include 0.1, 0.5, and 0.9 (columns) and true matrix ranks  $k$  including  $k \in [2, 3]$  (rows). The shared x-axis differentiates between the methods WNMFK and RNMFk using hues, while the shared y-axis represents the Pearson correlation values. Bars with different colors correspond to results at various  $k$  decompositions.

there are significantly more missing links to predict while relying on minimal information from the training set. As a result, the model's uncertainty quantification methods become less reliable.

The "Relative-Std-to-Mean Ratio of Uncertainty" plot, shown in the top row of Figure 9.11, provides a quantitative perspective on the behavior of uncertainty estimates across varying levels of sparsity (or test-set size). This metric is calculated for the test locations based on the uncertainty

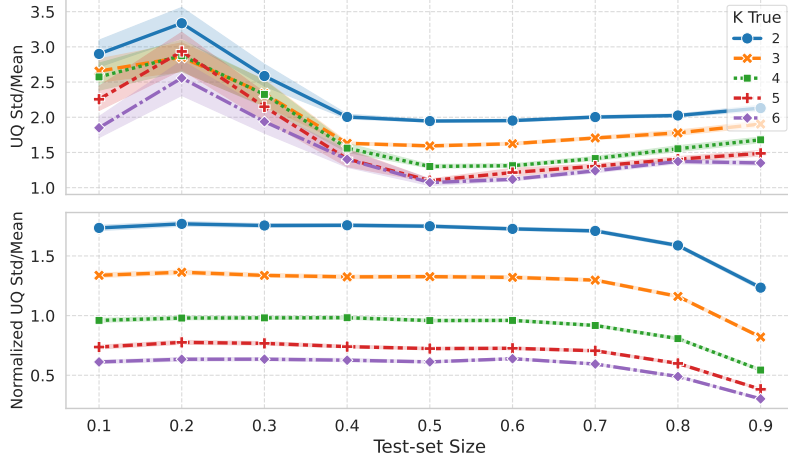


Figure 9.11: **Relative-Std-to-Mean Ratio of Uncertainty plot** provides a quantitative perspective on the behavior of uncertainty estimates across varying sparsity levels (or test set size) for each  $k$  true. The top figure shows the standard deviation (std) at test points divided by the overall average of certainty (UQ). The bottom figure investigates certainty saturation by normalizing the std-to-mean uncertainty ratio at the top plot with the correlation between the UQ and errors at test points.

matrix  $\mathbf{U}$  and is defined as:

$$\text{SMR} = \frac{\sigma_{\mathbf{U}}}{\mu_{\mathbf{U}}}, \quad (9.1)$$

where SMR is the standard deviation (std)-to-mean ratio,  $\sigma_{\mathbf{U}}$  is the standard deviation of UQ values at test locations, and  $\mu_{\mathbf{U}}$  is the mean of UQ values at test locations, calculated as:

$$\sigma_{\mathbf{U}} = \sqrt{\frac{1}{|\mathcal{I}_{\text{test}}|} \sum_{(i,j) \in \mathcal{I}_{\text{test}}} (\mathbf{U}_{ij} - \bar{\mathbf{U}})^2}, \quad \mu_{\mathbf{U}} = \frac{1}{|\mathcal{I}_{\text{test}}|} \sum_{(i,j) \in \mathcal{I}_{\text{test}}} \mathbf{U}_{ij}, \quad (9.2)$$

where  $\mathcal{I}_{\text{test}}$  is the set of test locations, and  $\bar{\mathbf{U}}$  is the mean uncertainty value at the test locations. This metric tracks the relationship between the variability of uncertainty (measured as the standard deviation of uncertainty) and the average uncertainty (mean of UQ) at the test points. By normal-

izing variability through its mean, the std-to-mean ratio captures whether the relative dispersion in uncertainty estimates decreases significantly as sparsity increases. In the top row of Figure 9.11, the std-to-mean ratio drops until around a 0.5 test-set size and stabilizes. Meanwhile, the correlation between errors and UQ remains high until a test-set size of 0.7 or 0.8. The drop in the std-to-mean ratio suggests that the variability of uncertainty at test locations is becoming more consistent relative to its mean. However, it does not necessarily indicate saturation (since the correlation remains high). The normalized std-to-mean ratio plot, shown in the second row of Figure 9.11, refines this measure by incorporating the correlation between UQ and errors. This metric is defined as:

$$\text{NSMR} = \frac{\text{SMR} \cdot r}{\max(r)}, \quad (9.3)$$

where  $r$  is the vector of correlation values between UQ and reconstruction error. This normalization adjusts the standard deviation-to-mean ratio (SMR) by weighting it with the correlation between UQ and error, scaled by the maximum correlation value observed. This scaling emphasizes variability in uncertainty estimates that predict error rather than those that fluctuate without meaningful correlation to errors. By incorporating the correlation term, this metric provides a refined measure of how uncertainty dispersion (the std-to-mean ratio) relates to actual predictive reliability. Their significance diminishes if uncertainty estimates are highly variable but poorly correlated with errors. Conversely, when variability in uncertainty remains well-correlated with errors, it suggests that the uncertainty estimates are informative.

In the second row of Figure 9.11, higher values, observed until around a test-set size of 0.7, indicate that the uncertainty estimates remain variable and meaningfully related to errors. This suggests that UQ values are informative about prediction confidence. The observed decline beyond a test-set size of 0.8, where both the std-to-mean ratio in the first row of Figure 9.11 and the correlation in Figure 9.10 are low, suggests that uncertainty estimates are saturating. This saturation implies

that the model’s certainty values no longer distinguish between correct and incorrect predictions, reducing their usefulness at higher sparsity levels.

## 9.3 Materials

The experiments test three hypotheses: (i) an  $\text{HNMF}_K \rightarrow \text{BNMF}_K + \text{LMF}$  pipeline can rediscover masked superconducting links; (ii) the posterior scores of the ensemble cleanly separate positive from negative edges; and (iii) the same model ranks unseen compounds so that known superconductors appear above chemically similar non-superconductors. These objectives guide the design choices detailed below.

### 9.3.1 Corpus construction and pre-processing

We build on a domain-specific corpus of 46,862 peer-reviewed articles that mention at least one of the 73 TMDs extracted by the BUNIE pipeline. Titles and abstracts are lower-cased, tokenised, stop-words removed, and converted to a  $V=48,712$ -term TF-IDF matrix with `min_df = 3` and `max_df = 0.98`. Documents with fewer than 20 non-zero terms are discarded, yielding a final  $46,715 \times 48,712$  matrix with 0.12% density.

### 9.3.2 Topic hierarchy ( $\text{HNMF}_K$ )

The TF-IDF matrix is factorised recursively with  $\text{HNMF}_K$ . Stable rank selection operates in the range  $k \in [2, 6]$  at every node, and the recursion stops automatically when any leaf contains  $< 150$  documents. Bootstrap stability (200 resamples) identified an optimal depth  $L = 3$ , producing  $k_1=3$ ,  $k_2=5$ , and  $k_3=18$  clusters on successive levels. The  $|\mathcal{T}| = 815$  leaf topics constitute the row index of the subsequent Materials-Property matrix.

### 9.3.3 Link-prediction Model

The ensemble first fits BNMFk with automatic rank selection to  $M$ , then freezes that rank and refines the reconstruction with Logistic Matrix Factorisation (LMF). The two reconstructions are combined as

$$\hat{M} = \sigma(WH + b_r + b_c),$$

where  $\sigma$  is the logistic link. Hyper-parameters ( $\lambda=0.1$ ,  $\text{lr}=0.01$ ,  $\text{epochs}=500$ ) were tuned on a held-out validation set.

### 9.3.4 Materials Property Matrix

The TMDs corpus comprises 46,862 peer-reviewed articles. TMDs, with the general formula  $MX_2$  ( $M$  = transition metal,  $X$  = chalcogen), are foundational to 2D materials research, known for their exceptional electronic, optical, and mechanical properties [78]. Despite only 72 known compounds, the combinatorial space of possible TMDs is vast and remains largely underexplored.

For hierarchical modeling, we convert titles + abstracts of the selected by BUNIE scientific texts to TF-IDF matrix and factorize it with HNMFk. A depth of **three** levels ( $L = 3$ ) proved optimal by bootstrap stability analysis, yielding coarse “super-topics” at the root and progressively finer subtopics at lower depths.

### 9.3.5 Top $k$ retrieval of hidden links

To evaluate whether our  $\text{BNMF}_k + \text{LMF}$  ensemble can *rediscover* true but withheld superconducting links, we perform a leave-out experiment on four benchmark transition-metal dichalcogenides:  $\text{NbSe}_2$ ,  $\text{MoS}_2$ ,  $\text{S}_2\text{Ta}$ , and  $\text{Se}_2\text{Ta}$ .

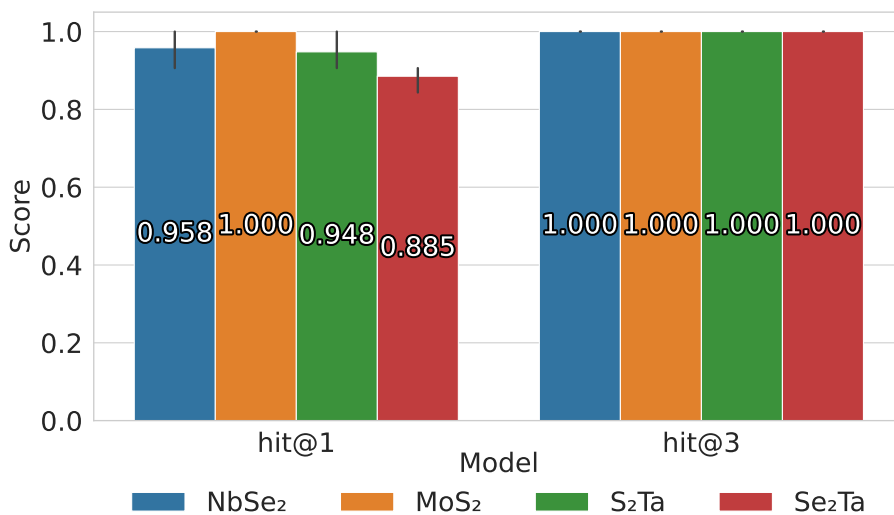


Figure 9.12: Mean hit@1 and hit@3 scores (bars, annotated) of the BNMF+LMF ensemble after masking every known superconducting link and an equal number of random zeros for each of the four benchmark compounds

- (1) **Masking.** For each compound, remove all verified superconducting links from the compound–superconductor cluster matrix, and randomly sample an equal number of non-superconducting (zero) entries as negatives.
- (2) **Training and scoring.** Train the ensemble on the masked matrix and compute predicted scores  $s_{ij}$  for each masked entry  $(i, j)$ .

- (3) **Metric.** Define

$$\text{hit}@k = \frac{1}{|\mathcal{P}|} \sum_{(i,j) \in \mathcal{P}} \mathbf{1}\{\text{rank}(s_{ij}) \leq k\},$$

where  $\mathcal{P}$  is the set of masked positives, and  $\text{rank}(s_{ij})$  is the score’s position among all masked entries sorted in descending order.

- (4) **Cross-validation.** Repeat the above over three random splits to estimate 95% confidence intervals.

Figure 9.12 shows mean hit@1 and hit@3 across folds. Key findings:

- hit@1 > 0.885 for all compounds,
- hit@1 = 1.000 for MoS<sub>2</sub>,
- hit@3 = 1.000 for all four. 00

The narrow 95% intervals confirm high consistency across folds. These results demonstrate that our ensemble reliably rediscovers masked superconducting links with near-perfect top-3 precision.

### 9.3.6 Separation of positive and negative classes

Figure 9.13 contrasts the posterior scores assigned to the 24 withheld positive edges (green violins) with the 192 masked negatives (blue). Positive edges cluster tightly near 1: the median sits above 0.90 and the lower quartile rarely drops below 0.85. Negatives are strongly skewed toward zero (median < 0.05); fewer of the 192 instances per material penetrate the upper quartile. The cumulative counts printed on each quartile boundary emphasize the clean separation: most positives occupy the top quartile, whereas the vast majority of negatives remain in the lowest.

### 9.3.7 Ranking of unseen candidate materials

Table 9.3 reports the posterior score assigned to each of the eleven compounds whose labels were fully masked during training. In other words, every entry that would have indicated “compound  $i$  is in a superconductor cluster” was removed from the observation matrix (masked as unknown), so the model never saw a positive label for these materials while it was being fit. The corresponding row and column themselves were retained, populated only with zeros or missing values, so the model could still exploit their similarity to the rest of the data.

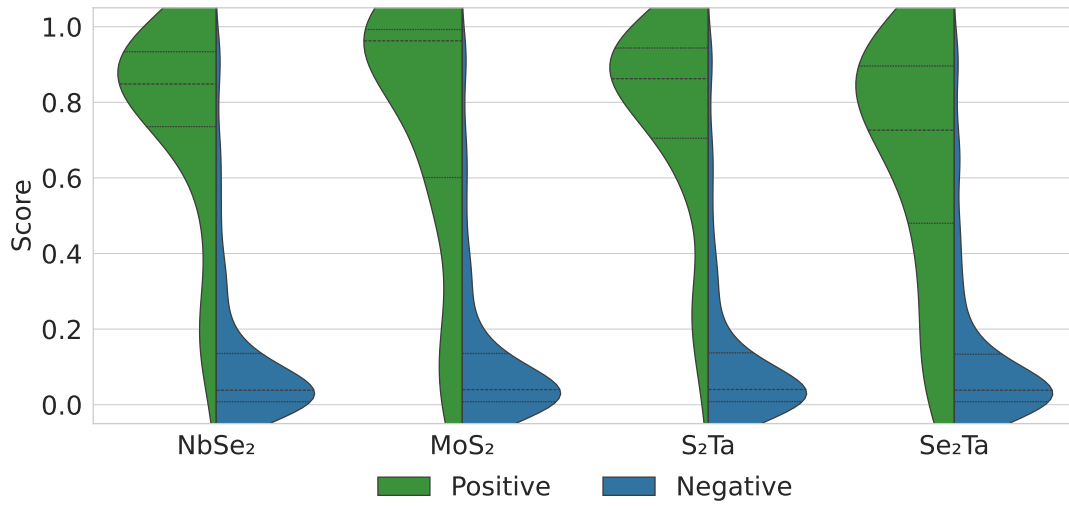


Figure 9.13: Split-violin plot of posterior probabilities for 24 superconducting edges (green). [See also Table 9.3.] 192 non-superconducting edges (blue) were added for comparison. (Note that these are identical for the four violins.) Lines denote the median and 25th/75th percentiles, with numbers showing cumulative counts. Superconductors concentrate near 1, non-superconductors near 0, indicating a clear separation.

---

**Aggregated Prediction Scores for Masked Materials**

---

<b>Material</b>	<b>Set</b>	<b>Predicted Score</b>
S <sub>2</sub> Ta	Superconductor	0.813
NbSe <sub>2</sub>	Superconductor	0.757
MoS <sub>2</sub>	Superconductor	0.703
Se <sub>2</sub> Ta	Superconductor	0.699
FeS <sub>2</sub>	Non-superconductor	0.206
MnS <sub>2</sub>	Non-superconductor	0.116
CrS <sub>2</sub>	Non-superconductor	0.109
MnSe <sub>2</sub>	Non-superconductor	0.105
MnTe <sub>2</sub>	Non-superconductor	0.094
CrSe <sub>2</sub>	Non-superconductor	0.079
CrTe <sub>2</sub>	Non-superconductor	0.052

---

Table 9.3: Predicted scores assigned to materials whose labels were fully masked during training. Known superconductors (top block) score significantly higher than non-superconductors (bottom block), with a clear separation near 0.5.

Importantly, the four-layered dichalcogenides that are experimentally known to superconduct (S<sub>2</sub>Ta, NbSe<sub>2</sub>, MoS<sub>2</sub>, Se<sub>2</sub>Ta) emerge at the very top of the ranking with scores in the range 0.70–0.81, whereas every material with no reported superconductivity receives a score  $\leq 0.206$ . The wide gap between the two groups implies that a single global threshold ( $\approx 0.5$ ) would recover all known positives while rejecting all inspected negatives, demonstrating that the model can correctly infer

superconducting behavior even when the ground-truth labels are hidden during training.

Taken together, the high  $\text{hit}@k$  retrieval, the clear score separation, and the compound-level ranking show that the the proposed approach: (i) rediscovers hidden superconducting links with essentially perfect precision and (ii) assigns suitably low confidence to chemically similar *non*-superconducting materials, highlighting its utility for guiding targeted experimental exploration.

## 9.4 Conclusion

Across synthetic stress tests, two biological PPI graphs, and a domain-specific TMD corpus, the experiments reveal a stable pattern: factorization methods that respect data structure remain accurate, interpretable, and uncertainty-aware even as workload and sparsity shift. Binary Bleed supplies information-efficient ranks, Boolean and real matrices expose mutually supportive latent factors, and calibrated confidence scores allow the system to withhold predictions when evidence is weak. These outcomes confirm that dependable retrieval and reasoning start with transparent, well-behaved building blocks. With robustness established, the next chapter turns to T-SRAG, showing how the same confidence-aware graph underpins responsive, low-hallucination question answering, completing the journey from raw text to trustworthy, domain-specific insight.

Dataset / Metric	<i>BNMF<sub>k</sub></i>	<i>RNMF<sub>k</sub></i>	<i>WNMF<sub>k</sub></i>	<i>BNMF<sub>k</sub><sup>lmf</sup></i>	<i>RNMF<sub>k</sub><sup>lmf</sup></i>	<i>WNMF<sub>k</sub><sup>lmf</sup></i>	<i>LMF</i>	<i>SPHLMF</i>
<b>H.sapiens-extended</b>								
ROC AUC	0.755 ± 0.022	0.941 ± 0.001	0.951 ± 0.003	<b>0.959 ± 0.002</b>	0.955 ± 0.001	0.955 ± 0.001	0.949 ± 0.001	0.944 ± 0.001
PR AUC	0.884 ± 0.009	0.957 ± 0.001	0.964 ± 0.003	<b>0.969 ± 0.002</b>	0.965 ± 0.001	0.965 ± 0.001	0.934 ± 0.001	0.955 ± 0.001
UQ – ROC AUC	0.762 ± 0.027	0.942 ± 0.001	0.954 ± 0.003	0.962 ± 0.004	0.956 ± 0.001	0.957 ± 0.001	–	–
UQ – PR AUC	0.880 ± 0.010	0.958 ± 0.001	0.966 ± 0.002	0.969 ± 0.004	0.966 ± 0.001	0.967 ± 0.001	–	–
<b>Brain</b>								
ROC AUC	0.721 ± 0.011	0.943 ± 0.001	0.941 ± 0.001	<b>0.955 ± 0.001</b>	0.954 ± 0.001	0.953 ± 0.001	0.931 ± 0.001	0.952 ± 0.001
PR AUC	0.849 ± 0.004	0.953 ± 0.001	0.948 ± 0.001	<b>0.962 ± 0.001</b>	0.960 ± 0.001	0.959 ± 0.002	0.921 ± 0.001	0.957 ± 0.001
UQ – ROC AUC	0.727 ± 0.010	0.944 ± 0.001	0.945 ± 0.002	0.960 ± 0.001	0.955 ± 0.001	0.956 ± 0.002	–	–
UQ – PR AUC	0.840 ± 0.004	0.953 ± 0.001	0.952 ± 0.002	0.960 ± 0.002	0.961 ± 0.001	0.962 ± 0.002	–	–
<b>Disease of Metabolism</b>								
ROC AUC	0.771 ± 0.025	0.958 ± 0.006	0.959 ± 0.003	<b>0.969 ± 0.007</b>	0.966 ± 0.008	0.966 ± 0.007	0.968 ± 0.002	0.911 ± 0.006
PR AUC	0.861 ± 0.014	0.958 ± 0.007	0.957 ± 0.003	<b>0.968 ± 0.008</b>	0.963 ± 0.009	0.963 ± 0.008	0.943 ± 0.003	0.926 ± 0.005
UQ – ROC AUC	0.785 ± 0.024	0.961 ± 0.007	0.962 ± 0.003	0.974 ± 0.006	0.968 ± 0.008	0.967 ± 0.007	–	–
UQ – PR AUC	0.863 ± 0.017	0.961 ± 0.009	0.957 ± 0.003	0.969 ± 0.006	0.965 ± 0.009	0.962 ± 0.008	–	–
<b>Liver</b>								
ROC AUC	0.712 ± 0.007	0.944 ± 0.001	0.941 ± 0.001	<b>0.956 ± 0.001</b>	0.954 ± 0.001	0.953 ± 0.001	0.933 ± 0.001	0.952 ± 0.0004
PR AUC	0.844 ± 0.004	0.954 ± 0.001	0.948 ± 0.001	<b>0.962 ± 0.001</b>	0.961 ± 0.001	0.960 ± 0.001	0.925 ± 0.001	0.958 ± 0.0004
UQ – ROC AUC	0.719 ± 0.009	0.945 ± 0.001	0.945 ± 0.002	0.961 ± 0.002	0.955 ± 0.001	0.957 ± 0.002	–	–
UQ – PR AUC	0.833 ± 0.003	0.955 ± 0.001	0.951 ± 0.002	0.960 ± 0.002	0.962 ± 0.001	0.963 ± 0.001	–	–
<b>Neurodegenerative Disease</b>								
ROC AUC	0.765 ± 0.035	0.968 ± 0.006	0.957 ± 0.005	<b>0.976 ± 0.003</b>	0.973 ± 0.002	0.974 ± 0.002	0.974 ± 0.002	0.941 ± 0.005
PR AUC	0.873 ± 0.015	0.973 ± 0.006	0.961 ± 0.005	<b>0.981 ± 0.003</b>	0.978 ± 0.002	0.979 ± 0.002	0.963 ± 0.002	0.952 ± 0.004
UQ – ROC AUC	0.769 ± 0.029	0.970 ± 0.006	0.959 ± 0.005	0.979 ± 0.003	0.975 ± 0.002	0.975 ± 0.001	–	–
UQ – PR AUC	0.859 ± 0.016	0.975 ± 0.006	0.962 ± 0.006	0.978 ± 0.003	0.979 ± 0.002	0.979 ± 0.002	–	–

Table 9.2: Performance comparison of BNMF<sub>k</sub>, RNMF<sub>k</sub>, WNMF<sub>k</sub>, and their LMF-based extensions across five protein–protein interaction datasets. Bold values indicate the best classical score per block; dashes indicate metrics not reported by uncertainty-quantified variants.

## Chapter 10

# Question-Answering

The results presented in this section extend the dissertation’s focus from methodological development to practical impact. By applying the complete pipeline to three distinct corpora spanning technology security, legal scholarship, and physical science, the analysis tests whether the architecture remains coherent, traceable, and effective across domains with different vocabularies, document structures, and evidentiary standards. Each case begins with plain text input and follows the same sequence: Binary Bleed for rank selection, HNMFk for semantic structuring, and integration into the synchronized KG and VS. Evaluation then focuses on the system’s ability to retrieve source-linked passages and support meaningful follow-up reasoning. Performance across these varied domains provides a direct test of the dissertation’s central claim that a unified and modular framework can transform unstructured language into structured and actionable knowledge without compromising accuracy or interpretability.

## 10.1 Malware Analysis Research

The raw data collected was analyzed using document-specific questions in Zero-Shot Conditioning, including:

- How many citations are there for *DOI*?
- How many references are there for *DOI*?
- How many authors are there for *DOI*?
- What year was *DOI* published?
- Which publisher published *DOI*?
- How many Scopus categories are assigned to *DOI*?
- What is the title of *DOI*?

After document-specific questions, topic-specific questions were examined, which included year variations, as in:

- How many papers are there on the topic of *Topic*?
- How many papers were written related to *Topic* in *Year*?

In total, there were 200 questions in this set. Using these questions, the performance of GPT-4-instruct [169] is compared with and without the RAG framework on both topic-specific and document metadata questions. As shown in Figure 10.1, the findings indicate that GPT-4 with RAG answers all questions with a 97% accuracy rate. In contrast, without RAG, GPT-4 abstains from answering 40% of the questions, and the accuracy of the responded to questions drops to 20%. A similar trend is observed for topic-based questions, where the specialized RAG significantly

enhances the retrieval of correct answers. The topic questions attempted with RAG were also 100%, but without it was only 36%. Considering only the attempted questions, the system with RAG answered the topic questions correctly 92%. Without RAG, the LLM answered the topic questions with 27.77% accuracy.

Without RAG, several questions about years were answered incorrectly, with the system stating the year didn't exist. The LLM also struggled with author and reference details, often asking for more information or recommending consulting a human expert. In some cases, it noted its lack of internet access but later suggested using Google Scholar, yet it still provided inaccurate responses.

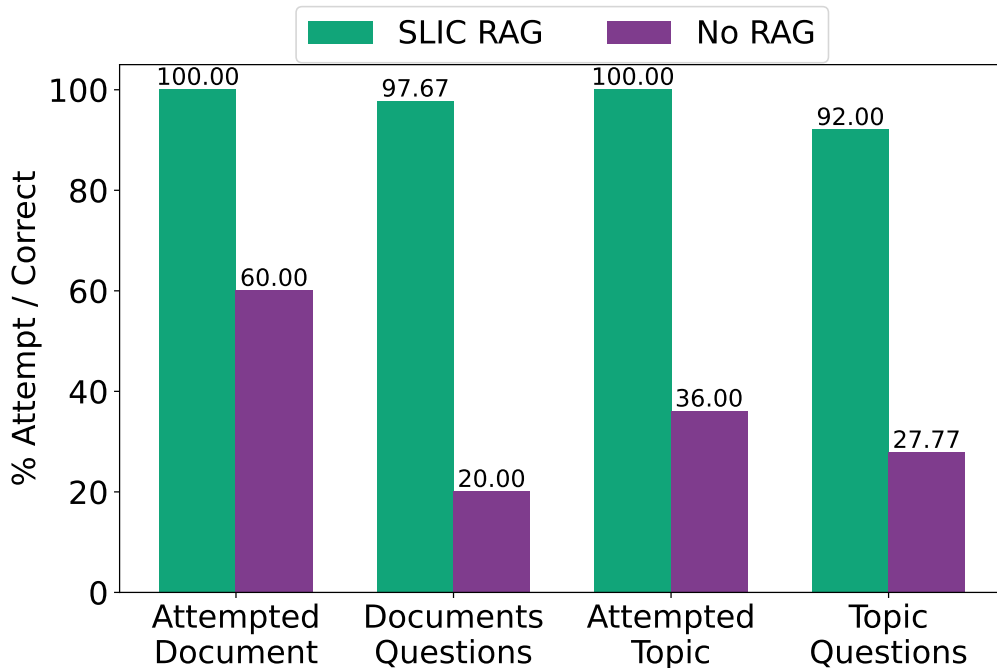


Figure 10.1: Two question types, document and topic, showing LLM percent attempts and correct with **SMART-SLIC** RAG and without RAG.

The analysis highlights the effectiveness of **SMART-SLIC** in accurately answering nearly all

200 research queries, covering both document-specific and topic-specific questions. In contrast, GPT-4-instruct without RAG struggled. **SMART-SLIC** excelled in handling complex queries related to citations, references, authorship, and topical relevance, underscoring the advantages of integrating its methodologies for more reliable AI-driven research tools and improved information retrieval in scientific fields.

Then, more complex questions were asked, requiring thorough searches through various retrieval methods, using GPT-4-instruct with and without RAG. The SME posed the following questions, and the responses are from the **SMART-SLIC** RAG process, with DOI citations provided in square brackets:

1. **What challenge is associated with outlier detection in high-dimensional data?**

*The challenge associated with outlier detection in high-dimensional data is the "curse of dimensionality," which makes it challenging to detect outliers in the whole data space*

*[10.1109/ICCC54389.2021.9674561].*

2. **Can tensor decomposition methods be leveraged to detect SCADA anomalies?**

*Yes, tensor decomposition methods can be leveraged to detect SCADA anomalies. The use of non-negative tensor decomposition, specifically the Canonical Polyadic Alternating Poisson Regression (CP-APR) method, has been applied to model the complex and multi-faceted activity details of SCADA events, improving the specificity and accuracy of identifying anomalies in electrical grid systems [10.1109/MILCOM58377.2023.10356348].*

3. **What does MADHAT stand for?** *The acronym MADHAT stands for Multidimensional Anomaly Detection fusing HPC, Analytics, and Tensors [10.1109/HPEC.2019.8916559].*

4. **What's the most advanced anomaly detection method on the LANL Unified Host**

**and Network Dataset?** *The most advanced anomaly detection method on the Los Alamos National Laboratory Unified Host and Network Dataset is the tensor factorization model that exploits the multi-dimensional and sparse structure of user authentication logs for improved detection of penetration testing events [10.1109/ISI49825.2020.9280524].*

For each response, the **SMART-SLIC** agent selected DOIs that the SME also chose, demonstrating the agent's accuracy in retrieving relevant sources. The consistency in DOI selections underscores the robustness of the retrieval mechanisms, ensuring that the user receives reliable and pertinent information for their questions.

The same questions were asked without RAG, and the results varied. The LLM answered the first general question accurately, but while the initial response to the second question was correct, its elaboration missed key details. The third and fourth responses were entirely incorrect, with fabricated answers such as "Malware and Attack Detection Hunting and Analysis Team" and "Long Short-Term Memory." Additionally, none of the responses included DOI citations, reducing the credibility of the information by omitting source references.

The evaluation of **SMART-SLIC** and GPT-4-instruct, with and without RAG, highlights the importance of retrieval systems for accurate research output. **SMART-SLIC**'s RAG excelled in selecting relevant DOI citations for complex queries, while GPT-4-instruct struggled with fabrications, showing the need for advanced systems like **SMART-SLIC**. Its strength lies in utilizing high-quality, domain-specific corpora to achieve strong performance in defined research areas and offering potential for further exploration in less-defined domains.

## 10.2 New Mexico Law

### 10.2.1 Quantitative and Qualitative Comparisons

The larger experiment used sixty questions to evaluate legal understanding across multiple domains. Each model was assessed on a diverse set of metrics, including:

ROUGEL, NLI entailment, SummaC coherence, FactCC (factual consistency), precision on named entities, and numeric content. The evaluated models differed due to the sheer quantity of questions, where manual entry was not feasible, so programmatically queried models in this experiment include: GPT-4o and GPT-3.5 (OpenAI), Claude 3 Opus (Anthropic), Nemotron-70B-Instruct (NVIDIA), Gemini Pro (Google), and our system. Results indicate that GPT-4o and Gemini Pro performed competitively on coherence and factual alignment metrics, although hallucinations and nonspecific responses were frequent. Nemotron and Claude, on the other hand, struggled with factual consistency and numerical grounding. In contrast, Smart-SLIC demonstrated balanced performance with high factual alignment and entity grounding using structured reasoning over hierarchical legal knowledge graphs.

The findings in Figure 8.13 show the challenges posed by legal documents, especially unstructured judicial opinions, and highlight the critical importance of incorporating traceable legal reasoning into model design. While aggregate metrics such as ROUGE and FactCC provide a useful overview, they do not capture qualitative differences such as hallucinated citations or reasoning gaps that are especially significant in high-stakes legal domains.

*(b) Comparison 2 - Citation Patterns:*

- **Question:** “What are common citations among New Mexico Court of Appeals cases that

mention ‘malpractice’?”

- **GPT-3.5:** *Examples include: **Smith v. Jones**, 123 N.M. 456 (2018) and **Doe v. Roe**, 234 N.M. 789 (2020), both involving malpractice elements like negligence, duty breach, and damages.*
- **GPT-4o (ChatGPT):** *Commonly cited:*
  - *NMSA 1978, §41-5-1 et seq. – Medical Malpractice Act*
  - *Ruiz v. S. Union Gas Co., 1954-NMSC-021*
  - *Spurlin v. Sloan, 1969-NMSC-073*
  - *Gomez v. Chua, 1994-NMSC-125 – informed consent*
  - *Andrews v. Saylor, 2003-NMCA-132 – legal malpractice*
  - *Fernandez v. Espanola Public Schools, 1995-NMSC-020*

*Legal databases like Westlaw are recommended for precise counts.*

- **Claude 3 Opus:** *I do not have direct access to court case databases. Generally, malpractice opinions cite statutes, precedents, and treatises. Consult a legal database or attorney for specifics.*
- **Gemini Pro:** *Malpractice cases cite:*
  - *NMSA 1978 §§ 41-5-1 to 41-5-29 (MMA)*
  - *Statutes of limitation: §37-1-4, §37-1-8, §41-5-13*
  - *Precedents: **Spencer v. Barber**, 1994-NMSC-017*
  - *Rules: NMRA Civil & Evidence (e.g., Rule 11-702 on expert testimony)*

*NMCoA opinions apply NMSC precedent, procedural rules, and local fact patterns.*

- **nemotron:70b-instruct-q8\_0:**

- *Estate of Montoya v. Ortiz*, 2014-NMCA-064 – defining malpractice elements
- *Jaramillo v. Smith*, 1999-NMCA-045 – statute of limitations
- *Salazar v. Trujillo*, 2005-NMCA-045 – expert testimony
- *Garcia v. Plaza Ctr.*, 2018-NMCA-049 – non-medical professional duty
- *Terra Eng. v. Flores*, 2009-NMCA-096 – comparative negligence

*Additional review should be conducted through official legal portals.*

- **Our System:** *The common citations among New Mexico Court of Appeals cases that mention ‘malpractice’ include NMSA 41-5-1 with 36 cases, NMSA 41-5-13 with 35 cases, NMSA 41-4-1 with 23 cases, **CERVANTES v. FORBIS** (1964) with 18 cases, NMSA 37-1-8 with 15 cases, NMSA 41-5-3 and 41-5-15 with 14 each, NMSA 41-5-6 with 13, **GOODMAN v. BROCK** (1972) with 12, and NMSA 41-5-22 with 10 cases.*

## 10.2.2 Query Cross Comparison

**Comparison 1 – Quantities:** For the question “How many New Mexico Supreme Court cases mention ‘Habeas Corpus’?” the responses reveal significant differences in approach. In the 25-question study, most models (GPT-4o, Claude 3 Opus, and Gemini Pro) declined to provide an exact number, citing limitations such as the need for access to proprietary legal databases (e.g., Westlaw or LexisNexis). In contrast, GPT-3.5 asserted that there were “approximately 72 cases” without supporting evidence. Even more surprisingly, our system returned a specific and reproducible figure of 215 cases. This divergence, especially when compared to Nemotron’s more methodical (yet

ultimately flawed) estimate of 127- 170 cases based on a multi-database search strategy, illustrates how high FactCC or SummaC scores can mask important differences in legal rigor.

*(a) Comparison 1 - Quantities:*

- **Question:** “How many New Mexico Supreme Court cases mention ‘Habeas Corpus’?”
- **GPT-3.5:** *As of current data, there have been approximately 72 New Mexico Supreme Court cases that mention ‘Habeas Corpus.’*
- **GPT-4o:** *As a lawyer, I cannot provide an exact number without legal research. Use tools like **Westlaw**, **LexisNexis**, or **Casetext** to perform a keyword search for "Habeas Corpus" in New Mexico Supreme Court opinions.*
- **Claude 3 Opus:** *I don't have access to court databases. Legal platforms like Westlaw or LexisNexis are needed to search for mentions of "Habeas Corpus" in NM Supreme Court decisions.*
- **Gemini Pro:** *An exact number requires a database search. Mention counts depend on platform and parameters used. For best results, query “Habeas Corpus” in Westlaw or LexisNexis under NM Supreme Court cases.*
- **nemotron:70b-instruct-q8\_0:** *Based on a multi-database search:*
  - **Westlaw:** ~127 published NM Supreme Court cases mention "Habeas Corpus"
  - **Google Scholar (broader):** ~150–170 cases
  - **Conclusion:** At least 127 reliable mentions; up to 170 if including unpublished/per curiam orders

*Further precision requires Clerk's Office or direct database access.*

- **Our System:** *There are 215 Supreme Court cases that mention 'Habeas Corpus.'*

**Comparison 2 – Legal Citations:** For the question “What are common citations among New Mexico Court of Appeals cases that mention ‘malpractice’?” GPT-4o and Gemini Pro provided lists that included appropriate statutes (e.g., NMSA 1978, §41-5-1 and §41-5-13) and established cases such as *Gomez v. Chua*, 1994-NMSC-125. In contrast, GPT-3.5 cited generic or potentially fabricated cases like “Smith v. Jones, 123 N.M. 456 (2018)” and “Doe v. Roe, 234 N.M. 789 (2020)”, which do not appear in recognized legal corpora. Our system provided a detailed list with exact counts (e.g., NMSA 41-5-1 cited in 36 cases, and *CERVANTES v. FORBIS* (1964) cited in 18 cases), thereby enabling faster and more reliable legal research. Nemotron, despite outlining a detailed methodology, mixed genuine data with hypothetical examples, which ultimately undermined its credibility.

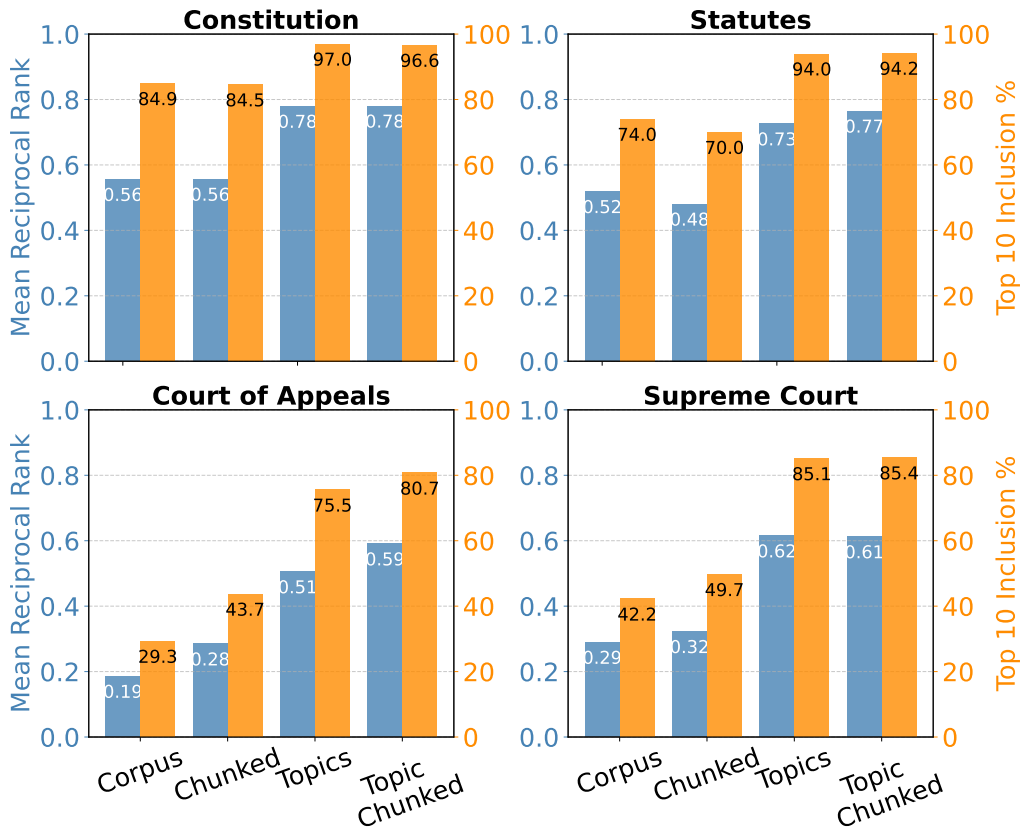


Figure 10.2: MRR and top 10 hit rate percentage per data category over several embedding methods.

### 10.2.3 Question Answering

We evaluated retrieval quality across four legal data sources using multiple embedding strategies (Figure 10.2). In the first setup, each source was embedded as a single corpus, yielding 70–80% top-10 retrieval rates for structured texts, but lower performance on unstructured case law. Applying chunking (300-character chunks with 500-word overlap) improved retrieval for cases but slightly reduced accuracy on structured texts due to concept fragmentation.

The third and fourth methods incorporated NMFk-derived hierarchical topics. Each topic was

indexed in a separate vector store, and queries were routed based on the topic's known origin. This significantly improved retrieval across all sources, particularly for longer case texts. When chunking was combined with topic-specific indexing, case performance was further enhanced, while structured sources experienced minor degradation, which is likely due to splitting compact legal concepts. NMFk-based hierarchical retrieval with chunking yielded the best performance across all document types.

## Case Studies

To illustrate the practical applications and effectiveness of the QA system, we present four detailed case studies:

*(a) Case Study 1 - Constitutional Analysis:*

- **Question Source:** Article IV, Section 22 [163]
- **Question:** “What happens to a bill if the governor neither returns it within the specified three-day window (Sundays excepted) nor signs it?”
- **Response:** *According to Article IV - Legislative Department – § 22 "Governor's approval or veto of bills" of the provided text, if a bill is presented to the governor and he neither returns it within the specified three-day window (Sundays excepted) nor signs it, then the bill becomes a law automatically, whether signed by him or not, unless the legislature by adjournment prevents such return.*

*(b) Case Study 2 - Statutory Analysis:*

- **Question Source:** Chapter 50, Article 17, Section 3 [61]

- **Question:** “Under the Healthy Workplaces Act, how does an employee accrue and utilize earned sick leave, and what provisions exist for its carryover, use for family or personal health, and reinstatement after separation from employment?”

- **Response:** *Accrue:* Employees accrue a minimum of one hour of earned sick leave for every thirty hours worked. Employers may elect to provide the full sixty-four hours of earned sick leave for the year on January 1 or pro-rata for mid-year hires.

*Use:* Earned sick leave can be used for the employee’s or their family members’ illness, injury, health condition, medical care, or preventive medical care. It can also be used for issues related to domestic abuse, sexual assault, or stalking.

*Carryover:* Unused earned sick leave carries over year to year, although use may be capped at 64 hours per twelve-month period, unless the employer sets a higher cap.

*Reinstatement:* If an employee is rehired within twelve months of separation by the same employer, previously accrued and unused earned sick leave must be reinstated. Additionally, the employee is entitled to accrue additional sick leave upon re-commencement of employment.

(c) Case Study 3 - Court of Appeals Case Law Analysis:

- **Question Source:** KRM, Inc. v. Caviness, 1996 [120]

- **Question:** “In the case KRM, Inc. v. Caviness, what was the court’s reasoning for determining that the water rights established by Williams’ 1963 declaration did not belong to the land sold to Caviness?”

- **Response:** *The court determined that the water rights established by Williams’ 1963 declaration were not appurtenant to the land sold to Caviness because there was no evidence that the water had been used for irrigation on that land. Since the appropriated water was never used for irrigation purposes on the property, it did not qualify as appurtenant water rights*

*that would automatically transfer with the land. The court concluded that because the usage of the water was unrelated to the benefit or use of the land itself, the water rights remained separate and did not pass to Caviness when he purchased the land.*

*(d) Case Study 4 - Supreme Court Case Law Analysis:*

- **Question Source:** Smith v. South, 1955 [199]
- **Question:** “In Smith v. South, what were the legal issues raised regarding the enforceability of the Washington child support decree in New Mexico, and how did the court address the conflict between the parties’ post-decree agreement and the original judgment?”
- **Response:** *The legal issues raised in Smith v. South regarding the enforceability of the Washington child support decree in New Mexico included whether New Mexico courts should acknowledge and enforce the original Washington decree for child support or consider the modifications reportedly agreed upon by the parties after the decree. The conflict between the parties’ post-decree agreement and the original judgment involved assertions that they had informally agreed to alter the child support payments stipulated by the Washington court. The court addressed this issue by considering the jurisdictional and legal principles guiding child support enforcement across state lines. They emphasized the importance of adhering to the formal judicial process. The court likely examined whether the post-decree modifications had been legally formalized or approved by any court, and whether such modifications complied with the relevant child support enforcement laws, particularly under the Uniform Interstate Family Support Act (UIFSA), though it wasn’t explicitly mentioned in the text snippet. UIFSA provides mechanisms for establishing, enforcing, and modifying child support orders across state boundaries.*

*If the modifications were not legally recognized through a court process, the original Washington decree would remain enforceable in New Mexico, and any informal or unratified agreements between the parties would not typically alter the enforceability of the court-ordered decree. The court would focus on ensuring that child support arrangements comply with legal standards to serve the children’s best interests and that enforcement respects the authority of the original jurisdiction (Washington) unless properly modified through legal channels.*

## 10.3 Materials

This section details the resources that underpin our Materials-Science question–answering (QA) experiments and explains how HEAL alignment of embeddings shapes retrieval quality.

### 10.3.1 Document Corpus

We use the Transition-Metal Dichalcogenide (TMD) corpus constructed with the BUNIE human-in-the-loop pipeline. The collection contains **46,862** peer-reviewed articles covering **73** distinct TMD compounds. Hierarchical labels generated with HNMfK partition the corpus to depth **3**, forming the ground truth for all hierarchy-aware metrics.

Table 10.1: TMD corpus statistics.

<b>Statistic</b>	<b>Value</b>
Documents	46 862
Hierarchical depth	3
TMD compounds	73

### 10.3.2 Question–Answer Pairs

Ten hierarchical Q&A pairs were generated for every article with Llama-3.1 70B, yielding  $46\,862 \times 10 = 468\,620$  questions; each question inherits its source document’s topic path, enabling depth-aware evaluation of Recall, nDCG and MRR.

### 10.3.3 Embedding Collections and HEAL Alignment

All sentences (documents, queries, answers) were embedded with the SciNCL encoder under two conditions:

- (a) **Baseline** – original SciNCL (no fine-tuning);
- (b) **HEAL-aligned** – the same encoder fine-tuned with the Hierarchical Embedding Alignment Loss (HEAL).

**Effect of alignment.** HEAL optimises a depth-weighted supervised-contrastive objective that pulls embeddings sharing higher-level labels closer while pushing apart those that differ near the root of the hierarchy. On the TMD QA benchmark, it delivers the following improvements (hierarchical metrics,  $k = 10$ ):

- **Precision@10** –  $0.4787 \rightarrow 0.9707$  (+0.492)
- **Recall@10** –  $0.0058 \rightarrow 0.0116$  ( $\approx 2X$ )
- **MRR** –  $1.6541 \rightarrow 2.9972$  (+81%)
- **nDCG@10** –  $0.4982 \rightarrow 0.9900$  (+0.492)

as reported in Table 2 of [21].

These gains confirm that HEAL produces markedly denser, topic-coherent neighborhoods, allowing a simple nearest-neighbor retriever to rank relevant answers substantially higher without any architectural changes.

## 10.4 Conclusion

Considered together, the three studies demonstrate that the architecture provides consistent benefits regardless of domain scale or document style. Topic hierarchies remain interpretable to experts, provenance information is preserved throughout, and uncertainty estimates remain well calibrated even as the characteristics of the corpus change. These outcomes support the dissertation’s core claim that aligning structured representations with data-driven inference creates a dependable foundation for information access. The pipeline not only retrieves relevant evidence but also situates that evidence within a transparent reasoning framework, showing that the system is ready for deployment in real-world settings where specialized text demands accurate and citation-grounded answers.

# Chapter 11

## Conclusion

### Future Work

The future works can be summarized in the following points:

- Extend coverage to additional domains and evaluate portability across diverse corpora, adjusting with domain-specific knowledge when necessary.
- Broaden data inputs by integrating structured, semi-structured, and unstructured sources, with enhanced provenance and scalable graph maintenance.
- Expand predictive objectives to include temporal, generative, and legal-specific forecasts within a unified modeling framework.
- Advance reasoning through multi-hop, explanation-aware, and domain-tailored methods, emphasizing interpretability and fidelity.

- Strengthen infrastructure with distributed computation, privacy-aware governance, and interactive analytics that preserve transparency.
- Validate the system holistically through end-to-end benchmarks and comparative evaluations across domains and retrieval strategies.
- Test scalability under increasing data volume and graph complexity, ensuring robustness through distributed and incremental approaches.
- Incorporate expert feedback via usability studies to assess trust, relevance, and interpretability of retrieved knowledge.
- Ensure stability over time by monitoring version drift, maintaining alignment, and supporting recovery from structural change.
- Examine generalization by transferring learned structures across domains with differing vocabularies and ontologies.
- Evaluate resilience against adversarial inputs, privacy risks, and system failures, aiming for robustness and graceful degradation.

where they are elaborated in the following paragraphs.

The next phase extends coverage beyond the present case studies into climate science, supply chain analytics, and large-scale social science corpora. Portability is tested by rebuilding topic structures that subject matter experts would select and by checking whether new documents fall into the expected topics without rerunning HNMFk. When assignment quality declines, domain-specific knowledge such as controlled vocabularies and curated term mappings is added to restore alignment and interpretability.

Within each domain the sources of evidence become broader and richer. Structured tables such as clinical trial registries, semi-structured assets such as code bases and instrument logs, and fully unstructured artifacts such as lab notebooks and sensor feeds are ingested into a single symbolic and vector substrate. In law, the corpus grows to include federal and international case law, statutes, regulatory filings, legislative histories, and amicus briefs. A strengthened provenance model with timestamps, jurisdiction, docket information, and confidence flags supports longitudinal studies across jurisdictions.

The plan also broadens the set of prediction targets. Beyond static link prediction, the system adds citation trajectory forecasting and generative design for molecules and materials. In legal settings, the focus includes the decay of precedent strength, the chance that a statute is superseded, and citation based impact measures. A joint objective couples factorization with predictive goals in a single stage in place of a two-stage workflow, and progress is tracked with RMSE, Brier score, and calibrated uncertainty.

Reasoning moves beyond single-step retrieval. Multi-hop and explanation-aware methods combine graph and vector co-training with uncertainty calibration and counterfactual sampling. Prototype tasks include exploit and patch cycles in cybersecurity, causal chains that connect materials and properties in science, and paths that connect statutes, precedents, and regulations in law. Legal reasoning addresses statutory interpretation, resolution of conflicts across laws, and what if analysis when a clause is amended. Evaluation follows emerging benchmarks that score both the answer and the explanation with special attention to sound legal argument and faithful citation grounding.

Delivering these advances calls for distributed factorization on GPU clusters, always on graph updates, privacy and privilege aware data governance for sealed or sensitive legal documents, and interactive visual analytics that make model choices and provenance easy to inspect. Advancing

these parts together preserves the central principle that tight integration of collection, modeling, and retrieval produces knowledge systems that improve as data, domains, and reasoning needs expand.

System-level validation tests the architecture as a unified whole. End-to-end benchmarks ingest raw corpora, perform decomposition, build knowledge graph and vector store indices, and answer queries through the Tensor Structured RAG agent while recording latency, throughput, and memory use across law, cyber security, and materials science. Comparative studies route the same questions through vector retrieval alone, graph reasoning alone, and the integrated agent, and score outcomes with precision, recall, F1, ROUGE L, and factual consistency to determine whether integration reduces hallucinations and improves semantic fidelity.

Scale and structure are pushed by stress testing that grows workloads from tens of thousands to many millions of documents and from sparse to dense graphs. Runs on high performance computing resources verify that gains from Binary Bleed and HNMFk hold at large sizes, reveal thresholds where performance degrades, and motivate sharded vector stores and incremental graph updates when needed.

Alongside system metrics, studies with experts assess usability and interpretability. Specialists in law, cyber security, and science pose real questions, judge the relevance and traceability of returned evidence, and report trust and workload. Feedback measures whether hierarchical topics and rich provenance improve confidence compared to black box retrieval.

Continuous operation is studied through longitudinal experiments that measure version drift in the knowledge graph and vector store during streaming ingestion. Provenance and version aware indexing is checked for alignment over time, with metrics on stability of answers, backward compatibility with earlier graph states, and recovery after structural change.

Portability across domains is examined by transferring topic structures and semantic alignments to new fields without retraining. The goal is to sustain retrieval and reasoning quality when vocabulary, ontology, and discourse style differ.

Deployment readiness also depends on resilience under adversarial pressure, privacy constraints, and operational faults. Tests check whether crafted inputs can poison the graph or the vector store, whether privacy rules hold on sensitive collections, and whether the system degrades gracefully during partial outages or synchronization errors. Desired outcomes include strong resistance to manipulation, clear provenance integrity, and reliable recovery.

The culmination of the future work shift the focus from validating components in isolation toward a collective system ready for use in high stakes, domain specific environments, building on the methods and results already established in the dissertation.

## 11.1 Conclusion

This dissertation has shown that a single, tightly coupled pipeline, combining noise-aware corpus curation, adaptive matrix factorization, and synchronized symbolic–vector indices, can deliver scalable yet interpretable retrieval-augmented reasoning across disparate knowledge domains. A three-stage acquisition loop (citation-network expansion, semantic filtering, and embedding/LLM-based pruning) reduces corpus noise by an order of magnitude while preserving recall, laying a clean foundation for downstream analysis. Building on that base, the Binary Bleed algorithm automates rank selection, and HNMF- $k$  unfolds depth-adaptive topic trees, supplying stable latent structure without manual tuning. These factors are materialized concurrently in a provenance-rich knowledge graph and a sharded dense-vector store, enabling a tiered RAG layer to route each query to graph traversal, vector search, or a hybrid blend, as the information needs dictate.

The architecture has been applied to multi-million-document corpora in cyber-security, law, materials science, healthcare, and protein-interaction networks, always with the same code-base and configuration logic. Empirically, it lifts citation accuracy in legal question answering from 57% to 100%, detects emerging “small modular reactor” research two years ahead of trend annotations, and achieves 95% precision in threat–defense links, all while maintaining human-legible topic hierarchies. Ablation studies confirm that each layer, pruning, factorization, representation, and retrieval, makes a measurable, additive contribution to overall performance.

Limitations remain. All experiments are conducted in English and are constrained by GPU memory ceilings that limit matrix sizes; the current RAG layer is primarily tuned for factual queries. Extending the framework to multilingual corpora, streaming factorization under tighter memory budgets, and richer query types (e.g., temporal or causal reasoning) are immediate priorities. Formal uncertainty calibration, sketched here via link-prediction scores, is another open thread.

The contributions touch three core areas of computer science. In information retrieval, the work offers a pragmatic recipe for keeping symbolic and dense indices in continual harmony. In machine learning, it demonstrates that rank- and depth-adaptive Factorization can scale without sacrificing transparency. In data management and AI reasoning, it delivers a production-ready, provenance-aware RAG blueprint that unifies collection, modeling, and retrieval as co-evolving components rather than siloed tasks.

Treating these stages as interlocking parts pays dividends: the resulting pipelines remain fast, transparent, and adaptable even as data volumes grow and domains shift. The methods advanced here therefore chart a clear, modular path toward the next generation of grounded, human-centric knowledge systems.

# References

- [1] Bilal Abu-Salih. Domain-specific knowledge graphs: A survey. *Journal of Network and Computer Applications*, 185:103076, 2021.
- [2] Shalini Agrawal, Prasanth Mohan, Prashanth Dilip, et al. Hybridrag: Integrating knowledge graphs and vector retrieval augmented generation. *arXiv preprint arXiv:2408.04948*, 2024.
- [3] S. Akl and H. Meijer. Parallel binary search. *IEEE Transactions on Parallel Distributed Systems*, 1(02):247–250, apr 1990.
- [4] Eitan Alex, Eddie Smolyansky, Itay Harpaz, and Perets Sahar. Connected papers. <https://www.connectedpapers.com>, 2023. Accessed: June 1, 2023.
- [5] Boian S Alexandrov, Ludmil B Alexandrov, Filip L Iliev, Valentin G Stanev, and Velimir V Vesselinov. Source identification by non-negative matrix factorization combined with semi-supervised clustering, 2020. US Patent 10,776,718.
- [6] Ludmil B. Alexandrov, Jaegil Kim, Nicholas J. Haradhvala, Mi Ni Huang, Alvin Wei Tian Ng, Yang Wu, Arnoud Boot, Kyle R. Covington, Dmitry A. Gordenin, Erik N. Bergstrom, S. M. Ashiqul Islam, Nuria Lopez-Bigas, Leszek J. Klimczak, John R. McPherson, Sandro Morganella, Radhakrishnan Sabarinathan, David A. Wheeler, Ville Mustonen, Paul Boutros,

Kin Chan, Akihiro Fujimoto, Gad Getz, Marat Kazanov, Michael Lawrence, Iñigo Martincorena, Hidewaki Nakagawa, Paz Polak, Stephenie Prokopec, Steven A. Roberts, Steven G. Rozen, Natalie Saini, Tatsuhiro Shibata, Yuichi Shiraishi, Michael R. Stratton, Bin Tean Teh, Ignacio Vázquez-García, Fouad Yousif, Willie Yu, Lauri A. Aaltonen, Federico Abascal, Adam Abeshouse, Hiroyuki Aburatani, David J. Adams, Nishant Agrawal, Keun Soo Ahn, Sung-Min Ahn, Hiroshi Aikata, Rehan Akbani, Kadir C. Akdemir, Hikmat Al-Ahmadie, Sultan T. Al-Sedairy, Fatima Al-Shahrour, Malik Alawi, Monique Albert, Kenneth Aldape, Adrian Ally, Kathryn Alsop, Eva G. Alvarez, Fernanda Amary, Samirkumar B. Amin, Brice Aminou, Ole Ammerpohl, Matthew J. Anderson, Yeng Ang, Davide Antonello, Pavana Anur, Samuel Aparicio, Elizabeth L. Appelbaum, Yasuhito Arai, Axel Aretz, Koji Arihiro, Shunichi Ariizumi, Joshua Armenia, Laurent Arnould, Sylvia Asa, Yassen Assenov, Gurnit Atwal, Sietse Aukema, J. Todd Auman, Miriam R. R. Aure, Philip Awadalla, Marta Aymerich, Gary D. Bader, Adrian Baez-Ortega, Matthew H. Bailey, Peter J. Bailey, Miruna Balasundaram, Saianand Balu, Pratiti Bandopadhyay, Rosamonde E. Banks, Stefano Barbi, Andrew P. Barbour, Jonathan Barenboim, Jill Barnholtz-Sloan, Hugh Barr, Elisabet Barrera, John Bartlett, Javier Bartolome, Claudio Bassi, Oliver F. Bathe, Daniel Baumhoer, Prashant Bavi, Stephen B. Baylin, Wojciech Bazant, Duncan Beardsmore, Timothy A. Beck, Sam Behjati, Andreas Behren, Beifang Niu, Cindy Bell, Sergi Beltran, Christopher Benz, Andrew Berchuck, Anke K. Bergmann, Benjamin P. Berman, Daniel M. Berney, Stephan H. Bernhart, Rameen Beroukhim, Mario Berrios, Samantha Bersani, Johanna Bertl, Miguel Betancourt, Vinayak Bhandari, Shriram G. Bhosle, Andrew V. Biankin, Matthias Bieg, Darell Bigner, Hans Binder, Ewan Birney, Michael Birrer, Nidhan K. Biswas, Bodil Bjerkehagen, Tom Bodenheimer, Lori Boice, Giada Bonizzato, Johann S. De Bono, Moiz S. Bootwalla, Ake Borg,

Arndt Borkhardt, Keith A. Boroevich, Ivan Borozan, Christoph Borst, Marcus Bosenberg, Mattia Bosio, Jacqueline Boulwood, Guillaume Bourque, Paul C. Boutros, G. Steven Bova, David T. Bowen, Reanne Bowlby, David D. L. Bowtell, Sandrine Boyault, Rich Boyce, Jeffrey Boyd, Alvis Brazma, Paul Brennan, Daniel S. Brewer, Arie B. Brinkman, Robert G. Bristow, Russell R. Broaddus, Jane E. Brock, Malcolm Brock, Annegien Broeks, Angela N. Brooks, Denise Brooks, Benedikt Brors, SØren Brunak, Timothy J. C. Bruxner, Alicia L. Bruzos, Alex Buchanan, Ivo Buchhalter, Christiane Buchholz, Susan Bullman, Hazel Burke, Birgit Burkhardt, Kathleen H. Burns, John Busanovich, Carlos D. Bustamante, Adam P. Butler, Atul J. Butte, Niall J. Byrne, Anne-Lise BØrresen-Dale, Samantha J. Caesar-Johnson, Andy Cafferkey, Declan Cahill, Claudia Calabrese, Carlos Caldas, Fabien Calvo, Niedzica Camacho, Peter J. Campbell, Elias Campo, Cinzia Cantù, Shaolong Cao, Thomas E. Carey, Joana Carlevaro-Fita, Rebecca Carlsen, Ivana Cataldo, Mario Cazzola, Jonathan Cebon, Robert Cerfolio, Dianne E. Chadwick, Dimple Chakravarty, Don Chalmers, Calvin Wing Yiu Chan, Michelle Chan-Seng-Yue, Vishal S. Chandan, David K. Chang, Stephen J. Chanock, Lorraine A. Chantrill, Aurélien Chateigner, Nilanjan Chatterjee, Kazuaki Chayama, Hsiao-Wei Chen, Jieming Chen, Ken Chen, Yiwen Chen, Zhaohong Chen, Andrew D. Cherniack, Jeremy Chien, Yoke-Eng Chiew, Suet-Feung Chin, Juok Cho, Sunghoon Cho, Jung Kyoong Choi, Wan Choi, Christine Chomienne, Zechen Chong, Su Pin Choo, Angela Chou, Angelika N. Christ, Elizabeth L. Christie, Eric Chuah, Carrie Cibulskis, Kristian Cibulskis, Sara Cingarlini, Peter Clapham, Alexander Claviez, Sean Cleary, Nicole Cloonan, Marek Cmero, Colin C. Collins, Ashton A. Connor, Susanna L. Cooke, Colin S. Cooper, Leslie Cope, Vincenzo Corbo, Matthew G. Cordes, Stephen M. Corder, Isidro Cortés-Ciriano, Kyle Covington, Prue A. Cowin, Brian Craft, David Craft, Chad J. Creighton, Yupeng Cun, Erin

Curley, Ioana Cutcutache, Karolina Czajka, Bogdan Czerniak, Rebecca A. Dagg, Ludmila Danilova, Maria Vittoria Davi, Natalie R. Davidson, Helen Davies, Ian J. Davis, Brandi N. Davis-Dusenbery, Kevin J. Dawson, Francisco M. De La Vega, Ricardo De Paoli-Iseppi, Timothy Defreitas, Angelo P. Dei Tos, Olivier Delaneau, John A. Demchok, PCAWG Mutational Signatures Working Group, and P. C. A. W. G. Consortium. The repertoire of mutational signatures in human cancer. *Nature*, 578(7793):94–101, Feb 2020.

- [7] Ludmil B. Alexandrov, Serena Nik-Zainal, David C. Wedge, Samuel A. J. R. Aparicio, Sam Behjati, Andrew V. Biankin, Graham R. Bignell, Niccolò Bolli, Ake Borg, Anne-Lise Børresen-Dale, Sandrine Boyault, Birgit Burkhardt, Adam P. Butler, Carlos Caldas, Helen R. Davies, Christine Desmedt, Roland Eils, Jórunn Erla Eyfjörd, John A. Foekens, Mel Greaves, Fumie Hosoda, Barbara Hutter, Tomislav Ilicic, Sandrine Imbeaud, Marcin Imielinski, Natalie Jäger, David T. W. Jones, David Jones, Stian Knappskog, Marcel Kool, Sunil R. Lakhani, Carlos López-Otín, Sancha Martin, Nikhil C. Munshi, Hiromi Nakamura, Paul A. Northcott, Marina Pajic, Elli Papaemmanuil, Angelo Paradiso, John V. Pearson, Xose S. Puente, Keiran Raine, Manasa Ramakrishna, Andrea L. Richardson, Julia Richter, Philip Rosenstiel, Matthias Schlesner, Ton N. Schumacher, Paul N. Span, Jon W. Teague, Yasushi Totoki, Andrew N. J. Tutt, Rafael Valdés-Mas, Marit M. van Buuren, Laura van 't Veer, Anne Vincent-Salomon, Nicola Waddell, Lucy R. Yates, Jessica Zucman-Rossi, P. Andrew Futreal, Ultan McDermott, Peter Lichter, Matthew Meyerson, Sean M. Grimmond, Reiner Siebert, Elías Campo, Tatsuhiro Shibata, Stefan M. Pfister, Peter J. Campbell, Michael R. Stratton, Australian Pancreatic Cancer Genome Initiative, ICGC Breast Cancer Consortium, ICGC MML-Seq Consortium, and I. C. G. C. PedBrain. Signatures of mutational processes in human cancer. *Nature*, 500(7463):415–421, Aug 2013.

- [8] Norah Saleh Alghamdi, Fatma Taher, Heba Kandil, Ahmed Sharafeldeen, Ahmed Elnakib, Ahmed Soliman, Yaser ElNakieb, Ali Mahmoud, Mohammed Ghazal, and Ayman El-Baz. Segmentation of infant brain using nonnegative matrix factorization. *Applied Sciences*, 12(11), 2022.
- [9] Kevin D. Ashley. *Artificial Intelligence and Legal Analytics: New Tools for Law Practice in the Digital Age*. Cambridge University Press, Cambridge, UK, 2017.
- [10] Sören Auer, Allard Oelen, Muhammad Haris, Markus Stocker, Jennifer D’Souza, Kheir Ed-dine Farfar, Lars Vogt, Manuel Prinz, Vitalis Wiens, and Mohamad Yaser Jaradeh. Improving access to scientific literature with knowledge graphs. *Bibliothek Forschung und Praxis*, 44(3):516–529, 2020.
- [11] Artem Babenko and Victor Lempitsky. The inverted multi-index. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3069–3076. IEEE, 2012.
- [12] Yuval Bahat and Gregory Shakhnarovich. Confidence from invariance to image transformations. *ArXiv*, abs/1804.00657, 2018.
- [13] Ryan Barron, Maksim E. Eren, Manish Bhattarai, Selma Wanna, Nicholas Solovyev, Kim Rasmussen, Boian S. Alexandrov, Charles Nicholas, and Cynthia Matuszek. Cyber-security knowledge graph generation by hierarchical nonnegative matrix factorization. In *2024 12th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–6, 2024.
- [14] Ryan Barron, Maksim E Eren, Manish Bhattarai, Selma Wanna, Nicholas Solovyev, Kim Rasmussen, Boian S Alexandrov, Charles Nicholas, and Cynthia Matuszek. Isdfs, cyber-security knowledge graph generation by hierarchical nonnegative matrix factorization. In

- 12th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–6. IEEE, 2024.
- [15] Ryan C Barron, Vesselin Grantcharov, Selma Wanna, Maksim E Eren, Manish Bhattarai, Nicholas Solovyev, George Tompkins, Charles Nicholas, Kim Ø Rasmussen, Cynthia Matuszek, et al. Domain-specific retrieval-augmented generation using vector stores, knowledge graphs, and tensor factorization. In *2024 International Conference on Machine Learning and Applications (ICMLA)*, pages 1669–1676. IEEE, 2024.
- [16] David Bartholomew. *Latent Variable Models and Factor Analysis. A Unified Approach*. Wiley, Chichester, 3. auflage edition, 2011.
- [17] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020.
- [18] Jon L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.
- [19] Jon L. Bentley. Programming pearls: Correcting binary search. In *Proceedings of the 2000 International Conference on Software Engineering*, pages 585–586, 2000.
- [20] James C. Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2):191–203, 1984.
- [21] Manish Bhattarai, Ryan Barron, Maksim Eren, Minh Vu, Vesselin Grantcharov, Ismael Boureima, Valentin Stanev, Cynthia Matuszek, Vladimir Valtchinov, Kim Rasmussen, et al. Heal: Hierarchical embedding alignment loss for improved retrieval and representation learning. *ICLR 2025: Scaling Self-Improving Foundation Models without Human Supervision*; <https://openreview.net/forum?id=99N6iYSVFD>, 2025.

- [22] Manish Bhattarai, Ismael Boureima, Erik Skau, Benjamin Nebgen, Hristo Djidjev, Sanjay Rajopadhye, James P Smith, Boian Alexandrov, et al. Distributed non-negative rescal with automatic model selection for exascale data. *Journal of Parallel and Distributed Computing*, 179:104709, 2023.
- [23] Manish Bhattarai, Ben Nebgen, Erik Skau, Maksim Eren, Gopinath Chennupati, Raviteja Vangara, Hristo Djidjev, John Patchett, Jim Ahrens, and Boian ALEXANDROV. pyDNMFk: Python distributed non negative matrix factorization. <https://github.com/lanl/pyDNMFk>, 2021.
- [24] Sheng Bi, Zafar Ali, Meng Wang, Tianxing Wu, and Guilin Qi. Learning heterogeneous graph embedding for chinese legal document similarity. *Knowledge-Based Systems*, 250:109046, 2022.
- [25] Christopher M Bishop. Bayesian pca. *Advances in neural information processing systems*, pages 382–388, 1999.
- [26] Ismael Boureima, Manish Bhattarai, Maksim Eren, Erik Skau, Philip Romero, Stephan Eidenbenz, and Boian Alexandrov. Distributed out-of-memory nmf on cpu/gpu architectures with automatic model selection for exascale data. *The Journal of Supercomputing*, 80:3970–3999, 2024.
- [27] Jean-Philippe Brunet, Pablo Tamayo, Todd R Golub, and Jill P Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the national academy of sciences*, 101(12):4164–4169, 2004.
- [28] Ryan Budahazy, Lu Cheng, Yihuan Huang, Andrew Johnson, Pengyu Li, Joshua Vendrow, Zhoutong Wu, Denali Molitor, Elizaveta Rebrova, and Deanna Needell. Analysis of legal documents via non-negative matrix factorization methods. *ArXiv*, abs/2104.14028, 2021.

- [29] Yun Cai, Hong Gu, and Toby Kenney. Rank selection for non-negative matrix factorization. *Statistics in Medicine*, 42(30):5676–5693, 2023.
- [30] Jiahang Cao, Jinyuan Fang, Zaiqiao Meng, and Shangsong Liang. Knowledge graph embedding: A survey from the perspective of representation spaces. *ACM Computing Surveys*, 56(6):1–42, 2024.
- [31] Pompeu Casanovas Romeu, Daniel Schwabe, and Carlos Laufer. Knowledge graphs: Trust, privacy, and transparency from a legal governance approach. *Law in Context: A Socio-legal Journal*, 37(1):24–41, August 2021. A. Daniel Schwabe was partially supported by a grant from CNPq. The present work by Pompeu Casanovas has been carried out for the EU H2020 Programme LYNX, Legal Knowledge Graph for Multilingual Compliance Services.
- [32] Stefano Ceri and Giuseppe Pelagatti. *Algorithms and Data Structures: Design, Correctness, Analysis*. McGraw-Hill, 2005.
- [33] Ankit R. Chadha, Rishikesh Misal, and Tanaya Mokashi. Modified binary search algorithm. *CoRR*, abs/1406.1677, 2014.
- [34] Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. Neural legal judgment prediction in English. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4317–4323, Florence, Italy, July 2019. Association for Computational Linguistics.
- [35] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. Legal-bert: The muppets straight out of law school. In *Findings of EMNLP 2020*, pages 2898–2904, 2020.

- [36] Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Katz, and Nikolaos Aletras. LexGLUE: A benchmark dataset for legal language understanding in English. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4310–4330, Dublin, Ireland, May 2022. Association for Computational Linguistics.
- [37] Ilias Chalkidis, Vasiliki Koukouloti, and Nikolaos Aletras. Paragraph-level rationalization of legal judgments using reinforcement learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1081–1090, 2021.
- [38] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada, July 2017. Association for Computational Linguistics.
- [39] D.Z. Chen. Efficient parallel binary search on sorted arrays, with applications. *IEEE Transactions on Parallel and Distributed Systems*, 6(4):440–445, 1995.
- [40] Guangfu Chen, Chen Xu, Jingyi Wang, Jianwen Feng, and Jiqiang Feng. Robust non-negative matrix factorization for link prediction in complex networks using manifold regularization and sparse learning. *Physica A: Statistical Mechanics and its Applications*, 539:122882, 2020.
- [41] Hainan Chen and Xiaowei Luo. An automatic literature knowledge graph and reasoning network modeling framework based on ontology and natural language processing. *Advanced Engineering Informatics*, 42:100959, 2019.

- [42] Penghe Chen, Yu Lu, Vincent W. Zheng, Xiyang Chen, and Boda Yang. Knowedu: A system to construct knowledge graph for education. *IEEE Access*, 6:31553–31563, 2018.
- [43] Weijian Chen, Yixin Cao, Fuli Feng, Xiangnan He, and Yongdong Zhang. Hognr: Explainable sparse knowledge graph completion via high-order graph reasoning network. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–13, 2024.
- [44] Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, Qibin Zhao, and Danilo P. Mandic. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions. *Found. Trends Mach. Learn.*, 9(4–5):249–429, December 2016.
- [45] Andrzej Cichocki, Danilo P Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar F Caiafa, and Anh Huy Phan. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32(2):145–163, 2015.
- [46] Andrzej Cichocki, Rafal Zdunek, Anh-Huy Phan, and Shun-ichi Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley, 2009.
- [47] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. SPECTER: Document-level representation learning using citation-informed transformers. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online, July 2020. Association for Computational Linguistics.

- [48] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 4 edition, 2022.
- [49] Jeremy Costello, Marek Reformat, and Francois Bolduc. Leveraging knowledge graphs and natural language processing for automated web resource labeling: Knowledge mobilization in neurodevelopmental disorders. (preprint). *Journal of Medical Internet Research*, 25, 12 2022.
- [50] Evan Rosen Daniel Ramage. Stanford topic modeling toolbox. <https://downloads.cs.stanford.edu/nlp/software/tmt/tmt-0.4/>, 2009. Accessed: June 1, 2023.
- [51] Derek Desantis, Erik Skau, Duc P. Truong, and Boian Alexandrov. Factorization of binary matrices: Rank relations, uniqueness and model selection of boolean decomposition. *ACM Trans. Knowl. Discov. Data*, 16(6), July 2022.
- [52] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.
- [53] Jaspreet Singh Dhani, Ruchika Bhatt, Balaji Ganesan, Parikshet Sirohi, and Vasudha Bhatnagar. Similar cases recommendation using legal knowledge graphs, 2024.
- [54] Biao Dong, Haoze Yu, and Haisheng Li. A knowledge graph construction approach for legal domain. *Tehnicki vjesnik - Technical Gazette*, 2021.
- [55] David Donoho and Victoria Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2003.
- [56] Liang Duan, Shuai Ma, Charu Aggarwal, Tiejun Ma, and Jinpeng Huai. An ensem-

- ble approach to link prediction. *IEEE Transactions on Knowledge and Data Engineering*, 29(11):2402–2416, 2017.
- [57] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization, 2024.
- [58] Candace Edwards. Hybrid context retrieval augmented generation pipeline: Llm-augmented knowledge graphs and vector database for accreditation reporting assistance, 2024.
- [59] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. In *Proceedings of SEMANTiCS 2016*, volume 1695 of *CEUR Workshop Proceedings*, pages 1–4, 2016.
- [60] Charles Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th international conference on Machine Learning (ICML-03)*, pages 147–153, 2003.
- [61] Employment law – healthy workplaces – earned sick leave; use and accrual, 2021. Effective July 1, 2022.
- [62] Jonathan Scott Enderle. Topic modeling tool. <https://senderle.github.io/topic-modeling-tool/documentation/2017/01/06/quickstart.html>, 2023. Accessed: June 1, 2023.
- [63] Maksim Eren, Nick Solovyev, Ryan Barron, Manish Bhattarai, Duc Truong, Ismael Boureima, Erik Skau, Kim Rasmussen, and Boian Alexandrov. Tensor Extraction of Latent Features (T-ELF). Technical report, Los Alamos National Laboratories, October 2023.
- [64] Maksim E. Eren, Manish Bhattarai, Robert J. Joyce, Edward Raff, Charles Nicholas, and Boian S. Alexandrov. Semi-supervised classification of malware families under extreme class

imbalance via hierarchical non-negative matrix factorization with automatic model selection.

*ACM Trans. Priv. Secur.*, sep 2023. Just Accepted.

- [65] Maksim E. Eren, Manish Bhattarai, Nicholas Solovyev, Luke E. Richards, Roberto Yus, Charles Nicholas, and Boian S. Alexandrov. One-shot federated group collaborative filtering. In *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 647–652, 2022.
- [66] Maksim E Eren, Luke E Richards, Manish Bhattarai, Roberto Yus, Charles Nicholas, and Boian S Alexandrov. Fedsplit: One-shot federated recommendation system based on non-negative joint matrix factorization and knowledge distillation. *arXiv preprint arXiv:2205.02359*, 2022.
- [67] Maksim Ekin Eren, Nick Solovyev, Manish Bhattarai, Kim Rasmussen, Charles Nicholas, and Boian Alexandrov. Senmfk-split: Large corpora topic modeling by semantic non-negative matrix factorization with automatic model selection. In *Proceedings of the ACM Symposium on Document Engineering 2022, DocEng '22*, New York, NY, USA, 2022. Association for Computing Machinery.
- [68] Maksim Ekin Eren, Nick Solovyev, Edward Raff, Charles Nicholas, and Ben Johnson. Covid-19 kaggle literature organization. In *Proceedings of the ACM Symposium on Document Engineering 2020, DocEng '20*, New York, NY, USA, 2020. Association for Computing Machinery.
- [69] Shangbin Feng, Zhaoxuan Tan, Wenqian Zhang, Zhenyu Lei, and Yulia Tsvetkov. KALM: Knowledge-aware integration of local, document, and global contexts for long document understanding, 2023.
- [70] Cédric Févotte and A Taylan Cemgil. Nonnegative matrix factorizations as probabilistic

- inference in composite models. In *17th European Signal Processing Conference*, pages 1913–1917, 2009.
- [71] D. Flocco, B. Palmer-Toy, R. Wang, H. Zhu, R. Sonthalia, J. Lin, A. L. Bertozzi, and P. Jeffrey Brantingham. An analysis of covid-19 knowledge graph construction and applications. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 2631–2640, Los Alamitos, CA, USA, dec 2021. IEEE Computer Society.
- [72] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 1*, 2:559–572, 1901.
- [73] Zuohui Fu, Yikun Xian, Yaxin Zhu, Shuyuan Xu, Zelong Li, Gerard de Melo, and Yongfeng Zhang. Hoops: Human-in-the-loop graph reasoning for conversational recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '21*, page 2415–2421, New York, NY, USA, 2021. Association for Computing Machinery.
- [74] Google. Google Bard, 2023.
- [75] Thomas Griffiths, Michael Jordan, Joshua Tenenbaum, and David Blei. Hierarchical topic models and the nested chinese restaurant process. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2003.
- [76] Maarten Grootendorst. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*, 2022.
- [77] Dandan Guo, Bo Chen, Ruiying Lu, and Mingyuan Zhou. Recurrent hierarchical topic-guided neural language models, 2020.

- [78] Sunny Gupta, Jun-Jie Zhang, Jincheng Lei, Henry Yu, Mingjie Liu, Xiaolong Zou, and Boris I Yakobson. Two-dimensional transition metal dichalcogenides: A theory and simulation perspective. *Chemical Reviews*, 2025.
- [79] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML'20*. JMLR.org, 2020.
- [80] Jamie Haddock, Tyler Will, Joshua Vendrow, Runyu Zhang, Denali Molitor, Deanna Needell, Mengdi Gao, and Eli Sadvnik. Neural nonnegative matrix factorization for hierarchical multilayer topic modeling. *Sampling Theory, Signal Processing, and Data Analysis*, 22(1):4, 2023.
- [81] Harry Horace Harman. *Modern factor analysis*. University of Chicago Press, Chicago, second edition, revised. edition, 1967.
- [82] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84, 1970.
- [83] Ahmad B.A. Hassanat. Furthest-pair-based binary search tree for speeding big data classification using k-nearest neighbors. *Big Data*, 6(3):225–235, 2018.
- [84] Winston Haynes. *Wilcoxon Rank Sum Test*, pages 2354–2355. Springer New York, New York, NY, 2013.
- [85] Jerry L. Hintze and Ray D. Nelson. Violin plots: A box plot-density trace synergism. *The American Statistician*, 52(2):181–184, 1998.

- [86] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, and Sebastian Rudolph. OWL 2 web ontology language primer (second edition). W3C Recommendation, 2012. <https://www.w3.org/TR/owl2-primer/>.
- [87] HKU Data Science Lab. Lightrag: Simple and fast retrieval-augmented generation. <https://github.com/HKUDS/LightRAG>, 2025.
- [88] Marvin Hofer, Daniel Obraczka, Alieh Saeedi, Hanna Köpcke, and Erhard Rahm. Construction of knowledge graphs: Current state and challenges. *Information*, 15(8):509, 2024.
- [89] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d’Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, Jose Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *ACM Comput. Surv.*, 54(4):71:1–71:37, 2021.
- [90] Kurt Hornik, Ingo Feinerer, Martin Kober, and Christian Buchta. Spherical k-means clustering. *Journal of Statistical Software*, 50:1–22, 09 2012.
- [91] Mehdi Hosseinzadeh Aghdam. A novel constrained non-negative matrix factorization method based on users and items pairwise relationship for recommender systems. *Expert Systems with Applications*, 195:116593, 2022.
- [92] Kexin Huang, Ying Jin, Emmanuel Candes, and Jure Leskovec. Uncertainty quantification over graph with conformalized graph neural networks. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 26699–26721. Curran Associates, Inc., 2023.

- [93] Zhiqi Huang, Shahrzad Naseri, Hamed Bonab, Sheikh Muhammad Sarwar, and James Allan. Hierarchical transformer-based query by multiple documents. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR '23*, page 105–115, New York, NY, USA, 2023. Association for Computing Machinery.
- [94] Nicolas Hug. Surprise: A python library for recommender systems. *Journal of Open Source Software*, 5(52):2174, 2020.
- [95] Aapo Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, United Kingdom, 2001.
- [96] Michael Iannacone, Shawn Bohn, Grant Nakamura, John Gerth, Kelly Huffer, Robert Bridges, Erik Ferragut, and John Goodall. Developing an ontology for cyber security knowledge graphs. In *Proceedings of the 10th Annual Cyber and Information Security Research Conference, CISR '15*, New York, NY, USA, 2015. Association for Computing Machinery.
- [97] InfiniFlow. Construct knowledge graph. [https://ragflow.io/docs/dev/construct\\_knowledge\\_graph](https://ragflow.io/docs/dev/construct_knowledge_graph), 2025.
- [98] InfiniFlow. Ragflow: Open-source rag engine. <https://github.com/infiniflow/ragflow>, 2025. Apache-2.0 License.
- [99] S.M. Ashiqul Islam, Marcos Diaz-Gay, Yang Wu, Mark Barnes, Raviteja Vangara, Erik N. Bergstrom, Yudou He, Mike Vella, Jingwei Wang, Jon W. Teague, Peter Clapham, Sarah Moody, Sergey Senkin, Yun Rose Li, Laura Riva, Tongwu Zhang, Andreas J. Gruber, Christopher D. Steele, Burcak Otlu, Azhar Khandekar, Ammal Abbasi, Laura Humphreys, Natalia Syulyukina, Samuel W. Brady, Boian S. Alexandrov, Nischalan Pillay, Jinghui Zhang, David J. Adams, Inigo Martincorena, David C. Wedge, Maria Teresa Landi, Paul Brennan, Michael R.

- Stratton, Steven G. Rozen, and Ludmil B. Alexandrov. Uncovering novel mutational signatures by de novo extraction with sigprofilerextractor. *Cell Genomics*, 2(11):100179, 2022.
- [100] Yu Ito, Shin ichi Oeda, and Kenji Yamanishi. *Rank Selection for Non-negative Matrix Factorization with Normalized Maximum Likelihood Coding*, pages 720–728. Society for Industrial and Applied Mathematics, 2016.
- [101] Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online, April 2021. Association for Computational Linguistics.
- [102] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023.
- [103] Anil K Jain and Richard C Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.
- [104] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.
- [105] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514, 2022.

- [106] Jinhao Jiang, Kun Zhou, Xin Zhao, Yaliang Li, and Ji-Rong Wen. ReasoningLM: Enabling structural subgraph reasoning in pre-trained language models for question answering over knowledge graph. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3721–3735, Singapore, December 2023. Association for Computational Linguistics.
- [107] Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Zheng Li, Ruirui Li, Xi-anfeng Tang, Suhang Wang, Yu Meng, and Jiawei Han. Graph chain-of-thought: Augmenting large language models by reasoning on graphs, 2024.
- [108] Xin Jin and Jiawei Han. *K-Means Clustering*, pages 563–564. Springer US, Boston, MA, 2010.
- [109] Yuan Jin, He Zhao, Ming Liu, Lan Du, and Wray Buntine. Neural attention-aware hierarchical topic model, 2021.
- [110] Christopher C. Johnson. Logistic matrix factorization for implicit feedback data. In *Proceedings of the NIPS 2014 Workshop on Distributed Machine Learning and Matrix Computations (co-located with NeurIPS 2014)*, 2014.
- [111] Ian T Jolliffe. *Principal component analysis for special types of data*. Springer, 2002.
- [112] Justia. Justia: Free law and legal information, n.d. Accessed: 2025-01-07.
- [113] Richard M. Karp and Robert D. Kleinberg. Noisy binary search and its applications. In *ACM-SIAM Symposium on Discrete Algorithms*, 2007.
- [114] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering.

- In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020.
- [115] Leonard Kaufman and Peter J. Rousseeuw. *Partitioning Around Medoids (Program PAM)*, chapter 2, pages 68–125. John Wiley and Sons, Ltd, 1990.
- [116] Yong-Deok Kim and Seungjin Choi. Weighted nonnegative matrix factorization. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1541–1544, 2009.
- [117] Rodney Michael Kinney, Chloe Anastasiades, Russell Authur, Iz Beltagy, Jonathan Bragg, Alexandra Buraczynski, Isabel Cachola, Stefan Candra, Yoganand Chandrasekhar, Arman Cohan, Miles Crawford, Doug Downey, Jason Dunkelberger, Oren Etzioni, Rob Evans, Sergey Feldman, Joseph Gorney, David W. Graham, F.Q. Hu, Regan Huff, Daniel King, Sebastian Kohlmeier, Bailey Kuehl, Michael Langan, Daniel Lin, Haokun Liu, Kyle Lo, Jaron Lochner, Kelsey MacMillan, Tyler Murray, Christopher Newell, Smita Rao, Shaurya Rohatgi, Paul L Sayre, Zejiang Shen, Amanpreet Singh, Luca Soldaini, Shivashankar Subramanian, A. Tanaka, Alex D Wade, Linda M. Wagner, Lucy Lu Wang, Christopher Wilhelm, Caroline Wu, Jiangjiang Yang, Angele Zamarron, Madeleine van Zuylen, and Daniel S. Weld. The semantic scholar open data platform. *ArXiv*, abs/2301.10140, 2023.
- [118] Donald E. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Addison–Wesley, 2nd edition, 1998.
- [119] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.
- [120] Krm, inc. v. caviness, 1996. Reversed and remanded for further proceedings.

- [121] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 957–966, Lille, France, 07–09 Jul 2015. PMLR.
- [122] Tiziano Labruna, Jon Ander Campos, and Gorka Azkune. When to retrieve: Teaching llms to utilize information retrieval effectively. *ArXiv*, abs/2404.19705, 2024.
- [123] Jakub Lála, Odhran O’Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodrigues, and Andrew D White. Paperqa: Retrieval-augmented generative agent for scientific research. *arXiv preprint arXiv:2312.07559*, 2023.
- [124] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.
- [125] Dohyeon Lee, Seung-won Hwang, Kyungjae Lee, Seungtaek Choi, and Sunghyun Park. On complementarity objectives for hybrid retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 13357–13368. Association for Computational Linguistics, 2023.
- [126] Donghyuk Lee, Difei Wang, Xiaohong R Yang, Jianxin Shi, Maria Teresa Landi, and Bin Zhu. Sutor: selecting the number of mutational signatures through cross-validation. *PLoS Computational Biology*, 18(4):e1009309, 2022.
- [127] Meng-Chieh Lee, Qi Zhu, Costas Mavromatis, Zhen Han, Soji Adeshina, Vassilis N. Ioannidis, Huzefa Rangwala, and Christos Faloutsos. Hybgrag: Hybrid retrieval-augmented generation on textual and relational knowledge bases. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025. Also available as arXiv:2412.16311.

- [128] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Urvashi Khandelwal, Mikel Artetxe, Hailey Schoelkopf, Moin Nadeem Sung, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474, 2020.
- [129] Mingze Li, Haoran Yang, Zhaotai Liu, Mirza Mohtashim Alam, Ebrahim, Harald Sack, and Genet Asefa Gesese. KGMistral: Towards boosting the performance of large language models for question answering with knowledge graph integration. In *Workshop on Deep Learning and Large Language Models for Knowledge Graphs*, 2024.
- [130] Pengyu Li, Christine Tseng, Yaxuan Zheng, Joyce A. Chew, Longxiu Huang, Benjamin Jarman, and Deanna Needell. Guided semi-supervised non-negative matrix factorization on legal documents. *ArXiv*, abs/2201.13324, 2022.
- [131] Stan Z. Li and Anil Jain, editors. *Hamming Distance*, pages 668–668. Springer US, Boston, MA, 2009.
- [132] Yuhan Li, Zhixun Li, Peisong Wang, Jia Li, Xiangguo Sun, Hong Cheng, and Jeffrey Xu Yu. A survey of graph meets large language model: Progress and future directions. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-234*, 2023.
- [133] Wenlong Liang, Qihao Liu, Hongcheng Zhang, Weizhi Li, Zhao Chen, Fenglin Zhang, Jianlong Fu, and Dongmei Feng. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*, 2023.
- [134] Jerry Liu. LlamaIndex, 11 2022.
- [135] Qinghua Liu, Andrew Henry Reiner, Arnaldo Frigessi, and Ida Scheel. Diverse personalized

- recommendations with uncertainty from implicit preference data with the bayesian mallows model. *Knowledge-Based Systems*, 186:104960, 2019.
- [136] Xianglong Liu, Zhujin Li, Cheng Deng, and Dacheng Tao. Distributed adaptive binary quantization for fast nearest neighbor search. *IEEE Transactions on Image Processing*, 26(11):5324–5336, 2017.
- [137] Yong Liu, Min Wu, Chunyan Miao, Peilin Zhao, and Xiao-Li Li. Neighborhood regularized logistic matrix factorization for drug-target interaction prediction. *PLOS Computational Biology*, 12(2):1–26, 02 2016.
- [138] Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun’ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.
- [139] LINHAO LUO, Yuan-Fang Li, Reza Haf, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *The Twelfth International Conference on Learning Representations*, 2024.
- [140] Xin Lv, Xu Han, Lei Hou, Juanzi Li, Zhiyuan Liu, Wei Zhang, Yichi Zhang, Hao Kong, and Suhui Wu. Dynamic anticipation and completion for multi-hop reasoning over sparse knowledge graph. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5694–5703, Online, November 2020. Association for Computational Linguistics.

- [141] Xindi Ma, Jie Gao, Xiaoyu Liu, Taiping Zhang, and Yuanyan Tang. Probabilistic non-negative matrix factorization with binary components. *Mathematics*, 9(11), 2021.
- [142] Yixiao Ma, Yunqiu Shao, Yueyue Wu, Yiqun Liu, Ruizhe Zhang, Min Zhang, and Shaoping Ma. Lecard: A legal case retrieval dataset for chinese law system. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 2342–2348, New York, NY, USA, 2021. Association for Computing Machinery.
- [143] David JC MacKay. Bayesian nonlinear modeling for the prediction competition. *ASHRAE transactions*, 100(2):1053–1062, 1994.
- [144] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, 1967.
- [145] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, and Graham Neubig. Self-refine: Iterative refinement with self-feedback. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.
- [146] Yury A. Malkov and Dmitry A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020.
- [147] Paolo Manghi, Andrea Mannocci, Francesco Osborne, Dimitris Sacharidis, Angelo Salatino, and Thanasis Vergoulis. New trends in scientific knowledge graphs and research impact assessment, 2021.

- [148] Xueyu Mao, Purnamrita Sarkar, and Deepayan Chakrabarti. On mixed memberships and symmetric nonnegative matrix factorizations. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 2324–2333. JMLR.org, 2017.
- [149] Martin Marinov, Youcef Benkhedda, Goran Nenadic, and Riza Batista-Navarro. Relation extraction for constructing knowledge graphs: Enhancing the searchability of community-generated digital content (CGDC) collections. In *Workshop on Deep Learning and Large Language Models for Knowledge Graphs*, 2024.
- [150] Nicholas Matsumoto, Jay Moran, Hyunjun Choi, Miguel E Hernandez, Mythreye Venkatesan, Paul Wang, and Jason H Moore. KRAGEN: a knowledge graph-enhanced RAG framework for biomedical problem solving using large language models. *Bioinformatics*, 40(6):btac353, 06 2024.
- [151] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.
- [152] Message Passing Interface Forum. Mpi: A message-passing interface standard, version 3.1. <https://www.mpi-forum.org/docs/mpi-3.1/mpi31-report.pdf>, 2015.
- [153] Microsoft. Graphrag documentation and code. GitHub repository and docs, 2024.
- [154] Microsoft Research. Graphrag: Unlocking llm discovery with graph-based RAG. <https://www.microsoft.com/en-us/research/blog/graphrag-unlocking-llm-discovery-with-graph-rag/>, 2024.
- [155] Pauli Miettinen and Stefan Neumann. Recent developments in boolean matrix factorization. *arXiv preprint arXiv:2012.03127*, 2020.

- [156] Sara Mifrah and El Habib Benlahmar. Topic modeling with transformers for sentence-level using coronavirus corpus. *International Journal of Interactive Mobile Technologies (iJIM)*, 16(17):pp. 50–59, Sep. 2022.
- [157] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.
- [158] Laura Muzzarelli, Susanne Weis, Simon B. Eickhoff, and Kaustubh R. Patil. Rank selection in non-negative matrix factorization: systematic comparison and a new mad metric. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019.
- [159] Morten MØrup and Lars Kai Hansen. Tuning pruning in sparse non-negative matrix factorization. In *2009 17th European Signal Processing Conference*, pages 1923–1927. IEEE, 2009.
- [160] Benjamin T Nebgen, Raviteja Vangara, Miguel A Hombrados-Herrera, Svetlana Kuksova, and Boian S Alexandrov. A neural network for determination of latent dimensionality in non-negative matrix factorization. *Machine Learning: Science and Technology*, 2(2):025012, 2021.
- [161] Elena Nenova, Dmitry I Ignatov, and Andrey V Konstantinov. An fca-based boolean matrix factorisation for collaborative filtering. *arXiv preprint arXiv:1310.4366*, 2013.
- [162] Neo4j, Inc. Neo4j: The #1 platform for connected data. <https://neo4j.com/>, 2023.
- [163] New mexico constitution, article iv, section 22 – governor’s approval or veto of bills. <https://law.justia.com/constitution/new-mexico/article-4/section-22/>, 1912. Accessed: 2025-03-18.

- [164] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proc. IEEE*, 104(1):11–33, 2016.
- [165] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three way model for collective learning on multi relational data. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML 2011)*, pages 809–816, Bellevue, WA, USA, June 2011. Omnipress.
- [166] Andreas L Opdahl, Tareq Al-Moslmi, Duc-Tien Dang-Nguyen, Marc Gallofré Ocaña, Bjørnar Tessem, and Csaba Veres. Semantic knowledge graphs for the news: A review. *ACM Computing Surveys*, 55(7):1–38, 2022.
- [167] OpenAI. GPT-3.5-based ChatGPT, 2021.
- [168] OpenAI. Function calling and other API updates. OpenAI Blog (June 2023), <https://openai.com/blog/function-calling-and-other-api-updates>, 2023.
- [169] OpenAI. Openai api, 2024. Accessed: 2025-01-08.
- [170] OpenAI’s DALL-E. Visual representations of llms, kg, & rag concepts, 2024.
- [171] Malte Ostendorff, Nils Rethmeier, Isabelle Augenstein, Bela Gipp, and Georg Rehm. Neighborhood contrastive learning for scientific document representations with citation embeddings. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11670–11688, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.
- [172] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

- [173] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.
- [174] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599, 2024.
- [175] Shirui Pan, Yizhen Zheng, and Yixin Liu. Integrating graphs with large language models: Methods and prospects. *IEEE Intelligent Systems*, 39(1):64–68, 2024.
- [176] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.
- [177] Fen Pei, Qingya Shi, Haotian Zhang, and Ivet Bahar. Predicting protein–protein interactions using symmetric logistic matrix factorization. *Journal of Chemical Information and Modeling*, 61(4):1670–1682, 2021. PMID: 33831302.
- [178] Lihua Peng, Yue Zhao, and Xiaolin Zhang. Matrix factorization for missing link prediction with negative sample selection. *PLOS ONE*, 18(7):e0289568, 2023.
- [179] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
- [180] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim

- Rocktäschel, and Sebastian Riedel. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of NAACL-HLT 2021*, pages 2523–2544. Association for Computational Linguistics, 2021.
- [181] Aditya Pingle, Aritran Piplai, Sudip Mittal, Anupam Joshi, James Holt, and Richard Zak. Relext: Relation extraction using deep learning approaches for cybersecurity knowledge graph improvement. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 879–886, 2019.
- [182] Nicholas Pipitone and Ghita Hour Alami. Legalbench-rag: A benchmark for retrieval-augmented generation in the legal domain, 2024.
- [183] Juliano Rabelo, Randy Goebel, Mi-Young Kim, Yoshinobu Kano, Masaharu Yoshioka, and Ken Satoh. Overview and discussion of the competition on legal information extraction/entailment (coliee) 2021. *The Review of Socionetwork Strategies*, 16(1):111–133, April 2022.
- [184] Toran Bruce Richards and *et al.* AutoGPT: An autonomous GPT-4 experiment. <https://github.com/Significant-Gravitas/Auto-GPT>, 2023. Software available online.
- [185] Livio Robaldo, Guido Boella, and Leendert van der Torre. A neural approach to automated legal case matching. *Artificial Intelligence and Law*, 28(1):65–81, 2020.
- [186] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.
- [187] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.
- [188] Diego Sanmartin. Kg-rag: Bridging the gap between knowledge and creativity, 2024.

- [189] Lawrence K. Saul and Sam T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *J. Mach. Learn. Res.*, 4(null):119–155, December 2003.
- [190] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th Extended Semantic Web Conference (ESWC 2018)*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018.
- [191] SciPhi AI. R2r documentation: Introduction. <https://r2r-docs.sciphi.ai/introduction>, 2025.
- [192] SciPhi AI. R2r: Production-ready agentic rag system. <https://github.com/SciPhi-AI/R2R>, 2025. MIT License.
- [193] D. Sculley. Web-scale k-means clustering. In *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, page 1177–1178, New York, NY, USA, 2010. Association for Computing Machinery.
- [194] Bilin Shao, Xiaojun Li, and Genqing Bian. A survey of research hotspots and frontier trends of recommendation systems from the perspective of knowledge graph. *Expert Systems with Applications*, 165:113764, 2021.
- [195] Noah Shinn, Beck Labash, Ashwin Gopinath, and Karthik Narasimhan. Reflexion: Language agents with verbal reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.
- [196] Leslie F Sikos. Cybersecurity knowledge graphs. *Knowledge and Information Systems*, 65(9):3511–3531, 2023.

- [197] Robert F. Simmons. Natural language question-answering systems: 1969. *Commun. ACM*, 13(1):15–30, jan 1970.
- [198] Nikos Sismanis, Nikos Pitsianis, and Xiaobai Sun. Parallel search of k-nearest neighbors with synchronous operations. In *2012 IEEE Conference on High Performance Extreme Computing*, pages 1–6. IEEE, 2012.
- [199] Smith v. south, 1955. Remanded with instructions to make findings of fact and conclusions of law on default issues.
- [200] Nicholas Solovyev, Ryan Barron, Manish Bhattarai, Maksim E Eren, Kim Ø Rasmussen, and Boian S Alexandrov. Interactive distillation of large single-topic corpora of scientific papers. In *2023 International Conference on Machine Learning and Applications (ICMLA)*, pages 1000–1005. IEEE, 2023.
- [201] Nicholas Solovyev, Ryan Barron, Maksim E. Eren, Kim Ø. Rasmussen, Manish Bhattarai, Ismael D. Boureima, and Boian S. Alexandrov. Slic: Scientific leadership identification and characterization: Interactive distillation of large single-topic corpora of scientific papers. Technical report, DOE Data Days (D3) at Lawrence Livermore National Laboratory, LA-UR-23-30223, 2023.
- [202] Karthik Soman, Peter W Rose, John H Morris, Rabia E Akbas, Brett Smith, Braian Peetoom, Catalina Villouta-Reyes, Gabriel Ceron, Yongmei Shi, Angela Rizk-Jackson, et al. Biomedical knowledge graph-enhanced prompt generation for large language models. *arXiv preprint arXiv:2311.17330*, 2023.
- [203] Francesco Sovrano, Monica Palmirani, and Fabio Vitali. Legal knowledge extraction for

- knowledge graph based question-answering. In *International Conference on Legal Knowledge and Information Systems*, 2020.
- [204] Steven Squires, Adam Prügel-Bennett, and Mahesan Niranjan. Minimum description length as an objective function for non-negative matrix factorization. *arXiv preprint arXiv:1902.01632*, 2019.
- [205] Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W. Cohen. Open-domain question answering using early fusion of knowledge bases and text. In *Proceedings of EMNLP 2018*, pages 4231–4242, 2018.
- [206] K. Sycara, S.F. Roth, N. Sadeh, and M.S. Fox. Distributed constrained heuristic search. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1446–1461, 1991.
- [207] Jiejun Tan, Zhicheng Dou, Yutao Zhu, Peidong Guo, Kun Fang, and Ji-Rong Wen. Small models, big insights: Leveraging slim proxy models to decide when and what to retrieve for llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, Thailand, 2024. Association for Computational Linguistics.
- [208] Vincent YF Tan and Cédric Févotte. Automatic relevance determination in nonnegative matrix factorization with the  $\beta$ -divergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1592–1605, 2012.
- [209] Minghu Tang. A joint weighted nonnegative matrix factorization model via fusing attribute information for link prediction. In Chenggang Yang, Honggang Wang, and Yun Li, editors, *Mobile Multimedia Communications: 15th EAI International Conference, MobiMedia 2022, Virtual Event, July 22–24 2022, Proceedings*, volume 451 of *Lecture Notes of the Institute for*

- Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 190–205. Springer, Cham, January 2023.
- [210] Minghu Tang, Wei Yu, Xiaoming Li, Xue Chen, Wenjun Wang, and Zhen Liu. Cold-start link prediction via weighted symmetric nonnegative matrix factorization with graph regularization. *Computer Systems Science and Engineering*, 43(3):1069–1084, 2022.
- [211] Yanran Tang, Ruihong Qiu, Hongzhi Yin, Xue Li, and Zi Huang. Caselink: Inductive graph learning for legal case retrieval. *arXiv preprint*, 2024. arXiv:2403.17780.
- [212] Robert E. Tarjan. *Data Structures and Network Algorithms*. Society for Industrial and Applied Mathematics, 1983.
- [213] Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101:1566 – 1581, 2006.
- [214] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- [215] Sanju Mishra Tiwari, Fatima N. Al-Aswadi, and Devottam Gaurav. Recent trends in knowledge graphs: theory and practice. *Soft Computing*, 25:8337 – 8355, 2021.
- [216] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966c.
- [217] Richard A. Tyrrell and D. Alfred Owens. A rapid technique to assess the resting states of the eyes and other threshold phenomena: The modified binary search (mobs). *Behavior Research Methods, Instruments, & Computers*, 20(2):137–141, 1988.

- [218] Raviteja Vangara, Manish Bhattarai, Erik Skau, Gopinath Chennupati, Hristo Djidjev, Tom Tierney, James P Smith, Valentin G Stanev, and Boian S Alexandrov. Finding the number of latent topics with semantic non-negative matrix factorization. *IEEE access*, 9:117217–117231, 2021.
- [219] Raviteja Vangara, Kim Rasmussen, Gopinath Chennupati, and B. Alexandrov. Determination of the number of clusters by symmetric non-negative matrix factorization. In *SPIE*, page 15, 04 2021.
- [220] Raviteja Vangara, Erik Skau, Gopinath Chennupati, Hristo Djidjev, Thomas Tierney, James P Smith, Manish Bhattarai, Valentin G Stanev, and Boian S Alexandrov. Semantic nonnegative matrix factorization with automatic model determination for topic modeling. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 328–335. IEEE, 2020.
- [221] Joshua Vendrow, Jamie Haddock, and Deanna Needell. A generalized hierarchical nonnegative tensor decomposition. In *ICASSP 2022*, pages 4473–4477, 2022.
- [222] Felipe Viegas, Washington Cunha, Christian Gomes, Antônio Pereira, Leonardo C. da Rocha, and Marcos André Gonçalves. Cluhtm - semantic hierarchical topic modeling based on clu-words. In *Annual Meeting of the Association for Computational Linguistics*, 2020.
- [223] Somn Wadhwa, Silvio Amir, and Byron Wallace. Revisiting relation extraction in the era of large language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15566–15589, Toronto, Canada, July 2023. Association for Computational Linguistics.

- [224] Chengbin Wang, Xiaogang Ma, Jianguo Chen, and Jingwen Chen. Information extraction and knowledge graph construction from geoscience literature. *Computers & Geosciences*, 112:112–120, 2018.
- [225] Ruijie Wang, Yuchen Yan, Jialu Wang, Yuting Jia, Ye Zhang, Weinan Zhang, and Xinbing Wang. Acekg: A large-scale knowledge graph for academic data mining. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM '18*, page 1487–1490, New York, NY, USA, 2018. Association for Computing Machinery.
- [226] Shuting Wang, Alexander Ororbia, Zhaohui Wu, Kyle Williams, Chen Liang, Bart Pursel, and C. Lee Giles. Using prerequisites to extract concept maps from textbooks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM '16*, page 317–326, New York, NY, USA, 2016. Association for Computing Machinery.
- [227] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5329–5336, 2019.
- [228] Xuran Wang, Xinguang Zhang, Vanessa Hoo, Zhouhang Shao, and Xuguang Zhang. Legal-reasoner: A multi-stage framework for legal judgment prediction via large language models and knowledge integration. *IEEE Access*, PP:1–1, 01 2024.
- [229] Selma Wanna, Nicholas Solovyev, Ryan Barron, Maksim E. Eren, Manish Bhattarai, Kim Ø. Rasmussen, and Boian S. Alexandrov. Topictag: Automatic annotation of nmf topic models using chain of thought and prompt tuning with llms. In *Proceedings of the ACM Symposium on Document Engineering 2024, DocEng '24*, New York, NY, USA, 2024. Association for Computing Machinery.

- [230] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.
- [231] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc Le, and Denny Zhou. Chain-of-Thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.
- [232] Nirmalie Wiratunga, Ramitha Abeyratne, Lasal Jayawardena, Kyle Martin, Stewart Massie, Ikechukwu Nkisi-Orji, Ruvan Weerasinghe, Anne Liret, and Bruno Fleisch. Cbr-rag: Case-based reasoning for retrieval augmented generation in llms for legal question answering. In Juan A. Recio-Garcia, Mauricio G. Orozco-del Castillo, and Derek Bridge, editors, *Case-Based Reasoning Research and Development*, pages 445–460, Cham, 2024. Springer Nature Switzerland.
- [233] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–273, 07 2003.
- [234] Yangyang Xu, Wotao Yin, Zaiwen Wen, and Yin Zhang. An alternating direction algorithm for matrix completion with nonnegative factors. *Frontiers of Mathematics in China*, 7(2):365–384, 2012.
- [235] Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. Retrieval-augmented generation with knowledge graphs for customer service question answering. In *Proceedings of the 47th International ACM SIGIR Conference*

- on Research and Development in Information Retrieval*, SIGIR '24, page 2905–2909, New York, NY, USA, 2024. Association for Computing Machinery.
- [236] Zhong Xu, Wen-Xu Du, and Zhong-Qian Liu. Link prediction via matrix factorization: A perturbation-based approach. *Scientific Reports*, 6:38938, 2016.
- [237] Y.-J Yang, B. Xu, J.-W Hu, M.-H Tong, P. Zhang, and L. Zheng. Accurate and efficient method for constructing domain knowledge graph. *Ruan Jian Xue Bao/Journal of Software*, 29:2931–2947, 10 2018.
- [238] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- [239] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023.
- [240] Yabing Yao, Yaling He, Zhenyu Huang, et al. Deep non-negative matrix factorization with edge generator for link prediction in complex networks. *Applied Intelligence*, 54:592–613, 2024.
- [241] Hongbin Ye, Honghao Gui, Aijia Zhang, Tong Liu, Wei Hua, and Weiqiang Jia. Beyond isolation: Multi-agent synergy for improving knowledge graph construction, 2023.
- [242] Donghan Yu, Chenguang Zhu, Yuwei Fang, Wenhao Yu, Shuohang Wang, Yichong Xu, Xiang Ren, Yiming Yang, and Michael Zeng. Kg-fid: Infusing knowledge graph in fusion-in-decoder for open-domain question answering. *arXiv preprint*, 2022. arXiv:2110.04330.

- [243] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop, 2016.
- [244] Sheng Yu, Zheng Yuan, Jun Xia, Shengxuan Luo, Huaiyuan Ying, Sihang Zeng, Jingyi Ren, Hongyi Yuan, Zhengyun Zhao, Yucong Lin, Keming Lu, Jing Wang, Yutao Xie, and Heung-Yeung Shum. Bios: An algorithmically generated biomedical knowledge graph, 2022.
- [245] Vanni Zavarella, Juan Carlos Gamero, and Sergio Consoli. A few-shot approach for relation extraction domain adaptation using large language models. In *Workshop on Deep Learning and Large Language Models for Knowledge Graphs*, 2024.
- [246] Xu-Yao Zhang, Guo-Sen Xie, Xiuli Li, Tao Mei, and Cheng-Lin Liu. A survey on learning to reject. *Proceedings of the IEEE*, 111(2):185–215, 2023.
- [247] Jie Zhou, Xin Chen, Hang Zhang, and Zhe Li. Automatic knowledge graph construction for judicial cases, 2024.
- [248] Junlin Zhu, Xudong Luo, and Jiaye Wu. A bert-based two-stage ranking method for legal case retrieval. In Gerard Memmi, Baijian Yang, Linghe Kong, Tianwei Zhang, and Meikang Qiu, editors, *Knowledge Science, Engineering and Management*, pages 534–546, Cham, 2022. Springer International Publishing.
- [249] Yongjun Zhu, Erjia Yan, and Il-Yeol Song. The use of a graph-based system to improve bibliographic information retrieval: System design, implementation, and evaluation. *Journal of the Association for Information Science and Technology*, 68(2):480–490, 2017.
- [250] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466, 2018.



ProQuest Number: 32277944

INFORMATION TO ALL USERS

The quality and completeness of this reproduction is dependent on the quality and completeness of the copy made available to ProQuest.



Distributed by  
ProQuest LLC a part of Clarivate ( 2025).  
Copyright of the Dissertation is held by the Author unless otherwise noted.

This work is protected against unauthorized copying under Title 17,  
United States Code and other applicable copyright laws.

This work may be used in accordance with the terms of the Creative Commons license  
or other rights statement, as indicated in the copyright statement or in the metadata  
associated with this work. Unless otherwise specified in the copyright statement  
or the metadata, all rights are reserved by the copyright holder.

ProQuest LLC  
789 East Eisenhower Parkway  
Ann Arbor, MI 48108 USA