

Classification and Representation of Biological Interactions in the Context of a Baseline Model

by

Casey E. Hansen

B.A., Washington & Jefferson College, 2014

Submitted to the Graduate Faculty of the
Swanson School of Engineering in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

University of Pittsburgh

2022

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This dissertation was presented

by

Casey E. Hansen

It was defended on

December 10, 2021

and approved by

James Faeder, PhD, Associate Professor, Department of Computational & Systems Biology

Jonathan Vande Geest, PhD, Professor, Department of Bioengineering

Lance Davidson, PhD, William Kepler Whiteford Professor, Department of Bioengineering

Peter Spirtes, PhD, Marianna Brown Dietrich Professor, Department of Philosophy, Carnegie Mellon University

Dissertation Director: Natasa Miskov-Zivanov PhD, Assistant Professor, Departments of Electrical and Computer Engineering, Computational & Systems Biology, and Bioengineering

Copyright © by Casey E. Hansen

2022

Classification and Representation of Biological Interactions in the Context of a Baseline Model

Casey E. Hansen, PhD

University of Pittsburgh, 2022

Machine reading tools are able to quickly and automatically process vast amounts of information from relevant published literature, identifying and extracting the relevant information from a given paper or papers. This information can be used to build biological computational models or expand upon existing models. However, the information gleaned by machine readers is both vast and varied in quality. Machine readers must work to extract standardized biological interactions from inconsistent terminology and complex sentence structures, which sometimes leads to extraction errors. Here we present VIOLIN (Verifying Interactions of Likely Importance to the Network) a tool to automatically classify and judge biological interactions extracted from relevant literature. With VIOLIN, we are able to take these literature extracted events (LEEs) and compare them to an existing biological model, determining whether a given LEE agrees with the model (corroborates), introduces new information to the model (extends), disputes the model (contradicts), or requires manual review (flagged). Each LEE is assigned four numerical values to represent its relationship to the model system (Match Score), its classification category (Kind Score), its frequency (Evidence Score), and extraction confidence (Epistemic Value). These values are combined into a Total Score to allow for automatic filtering and classification of large sets of LEEs curated from multiple sources. We present VIOLIN in the context of five different models: melanoma, T-cell differentiation, the BDNF pathway as it relates to major depressive disorder, pancreatic cancer, and glioblastoma multiforme. These varied inputs show that VIOLIN has great

utility across many biological systems, making it a powerful computational tool. We also show how VIOLIN integrates with other modeling tools as part of a larger model extensions framework. The goal of this work is to be able to automatically extend existing biological models using the vast amounts of relevant information already available.

Table of Contents

Acknowledgements	xiv
Nomenclature	xv
1.0 Introduction.....	1
1.1 Motivation	1
1.2 Scope	2
2.0 Background	6
2.1 Biological Interactions and Networks.....	6
2.1.1 Biological Interactions	7
2.1.2 Model Structure and Attributes	9
2.2 Approaches for Dynamical Modeling	12
2.2.1 Event-based modeling.....	13
2.2.2 Element-based Modeling	16
2.3 Knowledge Sources.....	19
2.3.1 Interaction Databases	19
2.3.2 Network and Model Databases	22
2.3.3 Literature	26
2.4 Network Representation Formats.....	26
2.5 Tools.....	29
2.5.1 Directed Network Inference Tools.....	29
2.5.2 Machine Reading Tools	30
2.5.2.1 Functionality	30

2.5.2.2 Output.....	31
2.5.2.3 Machine Reading Errors.....	34
2.5.3 Network Assembly Tools	35
2.5.4 Model Recommendation Tools	36
3.0 VIOLIN Implementation.....	38
3.1 Classification Definitions	39
3.1.1 Match definitions.....	39
3.1.2 Model corroboration requirements	40
3.1.3 Model contradiction requirements	42
3.1.4 Model extension requirements	44
3.1.5 Flagged interactions	45
3.1.6 Path and loop finding.....	46
3.2 Scoring	47
3.2.1 Evidence Score.....	47
3.2.2 Match Score	48
3.2.3 Kind Score	49
3.2.4 Epistemic Value and Total Score	50
3.3 Algorithm	51
3.4 Selecting VIOLIN input	53
3.5 Compatibility with Other Tools	55
3.6 VIOLIN UI & Visualization	56
4.0 VIOLIN Outcomes.....	59
4.1 Models and LEE Construction.....	59

4.2 Classification of Extracted Events in Context of Baseline Model	64
4.3 Automated Curation of Extracted Events.....	70
4.4 Classification Strategies Guided by Modeling Goals	71
4.5 Pathfinding Discovery of Loops	72
4.6 Reading Errors and VIOLIN output.....	74
5.0 Further VIOLIN Evaluation.....	79
5.1 Match Score Testing	80
5.1.1 Match Score Parameters	80
5.1.2 Match Score Results.....	82
5.2 Kind Score Testing	83
5.2.1 Kind Score Parameters	83
5.2.2 Kind Score Results	85
5.3 Classification Scheme testing.....	86
5.3.1 VIOLIN Speed.....	91
5.4 Corroboration Testing	94
5.5 Extension Testing.....	96
6.0 VIOLIN Variability	98
6.1 Use Cases for Other Scoring Values	99
6.2 Integration into Larger Framework	101
6.2.1 Integration with Filtering Methods	101
6.2.2 Integration with a Model Extension Tool	106
7.0 Conclusions & Future Work.....	112
7.1 Future Work	113

Appendix A Input Model Descriptions	115
Appendix B Input LEE Sets.....	118
Appendix C Example Typical Baseline Model	120
Bibliography	122

List of Tables

Table 2-1. Summary of Interaction Databases.....	20
Table 2-2. Summary of Network and Model Databases.....	23
Table 5-1. Match value sets used to test the Match Score	81
Table 5-2. Kind value sets used to test the Kind Score.....	85
Table 6-1. Match Score values for different use cases	99
Table 6-2. Number of LEEs for Each CLARINET Input.....	108
Appendix Table 1. LEE set parameters	118
Appendix Table 2. The model information copied from NDEx to a spreadsheet that can be saved as an input file for VIOLIN	121
Appendix Table 3. Model C transformed into the BioRECIPES format	121

List of Figures

Figure 1-1. The role of VIOLIN in automating model assembly and model extension.....	3
Figure 1-2. Levels of information comparison as it relates to model building.....	4
Figure 2-1. A biological interaction represented as a directed signed edge between two nodes.	7
Figure 2-2. A toy example of a model, showing input and output nodes, structures of edges and paths as they exist in such a model.....	10
Figure 2-3. Biological reaction representation in different modeling approaches.....	13
Figure 2-4. Examples of four common representation formats.....	27
Figure 2-5. A model in the BioRECIPES representation format	27
Figure 2-6. Information relating to machine reading output:	34
Figure 3-1. The internal framework of the VIOLIN tool.....	38
Figure 3-2. Examples of VIOLIN classifications and subclassifications of LEEs:	41
Figure 3-3. Interaction chart showing the outcome of all possible comparisons of a given LEE to given baseline model interactions (MI).....	53
Figure 3-4. Example output from the VIOLIN visualization function,	57
Figure 4-1. Metrics for our tested VIOLIN inputs	63
Figure 4-2. Classification distribution of our tested VIOLIN inputs.....	64
Figure 4-3. The corroboration subclassification judgements of the tested VIOLIN input..	66
Figure 4-4. The extension subclassification judgements of the VIOLIN input.	67
Figure 4-5. The contradiction subclassification judgements of the VIOLIN input.	68
Figure 4-6. The flagged subclassification judgements of the VIOLIN input.....	69

Figure 4-7. Classification breakdown and subclassification judgements for our two “negative” LEE sets	70
Figure 4-8. Overview of attributes for different scenarios:	72
Figure 4-9. Prevalence of machine reading errors in subcategories of contradiction and flagged categories,	76
Figure 5-1. Number of nteractions in common in the top 100 scored	82
Figure 5-2. Normalized total scores for each of our tested Match Level parameters.	83
Figure 5-3. Results from the Kind Score Parameterization:	86
Figure 5-4. The outcome chart of our three tested classification schemes compared side-by-side, including the different flagged subclassification definitions.	88
Figure 5-5. The presence of direct LEEs across the main VIOLIN classifications, comparing our three tested classification schemes.....	89
Figure 5-6. The number of LEEs found in contradictions and flagged classifications across our 3 classification schemes: (a) shows the number of contradictions in the “raw” sets, (b) shows the number of flagged in the “raw” set, (c) shows the number of contradictions comparing the filtering methods to their associated “raw” LEE sets, and (d) shows the number of flagged LEEs, comparing the filtering methods to their associated “raw” LEE sets	90
Figure 5-7. The presence of phosphorylations across the main VIOLIN classification, comparing our tested classification schemes.	91
Figure 5-8. The average processing time of VIOLIN.	92
Figure 5-9. Average processing times across classification schemes.	93

Figure 5-10. Classification Distributions for our two corroborative sets; the LEE sets were judged against the original models C and D, respectively	95
Figure 5-11. Classification Distributions for our two corroborative sets; the LEE sets judged against the altered models C' and D', respectively	97
Figure 5-12. Representation of the KRAS-Proliferation pathway.	97
Figure 6-1. How VIOLIN fits in to a larger modeling framework with other automated modeling tools.....	98
Figure 6-2. VIOLIN outcomes of our different modeling use cases.	100
Figure 6-3. Schematic of the FLUTE pipeline from [19].....	102
Figure 6-4. VIOLIN outcomes in conjunction with FLUTE filtering methods.....	103
Figure 6-5. VIOLIN output results for each LEE set, broken down by classification category.	104
Figure 6-6. Showing the retention percentages across the subclassification categories.	105
Figure 6-7. Schematic of the CLARINET framework	107
Figure 6-8. VIOLIN judgement of RG1 and RG2.	108
Figure 6-9. Cluster outputs from CLARINET for each of our input sets	110
Appendix Figure 1. Model C as it is presented on the NDEx database	120

Acknowledgements

This work would have never been possible without the following people: Natasa Miskov-Zivanov, Emilee Holtzapple, Yasmine Ahmed, Gaoxing Chou, Stefan Andjelkovic, Khaled Sayed, Adam Butchy, Kara Bocan, Cheryl Telmer, Faeze Movahedi, Mitali Patil, Sudanshu Shekar, John Ohodnicki, Jingyao Wu, Boeun Lee, Brian Lupish, Oleg Velikokhatnyi, Yang Wang, Markus Dittrich, Julia Kisslinger, Neal Krishna, Erika Rispoli, James Faeder, Jonathan Vande Geest, Peter Spirtes, Lance Davidson, Ryan Higginbottom, Faun Doherty, Jenny Kline, Roman Wong, Michael Woltermann, John Zimmerman, Michael McCracken, Michael Petersen, Dana Paczek, Savanna Starko, Riley Kern, Kate Rodriguez, Ben Kopchick, Maggie Tesone, Ka’Ron Spriggs-Bethea, Christina Poole, Lisa D’Costa, Jill Chipman, Joey Ni, Jeffrey Gonzales, Philip & Jeannie Ameris, my taekwondo students, Clinton Johnson, Linda Hansen, Steven Hansen, and, of course, Turtle, Tiger, and Peanut.

They say it takes a village to raise a child. If that is true, then you are my village, and this is my offspring. I dedicate this work to you, and thank you, most profoundly.

I would also like to dedicate this work to my Algebra students: wherever you find yourself planted, find the will to bloom.

Nomenclature

ACCORDION	ACCelerating model RecommenDatION
BDNF	Brain-derived neurotrophic factor
BEL	Biological Expression Language
BNGL	BioNetGen Language
CLARINET	CLARIfying NETworks
CoLoMoTo	Consortium for Logical Models and Tools
FLUTE	FiLter for Understanding True Events
GBM	Glioblastoma Mulitforme
INDRA	the Integrated Network and Dynamical Reason Assembler
KEGG	Kyoto Encyclopedia of Genes and Genomes
LEE	Literature Extracted Event
MINERVA	Molecular Interaction NETwoRks VisuAlization
NDEx	the Network Data Exchange
ODE	Ordinary Differential Equation
PCI	Protein-chemical interaction
PGI	Protein-gene interaction
PPI	Protein-protein interaction
REACH	Reading and Assembling Contextual and Holistic mechanisms from text
RRBM	Reaction Rule-based model
SBML	Systems Biology Markup Language

SIGNOR	the SIGnaling Network Open Resource
STITCH	Search Tool for InTeracting Chemicals
STRING	Search Tool for the Retrieval of Interacting Genes/Proteins
UniProt	the Universal Protein resource
VIOLIN	Verifying interactions of Likely Importance to the Network

1.0 Introduction

Computational biological models are a powerful scientific tool. They are used to replace physical experiments, investigate systems, predict behavior, and make diagnostic choices. The process of creating such models can be laborious. Often, creating models requires initial experimentation, expert understanding, or deep investigation into the literature. Even then, once a model has been assembled and validated, the model must be maintained as new knowledge or needs arise. Machine reading engines, interaction databases, and user-curated model databases aid in creating biological models, but this still requires significant human effort. In this work, we seek to identify the gaps in current methodologies, and create a tool to aid the process by which computational models are built.

1.1 Motivation

There are many effective tools and databases currently used to understand biological systems and create computational models accurately representing signaling networks or investigating specific signaling pathways. While these methods and databases greatly improve the ease with which a model can be assembled, they still require a great deal of expert knowledge and manual assembly. Automatic assembly methods [1, 2] struggle to judge relevant data, meaning that the user must either manually curate data for model building or use a “raw” machine reading output, which may contain irrelevant, erroneous, or duplicate interactions. And while there are tools with a comprehensive and reliable source of biomolecular interactions from the literature,

they do not have the ability to automatically compare these interactions with existing models, and to classify them within the context of the model [2].

Recently, efforts have been made to assemble knowledge bases of the COVID-19 [3] and Rheumatoid Arthritis [4] networks. These required massive amounts of manual labor, on the order of hundreds of contributors [3], searching through the literature and databases for relevant information. Manually assembling and validating such volumes of information is time-consuming and is limited by human capabilities. Having a tool that would automate this process would greatly reduce the time and number of human contributors required to assemble such networks. Even for smaller modeling goals, such a tool would be able to compare a model to the vast amounts of available information, expanding either the depth or breadth, or validating a new model to agree with the current literature and expertise.

1.2 Scope

In Figure 1-1, we outline the general framework of model-building, which often starts with a specific question or hypothesis, and seeks to determine an answer to this question. The next step in this framework usually consists of querying through the existing literature and knowledge bases for relevant information. In some cases, a baseline model may already exist, however, it does not satisfy all properties, or it does not recapitulate the modeled system's behavior. This baseline model must then be tested and extended to meet the system properties, and model analysis confirms if these properties have been met. Finally, the verified and validated model can be further explored to find answers to initial questions and hypotheses or to make novel predictions about the system.

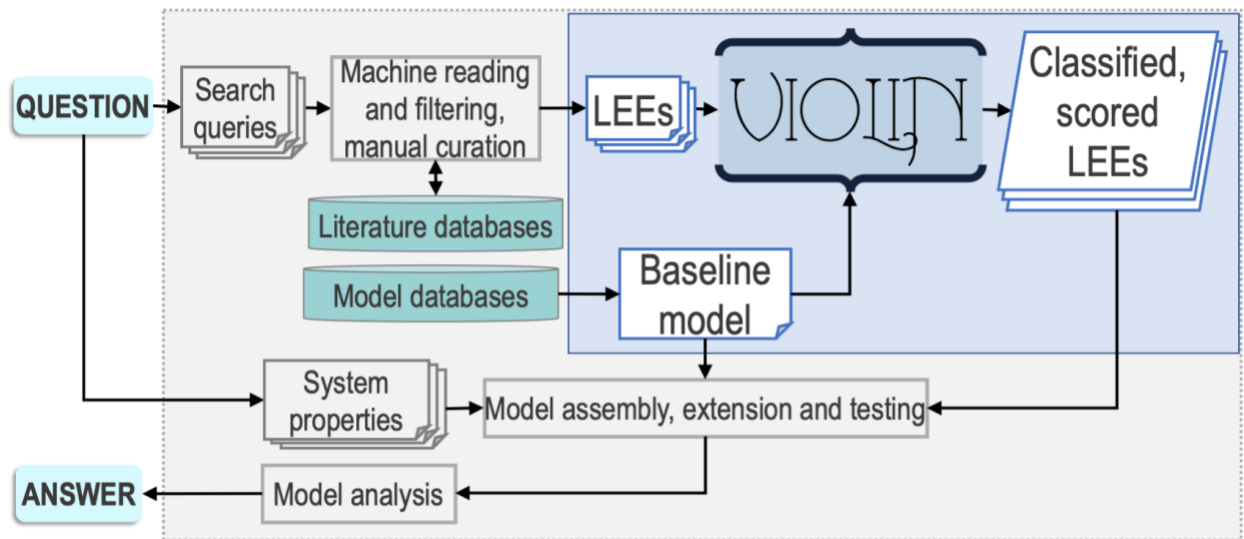


Figure 1-1. The role of VIOLIN in automating model assembly and model extension. To answer a question about the system, search queries and desirable system properties readable by machines (e.g., logical expressions) can be created. Machine reading engines use search queries to select relevant literature and extract a set of events (Literature Extracted Events - LEEs). VIOLIN uses the LEEs and an existing baseline model relevant for answering the question. The outputs of VIOLIN, classified and scored LEEs, can be used to extend the baseline model, or even assemble a new model. The model can be assembled/extended and tested iteratively until it satisfies desired properties. Analysis of the final model provides answers to the question.

Focusing in particular on the steps highlighted within the blue box in Figure 1-1, a set of interactions relevant to building a model can be assembled from literature and knowledge databases. However, due to the volume of knowledge available, the user must also judge how useful the available information is. We can identify three levels of judging useful information, as illustrated in Figure 1-2. At level 1, the user assembles a set of interactions and compares them to other interactions from literature or databases. An example of this would be finding a secondary source to corroborate an interaction. At level 2, the user compares interactions from the literature to an existing graph of causal influences. This is more advanced than level 1 as a causal influence

graph is a more sophisticated representation of interactions, where the nodes are elements representing biological entities and the edges are causal influences with polarity and sometimes mechanism details. However, doing this manually takes a great deal of time and expert knowledge. At level 3, the user can compare the causal influences of two graphs, the most advanced method of finding useful knowledge, as this level compares not only individual interactions, but the relationships between them.

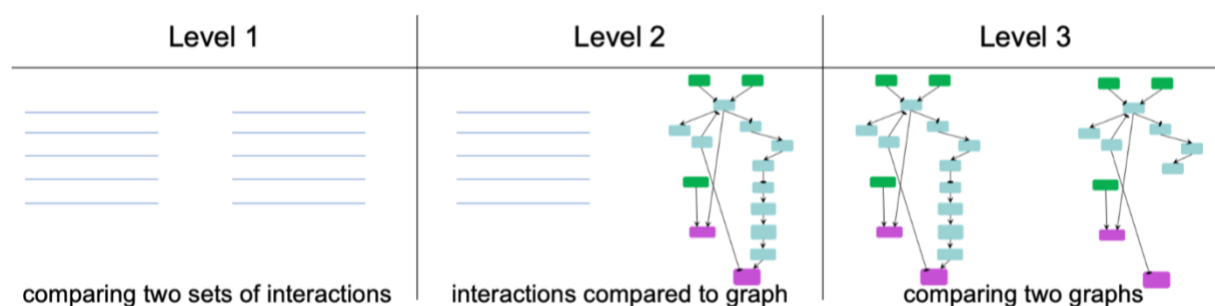


Figure 1-2. Levels of information comparison as it relates to model building.

At Level 1, one is comparing two sets of interactions, which may be assembled from multiple sources and vary in their usefulness. Level 2 is what is introduced by VIOLIN, comparing a set of interactions to an existing model. Level 3 is the most advanced level of comparison, being able to compare two directed graphs, representing networks of biological interactions.

To this end, we have designed VIOLIN (Validating Interactions Of Likely Importance to the Network), a tool for automatically classifying large sets of interactions, such as those extracted from research literature by machine readers, in the context of a given baseline model, which can be created from any model represented as a directed graph and split into individual directed interactions. VIOLIN automates the level 2 judgement by evaluating whether the new interactions support the information in the model (corroborations), identifies gaps or issues in the knowledge and published information (contradictions), suggests new connections to the model or more details

to existing connection (extensions), or indicates that the newly obtained information requires further investigation (flagged). Besides these main classification categories, VIOLIN uses various interaction attributes to determine more detailed classification subcategories, whenever the attribute information is available. Furthermore, VIOLIN has the capability to identify and search through pathways within a model when comparing to sets of new interactions, identifying new potential feedback and feed-forward loops. Finally, VIOLIN is fast, it performs the comparison and classification of large new interaction sets in the context of a baseline model at least three orders of magnitude faster than a human, thus enabling reliable high throughput curation tasks that would not be feasible otherwise.

As illustrated in Figure 1-1, VIOLIN can be used as part of the question-answer workflow, and being fully automated, it can significantly reduce the time of this common research process. This work is presented in 7 chapters. Chapter 2 introduces the structure of underlying methods, and the foundations of event-based and element-based modeling approached, as well as the current tools, databases and methods used to assemble computation models. Chapter 3 presents the implementation of the VIOLIN tool, and its functional framework, and its classification scheme. Chapter 4 describes the inputs created for VIOLIN development, describing the baseline models chosen, and how the literature information was assembled, as well as the outcomes of these inputs. In Chapter 5, we further evaluate the structure and capabilities of VIOLIN, suggesting default parameters for standard use, and showing how the classification scheme can be modified depending on modeling goals and questions. Chapter 6 investigates the integration VIOLIN with other modeling tools, towards the goal of creating a larger automated model-building framework. Finally, Chapter 7 concludes this work and outlines the areas of future work.

2.0 Background

In this section, we cover the background necessary to build a context for this work. Models that are used to study the dynamics of a system, that is, changes in its states over time, are often referred to as executable models. In Section 2.1 we provide definitions of an underlying structure and attributes of executable models of biological systems and in Section 2.2 we discuss common approaches for creating executable models. An overview of knowledge sources for biological interactions, networks, and models in Section 2.3 details all sources that can be used to create inputs for VIOLIN. Using many different online and published sources often requires translations between different representation formats, and therefore, we briefly describe the commonly used representation formats in Section 2.4. We summarize existing tools that can provide useful inputs for VIOLIN or utilize VIOLIN's outputs in Section 2.5.

2.1 Biological Interactions and Networks

The VIOLIN tool was developed to compare biological interactions curated from various sources to model networks, making use of specific attributes available from both interactions and networks to make judgements.

2.1.1 Biological Interactions

In this work, the main unit of comparison is a directed signed interaction between two elements, which, in the context of a graph, is represented as a directed signed edge between two nodes. We show an example of interaction in Figure 2-1 and in the following define more formally interactions (edges) and elements (nodes).

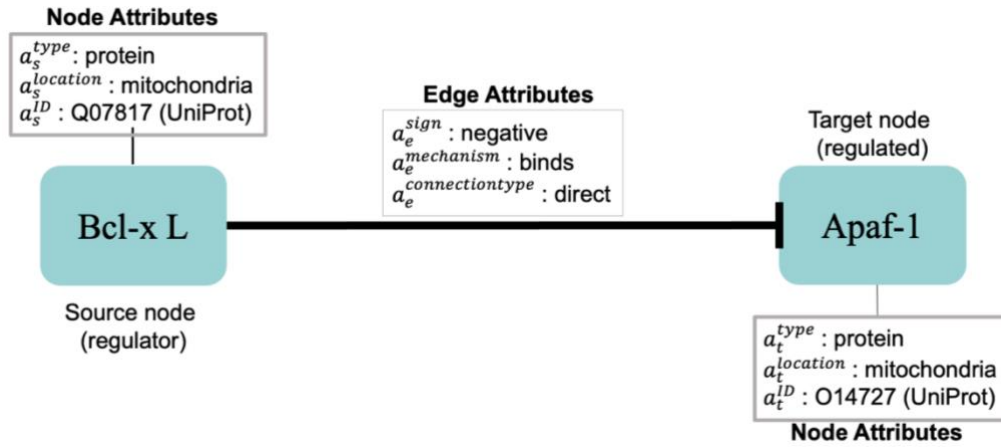


Figure 2-1. A biological interaction represented as a directed signed edge between two nodes.

Definition 1. An element (node), $v = v(\mathbf{a}^v)$, is defined by its name, type, and unique identifier (ID) and these attributes are written as a vector $\mathbf{a}^v = (a^{name}, a^{type}, a^{ID})$.

The attribute a^{name} is an element name, usually following the standard nomenclature used by biologists and in the literature (e.g., acronym ERK1 is used instead of a longer name “extracellular signal-regulated kinase 1”). The attribute a^{type} represents element type, usually genes, RNAs, proteins, chemicals, or biological processes. Biological entity names often have multiple synonyms (e.g., ERK1 may also be referred to as MAPK3), and therefore, unique

identifiers (IDs) are used, which are stored in attribute a^{ID} . These IDs can be obtained from standard databases such as UniProt [5], PubChem [6], or the Gene Ontology Databases (GO) [7].

In addition to these three required attributes, the node attribute vector \mathbf{a}^v may also include other attributes that help describe the element. For example, attributes $a^{location}$ and $a^{locationID}$ hold information about the cellular compartment where element is found and the compartment ID, respectively. We use the gene ontology database (GO) to obtain these location IDs [7].

Definition 2. A directed signed interaction (also referred to as a directed edge) $e = e(v_s, v_t, \mathbf{a}^e)$ is defined with its source element v_s , target element v_t , and vector of attributes \mathbf{a}^e . The interaction attribute vector always includes at least the sign a^{sign} and connection type $a^{connectiontype}$ attributes: $\mathbf{a}^e = (a^{sign}, a^{connectiontype})$. The direction of an interaction is always implicitly defined with source and target nodes, and therefore, not explicitly listed among its attributes.

The a^{sign} attribute indicates the sign (also referred to as polarity) of the influences, and it can take two values, $a^{sign} = \text{“positive”}$ (e.g., activation) or $a^{sign} = \text{“negative”}$ (e.g., inhibition). Sometimes, only the information about indirect influences on pathways of interest is known, and therefore, the attribute $a^{connectiontype}$ is used to indicate whether the interaction e is a direct physical interaction ($a^{connectiontype} = \text{“direct”}$) or an indirect influence from the source node to the target node ($a^{connectiontype} = \text{“indirect”}$). Since the interaction definition allows for indirect interactions, it is possible that source and target node are not in the same compartment, and this is the reason we assign the location attribute to nodes and not to the interaction.

The list of other attributes is not necessarily fixed; the components in it may vary, dependent on the goals of the analysis. A more specific information about the biological

mechanism of an interaction can be included in the $a^{mechanism}$ attribute. The edge attribute vector can also include the $a^{cellline}$, $a^{celltype}$, $a^{organism}$, and $a^{tissuetype}$ attributes, which hold the information about the cell line, cell type, organism, and tissue type where the interaction is observed, respectively. Finally, the $a^{evidence}$ attribute is used to store the information about the source where the interaction is found or observed. The evidence attribute can hold paper IDs (e.g., PMID [8]) and sentences from papers where the interaction is found, or an indicator that the information was obtained from an experiment or an expert’s suggestion.

2.1.2 Model Structure and Attributes

Since this work is focused on network models that can be represented as directed cyclic graphs, we provide here a definition of the network components that will be used throughout this thesis. We also show an example model in Figure 2-2.

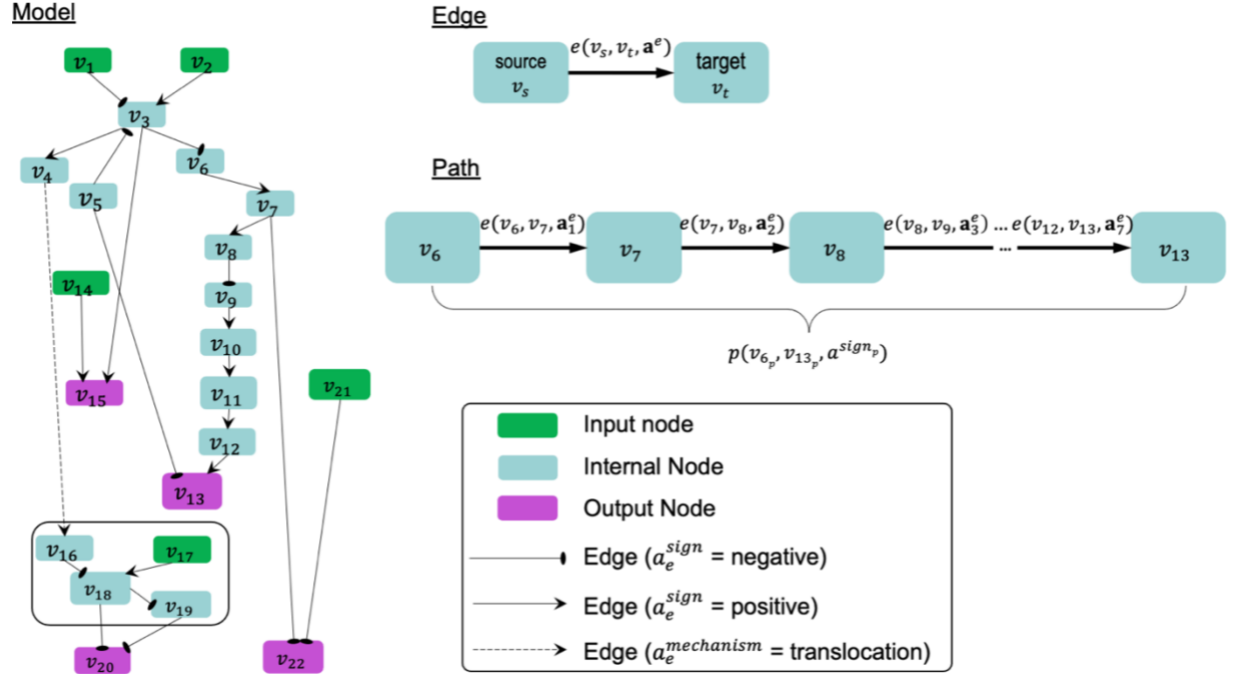


Figure 2-2. A toy example of a model, showing input and output nodes, structures of edges and paths as they exist in such a model.

Definition 3. The static structure of a model can be defined as a directed graph $G(V, E)$, where $V = \{v_1, v_2, \dots, v_N\}$ is a set of nodes, and each node $v_i = v(\mathbf{a}_i^v)$ is one model element, while $E = \{e_1, e_2, \dots, e_M\}$ is a set of directed edges, and an edge $e_j = e(v_{s_j}, v_{t_j}, \mathbf{a}_j^e)$ ($v_{s_j}, v_{t_j} \in V$) indicates a directed interaction between elements v_{s_j} and v_{t_j} , in which source node v_{s_j} influences target node v_{t_j} , an example of which is shown in Figure 2-2. Vectors \mathbf{a}_i^v and \mathbf{a}_j^e are formed following the definitions in Section 2.1.1. When the information about $a^{location}$, $a^{locationID}$, $a^{mechanism}$, $a^{cellline}$, $a^{celltype}$, $a^{organism}$, or $a^{tissuetype}$ is not available, these attributes are assigned an “empty” value.

Definition 4. An input node is a node that is not a target node of any edge in the model, and an output node is a node that is not a source node of any edge in the model.

We also refer to input and output nodes as “hanging” from the rest of the model, and they are often important for modeling outcomes: input nodes are used as pathway catalysts, and output nodes can represent model outcomes.

Definition 5. We define a path in a model as $n > 1$ connected edges: $p(v_{s_p}, v_{t_p}, a^{sign_p}) = (e(v_{s_p} = v_{k_1}, v_{k_2}, \mathbf{a}_{k_1}^e), e(v_{k_2}, v_{k_3}, \mathbf{a}_{k_2}^e), \dots, e(v_{k_n}, v_{k_{n+1}} = v_{t_p}, \mathbf{a}_{k_n}^e))$. The direction of the path is implicitly defined with the source node v_{s_p} and target node v_{t_p} . The regulation sign a^{sign_p} is considered positive when the number of negative signs in the set $\{a_{k_1}^{sign}, a_{k_2}^{sign}, \dots, a_{k_n}^{sign}\}$ is even, and negative when this number is odd. Cycles and feedback loops may be defined in cases where the path source is also the path target, i.e. $p(v_{s_p}, v_{s_p}, a^{sign_p})$.

In the case of Figure 2-2, we can observe the path from source node v_6 to target node v_{13} . This path is defined in such a way that the source node is not also an input node, it is in fact a section of a larger path starting from either input node v_1 or v_2 . In this case, the number of negative regulations is odd, there being one negative regulation from node v_8 to v_9 , and so the sign of this overall path is negative.

2.2 Approaches for Dynamical Modeling

We describe here two types of approaches for modeling dynamics in biological systems: event-based modeling and element-based modeling. We discuss several modeling approaches that fall within these types and examine whether the structure of built models can be represented using directed graphs. Event-based modeling [9-11] shows very clearly the processes driving system dynamics but is limited by the availability of certain event parameters. Element-based modeling can be accomplished with less complete information but can struggle to represent some of the mechanistic details. In Figure 2-3, we show how a set of example interactions could be represented across different modeling approaches within event-based and element-based methods. We begin by showing the example interactions, adapted from [12] and then how they would be represented as a set of rules. Next, we show the ODES for the system of reactions – these columns illustrate event-based methods described in Section 2.2.1. We then show how these reactions would be represented as a directed network, as described in Section 2.1.2, and the associate element rules. We show how this network would be represented in the BioRECIPES tabular format [13], and the element values would be updated using logical functions, discrete functions, and finally weighted functions. These later columns illustrate element-based methods described in Section 2.2.2.

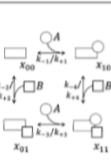
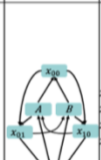
Event-based			Element-based																										
Reactions	Rules	ODEs	Graph	Element Update Rules	BioRECIPIES Format	Logical Update Functions	Discrete Update Functions	Weighted Functions																					
 <div>$\begin{aligned} x_{00} + A &\leftrightarrow x_{10} & k_{+1} & k_{-1} \\ x_{00} + B &\leftrightarrow x_{10} & k_{+2} & k_{-2} \\ x_{01} + A &\leftrightarrow x_{11} & k_{+3} & k_{-3} \\ x_{01} + B &\leftrightarrow x_{11} & k_{+4} & k_{-4} \end{aligned}$</div>	<div>$\begin{aligned} x_{00} + A &\leftrightarrow x_{10} & k_{+1} & k_{-1} \\ x_{00} + B &\leftrightarrow x_{10} & k_{+2} & k_{-2} \\ x_{01} + A &\leftrightarrow x_{11} & k_{+3} & k_{-3} \\ x_{01} + B &\leftrightarrow x_{11} & k_{+4} & k_{-4} \end{aligned}$</div>	<div>$\begin{aligned} \frac{d[x_{00}]}{dt} &= -k_{+1}[A][x_{00}] + k_{-1}[x_{10}] - k_{+2}[B][x_{00}] + k_{-2}[x_{10}] \\ \frac{d[x_{10}]}{dt} &= +k_{+1}[A][x_{00}] - k_{-1}[x_{10}] - k_{+2}[B][x_{00}] + k_{-2}[x_{10}] \\ \frac{d[x_{01}]}{dt} &= k_{+3}[A][x_{01}] + k_{-3}[x_{11}] + k_{+4}[B][x_{01}] - k_{-4}[x_{11}] \\ \frac{d[x_{11}]}{dt} &= -k_{+3}[A][x_{01}] - k_{-3}[x_{11}] - k_{+4}[B][x_{01}] + k_{-4}[x_{11}] \end{aligned}$</div>		<div>$\begin{aligned} x_{00,next} &= f(A, B, x_{10}, x_{01}) \\ x_{10,next} &= f(x_{00}, A, B, x_{11}) \\ x_{01,next} &= f(x_{00}, A, B, x_{11}) \\ x_{11,next} &= f(x_{01}, A, x_{10}, B) \\ A_{next} &= f(x_{00}, x_{10}, x_{01}, x_{11}) \\ B_{next} &= f(x_{00}, x_{10}, x_{01}, x_{11}) \end{aligned}$</div>	<table><thead><tr><th>Element</th><th>Positive Regulators</th><th>Negative Regulators</th></tr></thead><tbody><tr><td>A</td><td>x_{10}, x_{11}</td><td>x_{00}, x_{01}</td></tr><tr><td>B</td><td>x_{01}, x_{11}</td><td>x_{00}, x_{10}</td></tr><tr><td>x_{00}</td><td>x_{10}, x_{01}</td><td>A, B</td></tr><tr><td>x_{10}</td><td>A, x_{00}, x_{11}</td><td>B</td></tr><tr><td>x_{01}</td><td>B, x_{00}, x_{11}</td><td>A</td></tr><tr><td>x_{11}</td><td>A, B, x_{01}, x_{10}</td><td></td></tr></tbody></table>	Element	Positive Regulators	Negative Regulators	A	x_{10}, x_{11}	x_{00}, x_{01}	B	x_{01}, x_{11}	x_{00}, x_{10}	x_{00}	x_{10}, x_{01}	A, B	x_{10}	A, x_{00}, x_{11}	B	x_{01}	B, x_{00}, x_{11}	A	x_{11}	A, B, x_{01}, x_{10}		<div>$\begin{aligned} f_{x_{00}} &= x_{01} \text{ OR } x_{10} \text{ OR NOT } A \text{ OR NOT } B \\ f_{x_{10}} &= (x_{00} \text{ AND } A) \text{ OR NOT } B \text{ OR } x_{11} \\ f_{x_{01}} &= (x_{00} \text{ AND } B) \text{ OR NOT } A \text{ OR } x_{11} \\ f_{x_{11}} &= (x_{01} \text{ AND } A) \text{ OR } (x_{10} \text{ AND } B) \\ f_A &= \text{NOT } x_{00} \text{ OR } x_{10} \text{ OR NOT } x_{01} \text{ OR } x_{11} \\ f_B &= \text{NOT } x_{00} \text{ OR } x_{01} \text{ OR NOT } x_{10} \text{ OR } x_{11} \end{aligned}$</div>	<div>$\begin{aligned} f_{x_{00}} &= (x_{01} + x_{10}) - (A + B) \\ f_{x_{10}} &= (x_{00} * A) + x_{11} - 3 * B \\ f_{x_{01}} &= (x_{00} * B) + x_{11} - 2 * A \\ f_{x_{11}} &= (x_{01} * A) + (x_{10} * B) \\ f_A &= (x_{10} * x_{11}) - (x_{00} + x_{01}) \\ f_B &= (x_{01} + x_{11}) - (x_{00} + x_{10}) \end{aligned}$</div>	<div>$\begin{aligned} f_{x_{00}} &= (x_{01} + x_{10}) - (2 * A + 3 * B) + x_{11} - 3 * B \\ f_{x_{10}} &= (x_{00} * 2 * A) + x_{11} - 3 * B \\ f_{x_{01}} &= (x_{00} * 3 * B) + x_{11} - 2 * A \\ f_{x_{11}} &= (x_{01} * 2 * A) + (x_{10} * 3 * B) \\ f_A &= (x_{10} + x_{11}) - (x_{00} + x_{01}) \\ f_B &= (x_{01} + x_{11}) - (x_{00} + x_{10}) \end{aligned}$</div>
Element	Positive Regulators	Negative Regulators																											
A	x_{10}, x_{11}	x_{00}, x_{01}																											
B	x_{01}, x_{11}	x_{00}, x_{10}																											
x_{00}	x_{10}, x_{01}	A, B																											
x_{10}	A, x_{00}, x_{11}	B																											
x_{01}	B, x_{00}, x_{11}	A																											
x_{11}	A, B, x_{01}, x_{10}																												

Figure 2-3. Biological reaction representation in different modeling approaches.

The example reactions presented in the first column were adapted from [reference], and then the resultant reaction rules follow in the second column. Column three shows the ODEs that would result from these reactions, with the first three columns together illustrating the event-based approach. In the fourth column, we show how these reactions could be represented in a directed graph, with associated element update rules shown in column 5. Column six shows how the graph could be represented in the BioRECIPIES model format, and columns seven, eight, and nine show how the events in the graph could be represented with various updated functions used in element-based methods.

2.2.1 Event-based modeling

In event-based modeling of biological systems, biochemical reactions are modeled as events, while elements that participate in the event represent the entire populations of species participating in reactions. Therefore, a model is a collection of different events, while the same element may occur in multiple events. In the event-based columns of Figure 2-3, the outcome (creation of x_{10}) is defined by the reaction, or event, occurring between x_{00} and A, as well as the forward reaction rate, k_{+1} .

Event-based models can be analyzed in different ways, for example, using event-based simulations, or translating them into a set of ordinary differential equations (ODEs) [14, 15]. Most

often, a biochemical reaction network is represented by a set of ODEs, such that for each species v_i participating in reactions, we can express the changes in its concentration $[v_i]$ as:

$$\frac{d[v_i]}{dt} = f(v_1, v_2, \dots, v_n)$$

where v_1, v_2, \dots, v_n are all the species participating in the reaction network. The function f can be something like Michaelis-Menten mechanics:

$$\frac{d[v_1]}{dt} = k_{1,s} \frac{k_{1,2}[v_2]}{1 + k_{1,2}[v_2]} - k_{1,d}[v_1]$$

where v_1 is regulated by v_2 at a rate of $k_{1,2}$, and the species v_1 is synthesized by the system at a rate of $k_{1,s}$ and degrades at a rate of $k_{1,d}$. ODEs can capture or hold information about the connectivity and dynamics and enhance mechanistic understanding [16]. Many biological processes can be modeled by such functions, with the reaction rate parameters often coming from experimental data [8].

One of the strengths of event-based models is their ability to capture concentrations and dynamic behavior in a way that can easily be compared to experimental data [17-19]. They can also show the overall changes of the system without requiring focus on the specific behavior of individual entities.

The reaction rule-based modeling (RRBM) approach, also often referred to in literature as just “rule-based modeling” [20, 21], has been introduced as another type of event-based modeling to avoid the necessity for explicitly writing the entire network of reactions. In RRBM, only the center of reaction that is relevant for the interaction occurrence is included in the rule, while any other reaction context is not captured within the rule. Thus, RRBM does not require knowledge of all possible network species [22] and can address the complexity of protein interactions; proteins can interact with multiple other proteins and at multiple interaction sites, resulting in slightly

different reactions with the same reaction center being represented with a single reaction rule [20]. In other words, reaction rules allow for compact representation of reaction networks, while focusing on the elements of most interest [23].

Once reaction rules are defined, similar to individual reactions, rates are assigned to the rules. The rates can be either constants or user-defined functions [21, 22]. For example, Marchisio et al. used RRBMs to create a computational representation of eukaryotic gene circuits [23]. Different methods exist for studying the dynamics using RRBMs. Often, these models are analyzed by first generating the entire reaction network from them, and then either translating the network into a set of ODEs and solving the ODEs, or using the stochastic simulation algorithm (SSA) [24-27]. There are also network-free approaches that simulate RRBMs without generating the entire reaction network [28].

However, the methods of event-based modeling often require that parameters of reactions are well known [14, 20, 29]. For poorly understood systems, this is a disadvantage for effective modeling. Certain systems also require more sophisticated functions, requiring expert knowledge of mathematical techniques [14]. It is also often necessary to apply proper constraints to the system for the results to be accurate [16], for example the time scale.

While the network structure of event-based models such as reaction networks and RRBMs is a directed network, the translation to the network defined in Section 2.1.2 may not always be straightforward. The same reaction mechanism can be translated to different influence polarities, for example, phosphorylation is usually an activating event, although in some cases it may be considered an inhibiting event.

2.2.2 Element-based Modeling

In element-based modeling of biological systems [30-33], an element represents a biomolecular species (and in some cases even a biological process) and all reactions that lead to changes in a single species are lumped together within the corresponding element's update rule. Therefore, a model is a collection of element update rules, each update rule is associated with a different model element, which can be more formally defined as follows.

Definition 6. An element-based model is a triple $\mathcal{M}(G, \mathcal{X}, \mathcal{F})$, where $G(V, E)$ is a network structure of the model (defined earlier in Definition 3), $\mathcal{X} = \{x_1, x_2, \dots, x_N\}$ is a set of N state variables corresponding to nodes in $V = \{v_1, v_2, \dots, v_N\}$, and $\mathcal{F} = \{f_1, f_2, \dots, f_N\}$ is a set of N regulatory (update) functions such that each element $v_i \in V$ has a corresponding function $f_i \in \mathcal{F}$.

Definition 7. For each element $v_i \in V$, its state variable $x_i \in \mathcal{X}$ can take any value from a set or interval of values X_i .

Definition 8. For each element v_i , any element v_j that influences the state of v_i such that the function f_i is sensitive to the value of x_j is called a regulator of v_i .

Definition 9. For each element v_i , an influence set, denoted as $V_i^{influence} \subseteq V$, consists of all regulators of v_i . We will refer to the set of state variables that correspond to the elements in $V_i^{influence}$ as $\mathcal{X}_i^{influence}$.

Definition 10. The next state of element v_i , denoted as x_i^* , is computed given current states of all elements in its influence set, that is, given values of all variables in $\mathcal{X}_i^{influence}$:

$$x_i^* = f_i(\mathcal{X}_i^{influence}).$$

In general, functions in \mathcal{F} can have different types, discrete or continuous, and moreover, individual elements within the same model could have very different update functions, thus forming hybrid models. The set or interval of possible values, X_i , assigned to each model element x_i can also vary. The function and element types are usually decided based on the knowledge or the information available about the modeled system and its components. In the element-based columns of Figure 2-3, we illustrate how interaction events would be represented in an element-based model, showing the directed graph, as well as the element rules, tabular representation format [13], and possible update rules.

The element-based modeling approach can represent indirect influences between elements, and it can model systems where the knowledge about element interaction mechanisms is incomplete. Using element update rules in simulations allows for studies of cell dynamics, state transitions, and feedback loops [34, 35], and does not require full knowledge of the interaction mechanisms [36]. Element-based models can also allow for integration of both prior knowledge and data [13, 37, 38] and analysis of hybrid networks (systems involving protein-protein interactions, gene regulations, and/or metabolic pathways) [39].

An example of element-based models are discrete models, where each element state variable x_i is assigned a discrete set of values [38]. Following Definition 7, $x_i \in X_i: \{0, 1, 2, \dots, n_i - 1\}$, where n_i is the number of different states that element, v_i can have. Often, these different states represent different levels of activity or concentration for element v_i . Element update functions in

discrete models can be of different type, some examples are *min* and *max* functions, and (rounded) weighted sums.

Boolean models are a subset of discrete models [29, 30, 40], where elements can have only two values, 0 (also referred to as OFF or False) or 1 (also referred to as ON or True). In Boolean models, value 0 represents states such as “inactive”, “absent”, or “low concentration” and value 1 represents states such as “active”, “present”, or “high concentration”. Element update functions in these models are Boolean functions where logic operators such as AND, OR, and NOT are used [31]. As a subtype of Boolean networks, the Probabilistic Boolean Network (PBN), randomness is introduced by assigning multiple candidate Boolean functions to the variables. At each time step during simulation, one of element’s candidate functions is chosen at random to determine its state [41, 42].

Other examples of commonly used element-based models are Bayesian Networks [35, 43] and Dynamic Bayesian Networks [34]. Bayesian networks introduce probability distributions into the governing rules of elements, increasing the freedom in updating element states. Similar to Bayesian Networks are structural equation models (SEMs) [44]. SEMs are used to measure and analyze the relationships of observed, latent variables [45], and they have the ability to manage measurement error. In a set of relationships between variables, path coefficients are determined by methods developed by Wright [46], measuring the importance of the input of a given path to its output. The strength of SEM lies in analyzing these variable relationships, making it an excellent choice for disease studies.

Given that the element-based modeling approach can be used for indirect influences and it can abstract away from detailed reaction mechanisms, additional methods have been introduced to

account for the timing in biological systems, rates at which elements change, or delays in element updating and delays in pathways [38].

While the network structure defined in Section 2.1.2 may not always exactly match the structure of event-based models described in Section 2.2.1, element-based models do have such a structure and add element update rules to allow for studying the dynamics. Within the update rules of element-based models, the attribute a_e^{sign} is explicit, though it is not always explicitly stated in statements about influences that describe mechanisms (e.g., A phosphorylates B). In this case, it would be up to the user to either fill in this information from other sources or accept a default attribute assignment.

2.3 Knowledge Sources

Networks and models assembled from knowledge are typically built manually, using literature, interaction databases, model databases, conclusions from experiments and expert input. Additionally, networks can be inferred from data. Networks inferred from data are usually not directed and only represent correlations found in data, although recent efforts focus on inferring causal relationships from data. We discuss in this section the knowledge and data sources that can be used when creating inputs for VIOLIN.

2.3.1 Interaction Databases

The interaction databases contain information usually collected from the literature, dedicated collaborations of experts, experimental data, and general user input. Some databases

[47-51] include information from multiple molecule types, with curated biological interactions that belong to one or more of the following categories: protein-protein interactions (PPIs), protein-gene interactions (PGIs), protein-chemical interactions (PCIs), or proteins interacting with a biological process cascade (PBIs). Others may specialize in one specific type of interaction, such as the human protein reference database (HPRD) [52, 53] (PPIs), STRING [54-56] (PPIs), and STITCH [57] (PCIs).

The interaction databases are excellent, comprehensive sources of information, and often verified and curated by experts. However, this inclusive availability of data can mean a user must take the time to search through large collections of knowledge. The SIGNOR database [48] hosts over 30,000 interactions, STITCH [57] over 1.6 billion, STRING [54-56] over 20 billion, Pathway Commons [49] over 2 million interactions. IntAct [51] hosts over 5 million interactions. To make best use of these databases, a user must have well defined queries to find the relevant information, knowing a great deal about the system of interest. Table 2-1 shows a list of databases and their relevant characteristics.

Table 2-1. Summary of Interaction Databases

Database	Curation Method	Number of Entries	Formats	Grounding	Link
BioGRID [47]	Manual and automated	3+ million PPIs, PGIs, and PCIs	N/A	GO, HGNC, UniProt, HPRD	thebiogrid.org
Gene Ontology (GO) Database [7]	Manual and automated	43850 terms, 7.9 million annotations, 1.5 million gene products	N/A	N/A	geneontology.org

Table 2-1. (continued)

Database	Curation Method	Number of Entries	Formats	Grounding	Link
HPRD [52, 53]	Manual, staff curators	40000+ PPI, 36 pathways	BioPAX, SBML, PSI-MI	RefSeq, GenBank, OMIM, SwissProt, etc	hprd.org
IntAct [51]	Manual, staff curators	5.5 million PPIs, PGIs, PCIs	N/A	GO, UniProt, ChEBI	ebi.ac.uk/intact/
Open PHACTS [58]	Manual, staff curators	Unknown, private	Unknown, private	Unknown	openphacts.org
Pathway Commons [49]	Manual, data providers	5772 pathways, 2.4 million PPIs, PGIs, PCIs	GMT, SIF, SBGN, BioPAX	GO, UniProt, ChEBI	pathwaycommons.org
Reactome [50, 59]	Manual, staff curators	13827 PPIs, PGIs, and PCIs, 2536 pathways	PDF, SBML, SBGN, BioPAX, OWL, Directed network graph (PNG)	GO, UniProt, ChEBI	reactome.org
SIGNOR [48]	Manual, staff curators	29245 PPIs and PCIs	N/A	GO, UniProt, ChEBI	signor.uniroma2.it
STITCH [57]	Manual and automated	1.6 billion PCIs	N/A	Ensembl	stitch.embl.de
STRING [54-56]	Manual and automated	20+ billion PPIs	N/A	GO, PubMed, UniProt, Pfam, InterPro	string-db.org

2.3.2 Network and Model Databases

There exists a number of open source network and model databases which are user curated, and models are sourced from existing publications in the literature or ongoing explorations. In addition to individual interactions, Pathway Commons host over 5,000 pathway networks. Large databases such as KEGG [60], PCNet [61], BioModels [62-64], MINT [65], and Open PHACTS [58] are manually or semi-manually (meaning there are some automated functions) maintained and require large amounts of human input to be kept valid and useful. Some databases such as the Pathway Coexpression Network [66], OmniPath [67], and Path2Models [68] collect information from other databases, not introducing any new information, but making it easier to search across multiple knowledge sources from a single database.

Databases such as NDEx [69], Wikipathways [70], Reactome [50], Cell Collective [71], host user created models, or in the case of Cell Collective, allow users to assemble models from database information. The benefit of these databases is they host models in multiple formats, and at varying levels of detail. However, not all models hosted on these databases have been validated or peer-reviewed. Using an unvalidated model requires extra steps to confirm the veracity of the model results. Table 2-2 shows a list of network and model databases. Some databases allow for the uploading and downloading of system networks in specified formats, some databases present information as well-defined textual evidence which can be copied to a model or interaction set.

Table 2-2. Summary of Network and Model Databases

Database	Curation Method	Number of Entries	Formats	Grounding	Link
Bio2RDF [72]	Automated, from existing databases	35 datasets	RDF	N/A	bio2rdf.org
BioModels [62-64]	Manual, registered users	2647 models	SBML, MATLAB, MorpheusML, COMBINE, etc	GO, DOID, UniProt, etc	www.ebi.ac.uk/biomodels
Cell Collective [71]	Manual, registered users	28 public modules	SBML, Boolean expressions, Truth tables	Unknown	cellcollective.org
CoLoMoTo [73, 74]	Manual, consortium of experts, from existing databases	From BioModels, Cell Collective, GINsim, and PyBoolNet	SBML, Boolsim, XML, GNA	N/A	colomoto.org
HPRD [52, 53]	Manual, staff curators	40000+ PPI, 36 pathways	BioPAX, SBML, PSI-MI	RefSeq, GenBank, OMIM, SwissProt, etc	hprd.org

Table 2-2. (continued)

Database	Curation Method	Number of Entries	Formats	Grounding	Link
KEGG [60]	Manual, staff curators and data providers	59 pathways	Directed network graph, KGML	RefSeq, Pfam, NCBI Gene ID, HGNC, Ensembl, UniProt	kegg.jp
MINT [65]	Manual, registered users	90 models	Topoflow, JSON, etc.	N/A	mint.bio.uniroma2.it
NDEx [69]	Manual, registered users	5000+ networks	CX (cytoscape)	GO, UniProt, PMCID	ndexbio.org
OmniPath [67]	Automatics, from existing databases	Collects data from 100+ resources	R, Python, JSON	N/A	https://omnipathdb.org/
Path2Models [68]	Automated, from existing databases	~140000 models	SBML	GO, UniProt, ChEBI, PMCID	ebi.ac.uk/biomodels/path2models
Pathway Coexpression Network [66]	Automated, from existing databases	1330 pathways	Directed network graph, JSON	NCBI Gene ID	pcxn.org

Table 2-2 (continued)

Database	Curation Method	Number of Entries	Formats	Grounding	Link
Pathway Commons [49]	Manual, data providers	5772 pathways, 2.4 million PPIs, PGIs, PCIs	GMT, SIF, SBGN, BioPAX	GO, UniProt, ChEBI	pathwaycommons.org
PCNet [61]	Manual, staff curators, pulls from existing knowledge databases	21 human genome-wide interaction networks	Various	Various	Publication Link
Reactome [50, 59]	Manual, staff curators	13827 PPIs, PGIs, and PCIs, 2536 pathways	PDF, SBML, SBGN, BioPAX, OWL, Directed network graph (PNG)	GO, UniProt, ChEBI	reactome.org
Wikipathways [70]	Manual, registered users	1100+ pathways	BEL, SBML, CSV (interaction set)	Ensembl, GO, RefSeq, UniProt, WikiGenes	wikipathways.org

2.3.3 Literature

One of the biggest sources of knowledge, is of course, the published literature. Users can source information from textbooks, reviews, and articles from various sources. However, the tradeoff for this is often a large time expenditure. Some of the tools described in Section 2.5 can speed this process by automatically mining information, but these tools are often limited in their own way. These tools either require the user to manually enter the text or data for analysis [75], or they can only access literature which is available on open-source databases like PubMed [8], and recent enough to be fully digitized [76].

2.4 Network Representation Formats

Many of the tools for creating causal networks and databases hosting such networks generate output in or are compatible with the following representation formats: SBML [77], SBML-qual [78], BEL [79], BioPAX [80], and BNGL [22] BioRECIPES [13]. Figure 2-4 shows example models in the SBML, BNGL, BEL, and BioPAX formats.

(A)

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Created by COPASI version 4.24 (Build 197) on 2019-11-27 13:11 with 1
<sbml xmlns="http://www.sbml.org/sbml/level2/version4" level="2" version="
<model metaid="COPASI0" id="Yu2019__A_mathematical_model_of_tumor_immur
<notes>
<body xmlns="http://www.w3.org/1999/xhtml">
<pre>This is a mathematical model investigating interactions among
</body>
</notes>
<annotation>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xml
<rdf:Description rdf:about="#COPASI0">
<dc:creator>
<rdf:Bag>
<rdf:li rdf:parseType="Resource">
<vCard:N rdf:parseType="Resource">
```

(B)

```
d_ma 10 # mA degradation
d_mr 0.5 # mR degradation
d_a 1 # degradation of A
d_r 0.2 # degradation of R

kf 1 # Binding of A to activator or repressor promoter
kr1 50 # Different reversal rates for unbinding from act:
kr2 100
k3 2 # Binding of A to R to form complex

end parameters

begin molecule types
D(p,r) # DNA with both an activator promoter (p) region and
mA # Activator mRNA
A(d,r) # Activator protein with sites able to bind r
mR # Repressor mRNA
R(a) # Repressor protein with a site able to bind activato:
```

(C)

```
bp(GO:"response to fluid shear stress") -> r(HGNC:NOS3)
bp(GO:"response to fluid shear stress") -> act(p(HGNC:NOS3), ma(cat))
act(p(HGNC:NOS3), ma(cat)) => a(CHEBI:"nitric oxide")
a(CHEBI:"nitric oxide") -| bp(GO:"apoptotic process")
```

(D)

```
version="1.0" encoding="UTF-8"?>
<rdf:RDF
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:bp="http://www.biopax.org/release/biopax-level3.owl#" xml:base=""
<owl:Ontology rdf:about="">
<owl:imports rdf:resource="http://www.biopax.org/release/biopax-level3.
</owl:Ontology>
<bp:Protein rdf:about="http://my.example.com/biopax>HelloWorld">
<bp:displayName rdf:datatype="http://www.w3.org/2001/XMLSchema#stri
</bp:Protein>
</rdf:RDF>
```

Figure 2-4. Examples of four common representation formats.(A) SBML model [81], (B) BNGL model [82], (C) BEL assertions [79, 83], and (D) example of BioPAX representation [80].

The BioRECIPES format [13] was developed to represent this information, be readable by a model simulator, and also easily readable by a human user. An example of a BioRECIPES format model is shown in Figure 2-5. The format is very similar to machine reading output spreadsheets, as shown in Section 2.5.2. However, due to the specificity of this representation format, it sometimes requires transformation to be compatible with other modeling tools.

Element Name	Element Type	Element IDs	Cell line	Cell type	Organism	Tissue type	Location	Location ID	Variable	Positive Regulators	Negative Regulators
ADAM	Protein family	P78536,Q14672	Pancreatic	Pancreas	Human	Pancreas	plasma mem	GO:0005886	ADAMpf_memPCC	PKCApf_cytoPCC,SRGpn_cytoPCC	
AKT	Protein family	P31749,P31751	Pancreatic	Pancreas	Human	Pancreas	cytoplasm	GO:0005737	AKTpf_cytoPCC	PDPK1pn_cytoPCC	ICMTpn_erPCC,PP2Apf_cytoPCC,
AMPK	Protein family	Q13131,P54646,Q9Y478,O4	Pancreatic	Pancreas	Human	Pancreas	cytoplasm	GO:0005737	AMPKpf_cytoPCC	LKB1pn_cytoPCC	
AP1	Protein complex	P05412,P01100	Pancreatic	Pancreas	Human	Pancreas	nucleus	GO:0005634	AP1pf_nucPCC	(FOSpn_nucPCC,INRpf_nucPCC)	
ARF	Protein	Q8N726	Pancreatic	Pancreas	Human	Pancreas	nucleus	GO:0005634	ARFpn_nucPCC	E2Fpf_nucPCC	
ASK	Protein family	Q99683	Pancreatic	Pancreas	Human	Pancreas	cytoplasm	GO:0005737	ASKpf_cytoPCC	THIOREDOXINpn_cytoPCC,TRAFpf_cytoPCC	
ATG4A	Protein family	Q8WYN0,Q9Y4P1,Q96DT6	Pancreatic	Pancreas	Human	Pancreas	cytoplasm	GO:0005737	ATG4Apf_cytoPCC	ATG4Ama_nucPCC	
ATG4A	mRNA	Q8WYN0,Q9Y4P1,Q96DT6	Pancreatic	Pancreas	Human	Pancreas	nucleus	GO:0005634	ATG4Ama_nucPCC	ATG4Ama_nucPCC	

Figure 2-5. A model in the BioRECIPES representation format

Systems Biology Markup Language (SBML) [77] is often considered the community standard, compatible with many of the tools and databases listed in the Sections 2.3.1 and 2.3.2 [2,

50, 63]. It is a freely available, XML-based, software-independent format for representing biochemical reaction networks. An example is shown in Figure 2-4A. SBML was developed to help standardize the representation of models, making them more accessible across modeling, simulation, and analysis tools [78]. And while SBML was originally developed with event-based modeling in mind, much like BioNetGen Language (BNGL) shown in Figure 2-4B [22], an extension, SBML-*qual*, allows for further compatibility with logical models [78]. The largest challenge with SBML and SBML-*qual* is its readability – the SBML language generally requires expert knowledge of the language, and is not intuitive for the user to read, as it is meant to be read by computation tools.

The Biological Expression Language (BEL) [79] was developed to represent causal relationships between entities. An example of BEL such relationships, called BEL assertions, is shown in Figure 2-4C. It is most often compared to the Biological Pathway Exchange (BioPAX) language shown in Figure 2-4D, which was developed to enable visualization across biological pathways [80]. So where SBML aims to capture the dynamics of a biological pathway, BEL and BioPAX focus more on the causal relationships [84].

Another example of an effort to address the formatting disparity and standardize representation, in particular in the field of logical modeling, is The Consortium for Logical Models and Tools (CoLoMoTo) [73]. However, such goals usually require a great amount of effort, not only on part of the consortium, but also on part modeling and database developers.

2.5 Tools

To facilitate model building, several tools have been developed to speed up or automate knowledge retrieval, filtering and curation, as well as network assembly and model recommendation.

2.5.1 Directed Network Inference Tools

With the volume of experimental data available, some tools have taken the approach of building directed networks from data. The TETRAD project [85] seeks to use mathematically sound statistical standards to improve the process by which experimental data and background knowledge is used to build computational models. This work identifies the struggles with building a causal network from data which can be open to interpretation, or has multiple possible causal relationships. A review by Glymour et al. [86] showed that while these tools can speed the process by which data is used to build a directed network, their efficacy can be dependent on the modeling case [87, 88].

Additionally, there are programming packages to aid in creating networks. The `pcalg` R package is capable of causal structure learning and estimation of causal effects from observed data [89]. This package makes use of several algorithms [90-92], with the benefit of analyzing the size of causal effects, using the IDA method from Maathuis et al. [93] to create bounds based on the data. However, this package assumes the network has no feedback loops, and that the causal structure can be represented by a directed, acyclic graph [89]. The `bnlearn` R package contains algorithms for learning the structure of Bayesian networks [94]. This package uses constraint- and scored-based algorithms, and is able to analyze networks with either discrete or continuous

variables. The blearn package has the advantage of using permutation tests in its network learning algorithm, improving the experimental data analysis efficacy [94].

2.5.2 Machine Reading Tools

To aid in the creation of computational models, one can make use of machine reading engines, which can identify and extract interactions from published papers selected via specified queries [76, 95-98]. We call these interactions Literature Extracted Events, or LEEs, and a set of LEEs form a collection of nodes and edges very similar to a model, the main difference is that LEEs in a set may or may not form a fully cohesive network. The benefit of machine readers is that they are able to extract vast amounts of information very quickly. However, because of the variability and sheer volume of language, machine readers can encounter difficulties in correctly identifying events.

2.5.2.1 Functionality

Machine reading tools function by automatically parsing through published papers or user-input text and identifying interactions, compiling them into output for use. The tool REACH [76] is able to automatically query for papers on the PubMed Central [8] database and identify biological interactions from selected papers. These interactions are assembled into LEEs for output. The TRIPS machine reader [75] takes sentences or paragraphs from the user, and identifies the interaction or interactions represented in the text, though TRIPS does not assemble the identified interactions into LEE output [75]. Eidos [96] and PyBEL [97] are additional tools for extracting events from text, but these tools are specific to world-modeling systems and Biological Expression Language (BEL) documents [1], respectively. RUBICON [1, 98] is another machine

reader that can extract from biological literature, and also calculates an Epistemic Value for each LEE, based on recommendations by de Waard et al. [99]. The Epistemic Value assigns one of four numeric values, representing complete uncertainty, low certainty, high-likelihood, or complete certainty [100, 101], based on supporting text. Terms such as “unknown” or “unclear” suggest low certainty in the assertion in the rest of the evidence, whereas phrases like “this suggest...” or “the authors report...” can indicate possible knowledge, but it cannot yet be taken as absolute fact [99].

The tool INDRA (Integrated Network and Dynamical Reason Assembler) [2] is not a machine reader itself, but it can call on the machine readers REACH, TRIPS, and the BEL Large Corpus to assemble sets of LEEs. INDRA also calculates a “belief score” for each LEE, based on known success rates of each machine reader, and of the number of times an LEE has been extracted from the literature, as documented in a larger knowledge base [2].

2.5.2.2 Output

Some readers output into JSON format [102], which, while not easily readable, is readily enough converted into a spreadsheet with a simple script. Some (like TRIPS [95]) will only output lists. Assemblers like INDRA [2] can pull from multiple readers and compile them in a JSON format, which can then further be transformed into a spreadsheet for any manual human input. Figure 2-6A shows the information commonly included in the output of several machine readers. Information such as the node names and IDs, and edge mechanisms are easy to extract, but the finer details such as the cellular location and cell type are often not explicitly stated in the text, and therefore not always available in machine reader output. INDRA, which does not perform machine reading directly, but instead calls upon existing machine readers, converts machine reading output into INDRA statements, which cannot always capture the same information as the machine reader.

Definition 11. The output of machine readers, which is created when processing a set of papers, is a set of events, where each event is an interaction or a causal relationship between entities. We refer to the interactions identified by machine reading tools as literature extracted events, or LEEs.

Definition 12. A set of LEEs can form a graph, $G^{LEE}(V^{LEE}, E^{LEE})$, where $V^{LEE} = \{v'_1, v'_2, \dots, v'_{N'}\}$ is a set of nodes, and each node $v'_i = v'(\mathbf{a}_i^{v'})$, $i = 1, \dots, N'$ is an LEE element, while $E^{LEE} = \{e'_1, e'_2, \dots, e'_{M'}\}$ is a set of directed edges, and each edge $e'_j = e'(v'_{s_j}, v'_{t_j}, \mathbf{a}_j^{e'})$ ($v'_{s_j}, v'_{t_j} \in V^{LEE}$, $j = 1, \dots, M'$) corresponds to a directed interaction from one LEE, connecting its regulator (source node v'_{s_j}) and regulated (target node v'_{t_j}) elements.

Definition 13. Similar to Definitions 1-3 in Sections 2.1.1 and 2.1.2, we define vectors of node attributes in the LEE graph as

$$\mathbf{a}^{v'} \equiv (a^{name'}, a^{type'}, a^{ID'}, a^{location'}, a^{locationID'})$$

and edge attributes as

$$\mathbf{a}^{e'} \equiv (a^{sign'}, a^{connectiontype'}, a^{mechanism'}, a^{cellline'}, a^{celltype'}, a^{organism'}, a^{tissuetype'}, a^{evidence'}).$$

The attributes listed above are based on the common attributes of biological interactions, as described in Section 2.1.1, but those available from machine readers vary. LEEs obtained from the state-of-the-art machine readers often do not include information about $a^{location'}$, $a^{locationID'}$,

$a^{cellline'}$, $a^{celltype'}$, $a^{organism'}$, and $a^{tissuetype'}$, and in such cases an “empty” value is assumed for these attributes. The interaction sign attribute ($a^{sign'}$) is usually implicitly included in the tabular machine reading output with separate columns for positive and negative regulators (Figure 2-6B). The machine reading engines assign “direct” to the $a^{connectiontype'}$ attribute when they find in the text that an interaction within LEE represents direct physical interaction, most often with known mechanism (e.g., CK1 to APC), Figure 2-6A. The types of mechanisms that can be assigned to the $a^{mechanism'}$ attribute by readers are listed in Figure 2-6A. On the other hand, “indirect” is assigned to the $a^{connectiontype'}$ attribute when the machine reader does not find any explicit evidence that the interaction is direct. In Figure 2-6A, right, we show that the reader finds the interaction “Akt inhibits GSK-3beta” and identifies the interaction as indirect. The $a^{evidence'}$ attribute includes an ID of the paper (usually, PMCID [8]) where the LEE is mentioned, and the sentence in the paper from which it is extracted.

(A)

Interaction attribute list	REACH	TRIPS	INDRA	LEE example 1	LEE example 2	VIOLIN attributes
Source Name				CK1	Akt	α_{name}
Source ID				P04264	AKT	α_{ID}
Source Type				protein	protein family	α_{type}
Source Cellular Location				cytoplasm	-	$\alpha_{location}$
Source Cellular Location ID				GO: 0005737	-	$\alpha_{locationID}$
Target Name				APC	GSK3beta	α_{name}
Target ID				P25054	P49841	α_{ID}
Target Type				protein	protein	α_{type}
Target Cellular Location				cytoplasm	-	$\alpha_{location}$
Target Cellular Location ID				GO: 0005737	-	$\alpha_{locationID}$
Mutation				-	-	-
Interaction Residue				-	-	-
Residue Number				-	-	-
Mechanism				phosphorylation	-	$\alpha_{mechanism}$
Connection Type				direct	indirect	$\alpha_{connectiontype}$
Regulation Sign				positive	negative	α_{sign}
Cell line				-	-	$\alpha_{cellline}$
Cell Type				-	-	$\alpha_{celltype}$
Organism				-	-	$\alpha_{organism}$
Tissue Type				-	-	$\alpha_{tissuetype}$
Paper ID				PMC2654145	PMC1403772	$\alpha_{evidence}$
Evidence Text				*	#	$\alpha_{evidence}$
Epistemic value				-	-	-

Legend:

- available
- available, but difficult to extract
- unknown
- not available

Represented Mechanisms

Binding, Posttranslational Modifications, Transcription, Translocation, Amount Increase/Decrease
Binding, Phosphorylation, Dephosphorylation, Ubiquitination, Acetylation, Methylation, Transcription, Translocation, Activation, Inhibition, Increase or Decrease Amount, Promotes, Signaling, Reduce, Induce, Supports, Attenuates, Stimulate, Antagonize, Synergize, Abrogates

* This may be analogous to parallel mechanisms that promote GSK3 phosphorylation of beta-catenin in the absence of Wnt stimulation, such as by GSK3 and CK1 phosphorylation of Axin and APC

Therefore, given that Akt phosphorylates and inactivates GSK3beta, we hypothesized that Akt dependent inactivation of GSK3beta might be responsible for Notch potentiation

(B)

Element Name	Element Type	Element IDs	Cell Line	Cell Type	Organism	Location	Location ID	Variable	Positive Regulators	Negative Regulators
ERK	Protein Family	P27361, P28482	Pancreatic	Pancreas	Human	cytoplasm	GO: 0005737	ERKpf_cytoPCC	MEK1pf_cytoPCC	PP2Apf_cytoPCC
MDM2	Protein	Q00987	Pancreatic	Pancreas	Human	nucleus	GO: 0005634	MDM2pn_nucPCC	MDM2pn_cytoPCC	ARFpn_nucPCC
RPS6KB1	Protein Family	P23443, Q9UBS0	Pancreatic	Pancreas	Human	cytoplasm	GO: 0005737	RPS6KB1pf_cytoPCC	MTORC1pf_cytoPCC	

Figure 2-6. Information relating to machine reading output: (A) attributes included in the output of several machine reading engines, and the translation of evidence text [103, 104] to an LEE, along with the commonly supported interaction mechanisms. (B) example representation of biological interactions in a spreadsheet format.

2.5.2.3 Machine Reading Errors

While machine readers can speed the process of information curation, they are not infallible [105]. As described in [106], four types of errors are typically found in the machine reading output of biomedical literature, namely sign, direction, grounding, and omission errors. Sign errors occur when the machine reader assigns the wrong regulation sign to a source node. This is most common in phosphorylation interactions, as machine readers make the assumption that phosphorylation is always a positively regulating interaction. Direction errors occur when the machine reader identifies the regulating element as a target node, and the regulated element as a source node. Grounding errors occur when the machine reader does not assign the correct database ID to one or

both elements of an LEE. Finally, an omission error occurs when a sentence is misconstrued to represent an interaction which is not actually supported by the literature. The tool FLUTE (The FiLter for Understanding True Events) [106] has been developed to help address these errors, using the STRING [55], STITCH [57], and Gene Ontology [7] databases to identify high-confidence machine reading interactions.

2.5.3 Network Assembly Tools

Niarakis et al. identified best practices for model curation, and for the compatibility between multiple tools [74]. They also emphasized the importance of automating the integration of new information and making use of the vast quantity of interactions found in databases.

Tools like INDRA [2] help automate the model assembly process, by automatically curating information from relevant literature through the use of machine readers, which are described in Section 2.5.2. INDRA provides a database of statements written in a standardized format, with each statement containing information about a biomolecular interaction and a belief score, which is based on the interaction occurrence in literature and the probability of machine reading error. INDRA also has the functionality to extend existing models, so long as they are in an INDRA compatible format. The drawback is that the user must extend the model manually, curating the interactions into INDRA statements either manually defined or produced from machine reading output. This requires the user to query, judge, and choose information from the vast sources in the literature and databases.

Tools such as MINERVA [107], CARNIVAL [108], PathMe [109], VIPER [110], CLARINET [111], and Whistle [112] assemble models from existing knowledge databases such

as KEGG [60] and OmniPath [67] or from user-assembled interaction sets, but are limited only to the knowledge that exists in those sources. The BEL enrichment workflow [1] aims to automatically maintain knowledge graphs using literature, but still requires human input. Ostaszewski et al. worked to develop a knowledge base of disease maps [113], but their approach also required significant manual input from experts. Iyappan et al. developed a method for creating a pathway inventory for modeling disease mechanisms for the purpose of understanding Alzheimer’s disease [114]. Much like the databases described in Sections 2.3.1 and 2.3.2, this knowledge base must be updated frequently to stay relevant to the current understanding of the biology.

2.5.4 Model Recommendation Tools

In addition to network assembly, there are tools which make recommendations on executable model building and extension. The CaSQ tool [115] takes existing directed network graphs and transforming them into dynamic Boolean models to allow the investigation into dynamic properties. This tool does not add any outside information to an existing model, but increases the information which can be gleaned from it.

On the other hand, several methods have been proposed to iteratively extend executable models by selecting subsets of the available LEE set and testing whether the new model improves the performance when compared to the original one. The tool ACCORDION [116] automatically extends and recommends executable models by processing large LEE sets and identifying clusters of LEEs that are most relevant to a given baseline model. Another method developed by Liang et al. [117] uses a different approach, selecting only those interactions from the LEE set that are

directly connected to the model being extended. Finally, the method presented by Sayed et al. [118] uses genetic algorithm to select most useful subset of LEEs.

The DiSH simulator [38, 119] provides model analysis under different simulation schemes, and can generate input for, or validate output of, model recommendation tools, or it can be integrated as part of these tools (e.g., in ACCORDION). Another model analysis tool, model checking, is often used to evaluate how well the model satisfies the necessary system properties [118, 120]. For example, the model extension tools described above use model checking to find candidate LEEs that allow for satisfying desired system properties. In Chapter 6.0, we discuss how VIOLIN can help generate or focus the input for such tools.

The tools presented in this chapter aim to reduce the time and effort required to assemble information from research literature, however, it is up to the user to determine the relevance and usefulness of the extracted information. We present VIOLIN as a tool by which this process can be automated. VIOLIN is as a tool which can aid network assembly, both by reducing the need for manual input and by judging the relevance of potential additions to the network, irrespective of the source of the information. In addition to aiding the network and model building process, VIOLIN maintains the importance of feedback loops, which are sometimes excluded for efficacy [89] and pathway influences, which play a vital role in certain modeling approaches [45, 46].

3.0 VIOLIN Implementation

In this chapter we present the implementation of the VIOLIN tool. We define the classification categories and subcategories (Section 3.1), how the scores are calculated (Section 3.2), the algorithm VIOLIN uses to determine the classifications and scores (Section 3.3), necessary considerations for selecting VIOLIN input (Section 3.4), VIOLIN's compatibility with existing tools (Section 3.5), and the VIOLIN user interface (Section 3.6). The VIOLIN Framework is shown in Figure 3-1. First VIOLIN takes the set of LEEs and the baseline model as input. Next, information from the LEE set is used to compute the Evidence Score (detailed in Section 3.2.1) and identify the Epistemic value when available (detailed in Section 3.2.4).

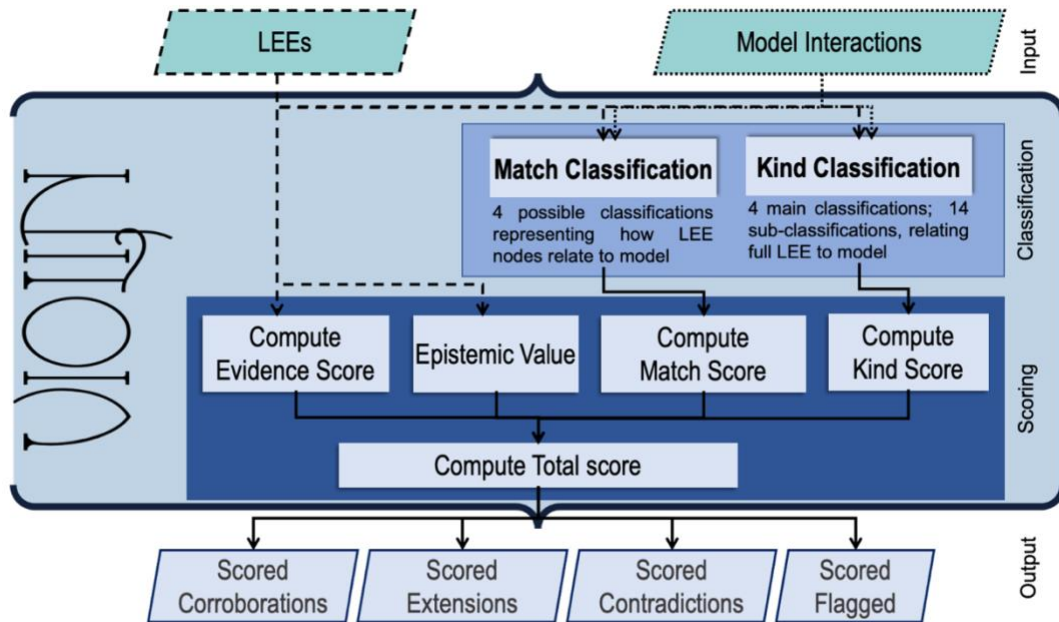


Figure 3-1. The internal framework of the VIOLIN tool. LEEs and a set of baseline model interactions are taken as input, and LEEs are given numerical values representing their different classifications, and these values are then culminated into a final score.

This also begins the classification process, comparing the nodes (Match) and edges and attributes (Kind) of the LEE set to the baseline model. From these classifications, numerical scores are assigned (described in Sections 3.2.2 and 3.2.3), and are then combined with the scores obtained from the LEE set information into a final Total Score, which represents the overall usefulness and relevance of a given LEE. The LEEs are then output based on their Total Scores and their main classification category (detailed in Sections 3.1.2-3.1.5) for use.

Definition 14. Within our approach, we define model extension as the process of changing an existing model towards the goal of improving its representation of a system. This can mean increasing the size of the system network, increasing the detail of the represented interactions, correcting outdated information, or validating modeling interactions using outside sources (such as literature, experimental data, or expert knowledge).

3.1 Classification Definitions

3.1.1 Match definitions

The basis for much of the judgement made by VIOLIN revolves around looking for matches between the LEEs and the baseline model. The Total Score assigned to each LEE is built from these matches, considering a rough comparison between the LEEs and the model network, as represented by the Match Score, and then a finer comparison between interaction details (or attributes), as represented by the Kind Score. Matching information serves to allow for comparing literature information to the baseline model, whereas mismatched information can provide insight

into how the baseline model may be extended or identify reading errors, depending on the nature of the mismatch.

Definition 15. The necessary element match condition for an LEE element v' to match a baseline model element v is that v' and v have the same type and they have either the same unique ID or the same name, that is, formally: $(a^{type'} = a^{type})$ AND $((a^{ID'} = a^{ID})$ OR $(a^{name'} = a^{name}))$. The necessary interaction match condition for an LEE interaction $e'(v'_s, v'_t, \mathbf{a}^{e'})$ to match an interaction $e(v_s, v_t, \mathbf{a}^e)$ in the baseline model is that elements v'_s and v'_t satisfy the necessary condition to match elements v_s and v_t , respectively, and that the interaction signs are same $(a^{sign'} = a^{sign})$.

If two corresponding attributes in the model and in the LEE are different (e.g., $a^{location'} \neq a^{location}$), we refer to this as attribute mismatch. We note here that, in the case when the attribute value in the LEE interaction or in the baseline model interaction is “empty”, this will not be considered as an attribute mismatch. Additionally, the LEE evidence attribute, $a^{evidence'}$, is not compared with the model, as it does not include information about the interaction itself, but instead it contains interaction metadata.

3.1.2 Model corroboration requirements

Definition 16. To classify an LEE as a **corroboration**, VIOLIN first determines whether there exists an interaction $e(v_s, v_t, \mathbf{a}^e)$ in the baseline model to satisfy the necessary interaction match condition for the LEE interaction $e'(v'_s, v'_t, \mathbf{a}^{e'})$.

Since this condition is not sufficient for the LEE to corroborate an existing model interaction, the comparison of other corresponding attributes between the LEE interaction and the baseline model is required. These comparisons and the classification decisions for all classification categories are outlined Section 3.3. We define a *strong corroboration* when, besides the necessary interaction match condition, it also holds that the $a^{location'}$ and $a^{locationID'}$ parameters of both v'_s and v'_t match the corresponding parameters in v_s and v_t , respectively, and that $a^{connectiontype'} = a^{connectiontype}$. We show an example of this classification in Figure 3-2. In this figure, we show how example evidence text from an LEE would be represented alongside the model network in each classification case. Additionally, if a user adds more attributes, the list of attributes that are required to match can be extended.

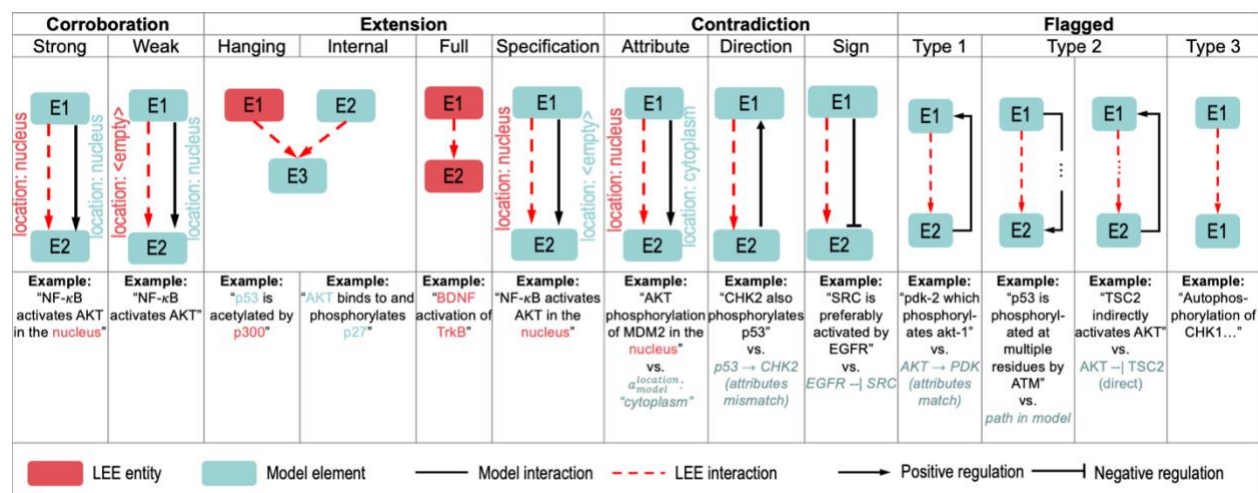


Figure 3-2. Examples of VIOLIN classifications and subclassifications of LEEs: strong corroboration, weak corroboration, hanging, internal, and full extensions, specification, contradictions, and flagged interactions. This figure allows easier visualization the formal definitions. Model elements are represented by blue nodes, and model edges by black arrows. LEE entities, edges, and attributes are represented in red.

We also define several types of *weak corroborations* as follows. When there is a baseline model interaction such that the LEE satisfies necessary conditions to match this interaction, and there is no mismatch among the remaining attributes between the LEE and the model interaction, with at least one of the remaining attributes in the LEE being “empty”, we define this as a *weak corroboration type 1*. Next, a *weak corroboration type 2* is an indirect LEE that satisfies necessary conditions to match with a direct baseline model interaction ($a^{connectiontype'} = \text{“indirect”}$, $a^{connectiontype} = \text{“direct”}$), and there is no mismatch among the remaining attributes (“empty” attributes are allowed). Finally, a *weak corroboration type 3* is an indirect LEE with nodes v'_s and v'_t that satisfy necessary conditions to match two nodes in the baseline model while the LEE does not match any baseline model interaction, and instead it satisfies necessary conditions to match a path in the baseline model ($v'_s = v_{sp}$, $v'_t = v_{tp}$, $a^{sign'} = a^{signp}$). Weak corroborations of type 2 and type 3 are called “weak” due to the assumption by the machine readers that $a^{connectiontype'} = \text{“indirect”}$ by default, unless very specific criteria is met to indicate $a^{connectiontype'} = \text{“direct”}$, meaning LEEs with $a^{connectiontype'} = \text{“indirect”}$ may not capture a direct relationship if it is not explicitly stated in the text, or the text may convey a more broad understanding of a relationship between two elements than is found in the model. Therefore, in these two subcategories, although the LEE agrees with model, the model presents additional details about the relationship between two elements (Figure 3-2).

3.1.3 Model contradiction requirements

Definition 17. An LEE is classified as a **contradiction** if it contains information that disputes the model.

Figure 3-2 outlines the attribute comparison details that determine whether an LEE is a contradiction. Additionally, VIOLIN recognizes several different types of contradictions. For a given LEE interaction $e'(v'_s, v'_t, \mathbf{a}^{e'})$, if there exists a baseline model interaction $e(v_s, v_t, \mathbf{a}^e)$ such that v_s satisfies the necessary condition to match v'_t , v_t satisfies the necessary condition to match v'_s , and either the model interaction is indirect ($a^{connectiontype} = \text{"indirect"}$), or the model interaction is direct ($a^{connectiontype} = \text{"direct"}$) but there is a mismatch between the LEE interaction and the model interaction in at least one attribute other than the sign attribute, then we classify this LEE as a *direction contradiction* (Figure 3-2). The reasoning behind this definition is that an indirect model interaction is only a placeholder in the absence of more detailed knowledge, and the new LEE interaction can dispute this previously incomplete knowledge. On the other hand, when the model interaction is direct, and there is a contradiction with LEE in direction and other attributes (e.g., location), this indicates a potential machine reading error or an incorrect model interaction. Other cases of direction mismatch are classified under different category as will be described later in this section. If elements of an LEE satisfy necessary conditions to match elements of a baseline model interaction, but the model and LEE interactions have different signs ($a^{sign} \neq a^{sign'}$), then we classify this LEE as a *sign contradiction* (Figure 3-2). When there exists an interaction in the model that satisfies the necessary conditions to match the LEE interaction, and there is a mismatch between the model interaction and the LEE interaction in at least one attribute other than $a^{connectiontype}$, then we call this *attribute contradiction* (Figure 3-2).

3.1.4 Model extension requirements

Definition 18. An LEE is classified as an **extension** when it introduces a new interaction to the baseline model that does not contradict it.

VIOLIN recognizes several different cases when this occurs. When there is no baseline model element that matches either of the LEE elements, then the LEE interaction cannot have a corresponding interaction in the model, and we call such LEEs *full extensions* (Figure 3-2). If only one of the LEE elements matches a model element, the LEE interaction still cannot have a corresponding interaction in the model, and we call such LEEs *hanging extensions* (Figure 3-2). If there exist two different model elements v_i and v_j such that the LEE elements v'_s and v'_t satisfy at least the necessary condition to match v_i and v_j , respectively, while there is no interaction $e(v_i, v_j)$ in the model, and either $a^{connectiontype'} = \text{"direct"}$, or $a^{connectiontype'} = \text{"indirect"}$ and there is also no path $p(v_i, v_j)$ in the model, then we call such LEEs *internal extensions* (Figure 3-2). When an LEE satisfies the necessary condition to match a baseline model interaction and, either all attributes match and there is at least one "empty" model interaction attribute and a non-"empty" corresponding LEE attribute, or $a^{connectiontype'} = \text{"direct"}$ and $a^{connectiontype} = \text{"indirect"}$ and all the remaining attributes match, we call such LEEs *specifications* (Figure 3-2). They do not replace existing model interactions but instead add more specific information to them.

3.1.5 Flagged interactions

Definition 19. There are some cases where a combination of matched, mismatched, and “empty” attributes requires further manual inspection before it can be classified, and therefore, these LEEs are *flagged* by VIOLIN.

Within the flagged category, there are three subcategories. For a given LEE interaction $e'(v'_s, v'_t, \mathbf{a}^{e'})$, if there exists a direct ($\alpha^{connectiontype} = \text{“direct”}$) model interaction $e(v_s, v_t, \mathbf{a}^e)$ such that v'_s and v'_t satisfy the necessary condition to match v_t and v_s , respectively, and there is no mismatch among node attributes (“empty” attributes are allowed), then we classify this LEE as *flagged type 1* (Figure 3-2). A *flagged type 2* LEE is an indirect LEE ($\alpha^{connectiontype'} = \text{“indirect”}$) with nodes v'_s and v'_t that satisfy the necessary condition to match two nodes in the baseline model, v_i and v_j , respectively, while the LEE does not match any baseline model interaction, and instead there is either a path $p(v_i, v_j)$ in the model with at least one mismatched attribute, or a path $p(v_j, v_i)$ (Figure 3-2). When both LEE elements satisfy the necessary condition to match the same model element, such an LEE appears to be a self-regulation, and is classified as *flagged type 3*. (Figure 3-2). We flag these LEEs as in some cases they may indeed be self-regulation interactions, but more often the self-regulation is a result of grounding and the level of abstraction in the baseline model. Grounding is an extraction step that uses public databases [5, 7] to pair a standard identified with the common name used in the literature text, though it can occasionally result in error. For example, in the literature, Caspase-3 and Caspase-8 interact with each other as separate entities, while in the melanoma model that we use as one of our case studies, all the caspases are grouped in a single variable. Thus, an LEE representing the interaction between Caspase-3 and

Caspase-8 is classified as flagged type 3 (self-regulation), and the user can then decide how to use the LEE. Self-regulation LEEs sometimes also result from machine reading errors, if a qualifier such as “mutated” or “phosphorylated” is ignored by the reader. For example, the statement “R2834H DP decreases phosphorylation of DP compared with WT” describes how a mutation of desmoplakin protein reduces the phosphorylation of the unmutated protein, the extracted event is “DP negatively regulates DP,” ignoring the important distinction of the mutated form.

3.1.6 Path and loop finding

In the process of classifying an LEE, when it finds two nodes in the baseline model graph that match v'_s and v'_t elements, VIOLIN often searches for paths between the two model nodes. In turn, such a search can provide additional information about the relationship between the baseline model and the LEE. Given the path definition in Section 2.1.2, if VIOLIN finds a path $p(v_{s_p}, v_{t_p}, \mathbf{a}^{sign_p})$ in the baseline model, such that v_{s_p} matches v'_s and v_{t_p} matches v'_t , then this LEE could potentially form a feed-forward loop if added to the model. If the path and the LEE interaction signs are the same ($a^{sign_p} = a^{sign'}$), the feed-forward loop would be positive, and if the signs are different, the loop would be negative (Figure 3-2). If v_{s_p} matches v'_t , and v_{t_p} matches v'_s , in such cases, the LEE could form a feedback loop when added to the model; depending on whether the path and the LEE interaction signs are the same or different, the feedback loop would be positive or negative, respectively.

3.2 Scoring

Within a given LEE set, VIOLIN evaluates each LEE using several different scores: Evidence score, Match score, Kind score, Epistemic score, and Total score. The Evidence and Epistemic scores depend on the literature that was selected and read by machine readers, and they are independent of a baseline model, while the Match and Kind scores result from the comparisons between the LEEs and the baseline model. The Total score is a combination of the other four scores. In the following, we provide a detailed description of the purpose of each score type and the method for computing it.

3.2.1 Evidence Score

It is usually assumed that if there are more mentions of an event in the literature, it is more likely that the event is both correct and useful to the model. Tools such as STRING [55] and INDRA [2] base their judgement of interactions on how frequently the interaction is either found in online databases or extracted with machine reading engines, respectively. The **Evidence score** (S_E) counts all the mentions of each distinct interaction within an LEE set. Similar to the necessary conditions for a match between LEE and model interactions, described in Section 3.1.1 we consider two LEE interactions $e'_{j1} = e'(v'_{sj1}, v'_{tj1}, \mathbf{a}_{j1}^{e'})$ and $e'_{j2} = e'(v'_{sj2}, v'_{tj2}, \mathbf{a}_{j2}^{e'})$ to be same if they satisfy the following condition: $(a_{sj1}^{name'} = a_{sj2}^{name'} \text{ OR } a_{sj1}^{ID'} = a_{sj2}^{ID'}) \text{ AND } a_{sj1}^{type'} = a_{sj2}^{type'} \text{ AND } (a_{tj1}^{name'} = a_{tj2}^{name'} \text{ OR } a_{tj1}^{ID'} = a_{tj2}^{ID'}) \text{ AND } a_{tj1}^{type'} = a_{tj2}^{type'} \text{ AND } a_{j1}^{sign'} = a_{j2}^{sign'} \text{ AND } a_{j1}^{connectiontype'} = a_{j2}^{connectiontype'}$. If two interactions do not satisfy this necessary condition, they are called distinct.

VIOLIN also allows users to add other attributes to this condition, if necessary. As can be

concluded from the definition of S_E , it is computed with respect to the overall LEE set, and therefore, it is also dependent on the user question and the selected literature. While computing S_E for each distinct LEE, VIOLIN collects the PMCID of the papers from which these LEEs have been extracted and adds them to the $a^{evidence'}$ attribute.

3.2.2 Match Score

The **Match score** (S_M) is used to quantify the match between elements of an LEE (v'_s and v'_t) and elements in the model. Following the definition of a match between an LEE element and a model element from Section 3.1.1, VIOLIN recognizes four different cases of matching the two LEE elements with the baseline model, and it assigns a value to each case. In other words, we define a set of values $\{SM_1, SM_2, SM_3, SM_4\}$, and assign one of these values to the S_M score of a given LEE, as follows: SM_1 , when both v'_s and v'_t match elements in the baseline model; SM_2 , when v'_s matches an element in the model, and v'_t does not match any model elements; SM_3 , when v'_t matches an element in the model, and v'_s does not; SM_4 , when neither v'_s nor v'_t match any elements in the model. The four S_M values are not fixed, VIOLIN allows their change, such that they can fit different use cases. For example, if we are interested in adding only new edges to the baseline model graph, we can increase the SM_1 value, thus increasing the score for those LEEs that are classified as internal extensions. On the other hand, if we are interested in expanding the baseline model network, we can increase the other three scores (SM_2 , SM_3 , and SM_4). Obviously, adding to the model those LEE interactions that are classified as full extensions (assigned score SM_4) will lead to a disconnected graph, unless there are interactions in the LEE set that are classified as hanging extensions (assigned either score SM_2 or SM_3) that can connect full extensions with the model.

The default values used to represent the Match classifications are defined by the set $\{s_{M1} = 10, s_{M2} = 1, s_{M3} = 100, s_{M4} = 0.1\}$, representing a scoring scheme for the case of general model extension, where a user seeks to expand both the breadth and depth of the baseline model network without creating excessive output nodes. We will explore the influence of the selection of s_{M1} , s_{M2} , s_{M3} , s_{M4} values on the classification, and consequently, on modeling, in Section 5.1.

3.2.3 Kind Score

The *Kind score* (S_K) is used to quantify the relationship of an LEE interaction to the baseline model. We define a set of values $\{s_{K1.1}, s_{K1.2}, s_{K2.1}, s_{K2.2}, s_{K3}, s_{K4}\}$ for the classification categories described in Section 3.1. Since the difference between full, hanging and internal extensions is already captured with the Match score, we use the same value, $s_{K1.1}$, for these three extension types, and another value, $s_{K1.2}$, for the specification extensions. Strong corroborations are assigned value $s_{K2.1}$ and weak corroboration are assigned value $s_{K2.2}$. All three contradiction sub-categories, the direction, sign, and attribute contradictions, are assigned value s_{K3} . Any remaining, flagged LEE is assigned value s_{K4} . Similar to S_M , the six S_K values may be selected by the user, and VIOLIN allows their change, in order to fit different use cases. The default values used to represent the Kind classifications are defined by the set $\{s_{K1.1} = 2, s_{K1.2} = 1, s_{K2.1} = 40, s_{K2.2} = 30, s_{K3} = 10, s_{K4} = 20\}$, again for the general model extension case. Here, we assign highest importance to extensions, and then to flagged LEEs which may identify feed-forward or feedback loops, and then contradictions which may represent corrections to outdated information in the baseline model. We will discuss the influence of the Kind score value on modeling in Section 5.2.

3.2.4 Epistemic Value and Total Score

Some machine reading engines and information assembly tools output a quantified measure of the believability in the extracted interaction, such as INDRA’s belief score, which is based on information directly from the machine readers. If available, we will use this value as an *Epistemic score* (S_B). Another such value is RUBICON’s Epistemic Value [98], where Zero, Low, Moderate, and High believability correspond to numerical scores of 0.0, 0.33, 0.67, and 1.0, respectively. When no value for the Epistemic score is available, the default value is accepted as 1.0, so that all LEEs are assumed to have the same believability.

Finally, for each LEE in a given LEE set, we combine the previous four scores into the *Total score* (S_T) using the following equation:

$$S_T = (S_K + (S_E \times S_M)) \times S_B \quad 3-1$$

In other words, S_T represents the quantity, quality, and believability of the LEE. The Match score is the most influential factor in the Total score, combined with the Evidence score because the addition of elements is generally the driving force behind model extension. In other words, multiplying the Match score by the Evidence score increases the Total score for the interactions found multiple times in the literature. We designed the scoring method this way so that an LEE with a high Evidence score may overcome a low Match score. For example, an LEE with $S_E = 12$ can move an LEE to its next highest Match ranking, assuming that the user has compiled their LEE set from literature relevant to the baseline model system, and so a well-documented LEE of a less desirable Match classification is of higher value than a poorly documented LEE of a more desirable Match classification.

However, we want to allow for the possibility that novel or unique interactions which have not necessarily been well studied would be good candidates for extension, and we capture this possibility in the Kind score. The Kind score can be used as a filtering method to select for specific classifications, but it remains as an addition factor so that while it can increase the total score, it cannot *overcome* the influence of the Match and Evidence scores. When available, the Epistemic score may also significantly influence the Total score. If we do not have high enough confidence in an LEE, the Epistemic score smaller than 1 decreases the Total score, even when the LEE has a high evidence count.

3.3 Algorithm

VIOLIN makes decisions on the classification and scoring of an LEE based on the attributes and nodes as they are defined in Section 2.5.2.2. Figure 3-3 shows the step-wise comparisons for a given LEE, and how these comparisons lead to the final classification outcomes. First VIOLIN determines and compares the connection types of the LEE and Model interaction. This can be explicitly defined in the input files, or VIOLIN can assign a default. For LEEs, the default connection type is $a^{connectiontype} = \text{“indirect,”}$ following the REACH method for identifying connection type [121]. For model interactions, the VIOLIN-defined default is $a^{connectiontype} = \text{“direct,”}$ assuming that most models are assembled of known interactions, however, this default can be reassigned to “indirect” by the user, if the model is known to present more general interactions.

Next VIOLIN makes use of the implicit attributes; the $a_e^{direction}$ of the LEEs and model interactions coming from the definition of the source and target nodes, and then a_e^{sign} , which may

come from identified columns in the input files, or from a dedicated attribute (depending on source of the LEE set and baseline model). Finally, VIOLIN compares any additional attributes available in the LEEs and baseline model. The necessary attributes are either found to be a match or mismatch, denoted by “M” or “X” in Figure 3-3, respectively, and based on the definition of a match as given in Section 3.1.1. However, the “Other attributes” as noted in the figure (e.g. $\alpha^{location}$) can have a more complex comparison. These attributes may be missing from either the LEE or model interaction, denoted as “– +” if present only in the model, “+ –” if present only in the LEE. The attributes may also be absent in both LEE and model interaction or present and matching, this case which is represented by “+ / – M” in Figure 3-3. Finally, the attribute may be present in both the LEE and model interaction, but mismatched, denoted by “+ + X.” These more specified cases allow for the weak corroboration and specification classifications detailed in Section 3.1.

Step 6 of Figure 3-3 shows the final classification outcome defined by the combination of outcomes from the preceding steps. The full detail of the steps of the VIOLIN algorithm can be found in the Supplement. The classification decisions are based on biological principles – how

biological entities are known to interact. In Chapter 5.0 we further investigate the robustness of this classification scheme.

STEP 1	STEP 2	STEP 3	STEP 4	STEP 5	STEP 6
LEE	MI	Direction	Sign	Other attributes	Classification
Direct	Indirect	M	M	+/-M	
		M	M	-+	
		M	M	+-	
		M	M	++X	
		M	X	-----	
		X	-----	-----	
		M	M	+/-M	
		M	M	-+	
		M	M	+-	
		M	M	++X	
		M	X	-----	
		X	M	+/-M	
	Direct	X	M	-+	
		X	M	+-	
		X	M	++X	
		X	X	+/-M	
		X	X	-+	
		X	X	+-	
		X	X	++X	
		-----	-----	-----	
		-----	-----	-----	
		-----	-----	-----	
		-----	-----	-----	
		-----	-----	-----	
	Path	-----	-----	-----	
	NO	-----	-----	-----	
Indirect	Indirect	M	M	+/-M	
		M	M	-+	
		M	M	+-	
		M	M	++X	
		M	X	-----	
		X	-----	-----	
		M	M	+/-M	
		M	M	-+	
		M	M	+-	
		M	M	++X	
		M	X	-----	
		X	M	+/-M	
	Direct	X	M	-+	
		X	M	+-	
		X	M	++X	
		X	M	+/-M	
		X	X	-+	
		X	X	+-	
		X	X	++X	
		-----	-----	-----	
		-----	-----	-----	
		-----	-----	-----	
		-----	-----	-----	
		-----	-----	-----	
	Path	-----	-----	-----	
	NO	-----	-----	-----	

MATCH COMPARISON	
present/absent in both, match	+/-M
absent in LEE, present in MI	-+
present in LEE, absent in MI	+-
present in both, mismatch	++X
NA	-----
OUTCOME	
Contradiction	
Extension/Specification	
Strong/Weak Corroboration	
Flagged	

Figure 3-3. Interaction chart showing the outcome of all possible comparisons of a given LEE to given baseline model interactions (MI).

3.4 Selecting VIOLIN input

When selecting VIOLIN input, or seeking to add a new benchmark, there are several things to consider. First, there is a minimum information requirement for both the LEE set and the baseline model. For a given LEE set, each LEE must identify the $a^{name'}$, $a^{type'}$, and $a^{ID'}$ for both v_s' and v_t' , as well as $a_e^{direction'}$ and $a_e^{sign'}$. In cases where $a_e^{connectiontype'}$ is not available, it is assumed to be “indirect” following the REACH machine reading methods [76]. Similarly, for a baseline model, the model must contain a^{name} , a^{type} , and a^{ID} for v_s and v_t , as well as $a_e^{direction}$

and a_e^{sign} . When $a_e^{connectiontype}$ is not available, the user can input what VIOLIN should assume as the default.

In addition to the minimum information requirements, it's important to consider the types of interactions contained in the LEE set and the baseline model. Machine reading engines can generally identify PPIs, PGIs, PCIs, and PBIs from the literature, as defined in Section 2.3.1. When selecting a baseline model, the user has to decide if they want to only consider those types of LEEs which match the types of interactions found in the model, or if they want to extend the types of interactions. Many baseline models on databases such as NDEx [69] or Wikipathways [70] are strictly PPIs, an LEE set which contains mostly PGIs, PCIs, and PBIs, may not produce VIOLIN output with high relevance or usefulness, unless the user's goal is specifically to introduce genes, chemicals, and/or biological processes to the model.

The user's choices can be determined by their modeling use case. A breadth-first extension would increase the network size of the model, and this extension could be further specified to show preference for finding new regulators or input nodes (looking for the drivers of a system) or preference for finding new regulated or output nodes (looking for new outcomes from the network catalysts). A depth-first extension would increase the edges in a baseline model, thus increasing the knowledge about the interactions between nodes. And then a validation extension would confirm that the interactions represented in the baseline model are backed up by the current literature.

3.5 Compatibility with Other Tools

Currently, VIOLIN is compatible with many of the modeling tools and databases already available. The default LEE input expected is based on the REACH machine reading tool [121], which the output from the INDRA tool [2] can easily be transformed into. Some machine readers and tools do not distinguish between positive and negative regulator nodes [75], instead representing the regulation sign as a distinct attribute a^{sign} . VIOLIN contains a built-in function to handle this case without a loss of information. VIOLIN also does not specify the nomenclature used to refer to interaction elements. While we define the interaction elements as source and target nodes (Section 2.1.2), other tools use the terms regulator/regulated or element. VIOLIN has the capability to accept them all and recognize the appropriate function of the element.

The default model input is expected as the BioRECIPES format [13], meant for dynamic model simulation and also for easy user understanding. However, a common representation format of models across databases is as a node-edge list [69, 70], much like the machine reading output. VIOLIN was built with the functionality to transform multiple model formats. Specifically, between the BioRECIPES format and the node-edge format found on the NDEx [69], WikiPathways [70], and Reactome [50] databases. VIOLIN is also compatible with the INDRA output, meaning that VIOLIN can access machine reading output from REACH [76], TRIPS [75], and the BEL large corpus [97]; that is, it can transform any set of statements from INDRA into the input LEE set. While INDRA uses the JSON representation format for its output statements [102], which is suitable for machines but not practical for human use, VIOLIN's transformations functions help bridge the gap between machine and human use.

3.6 VIOLIN UI & Visualization

VIOLIN was written as a Python package, with pandas and NumPy dependencies, and can be found at <https://bitbucket.org/biodesignlab/violin/src/master/>. The associated documentation can be found at <https://violin-tool.readthedocs.io/>. While VIOLIN does not link to machine readers or information assembly tools directly, it is compatible with many forms of input. The current user interface is script-based; the user assembles either python script or Jupyter notebook calling the desired VIOLIN functions. A tutorial ipy notebook showing use of VIOLIN with default settings can be found at https://bitbucket.org/biodesignlab/violin/src/master/violin_tutorial/.

Currently, the default input format for VIOLIN is a spreadsheet from REACH or INDRA machine reading output and the BioRECIPES model format. For input outside of these formats, VIOLIN has functions to transform between representation formats, allowing for the compatibility with tools such as FLUTE [106] and CLARINET [111] as described in Chapter 6.0, along with databases and tools such as NDEx [69] and INDRA [2].

VIOLIN output is organized into five spreadsheets: one total output, with all the LEEs listed by their Total Score, and one each for the corroborations, extensions, contradictions, and flagged LEEs. This output can be used as input for a model extension tools such as ACCORDION or CLARINET, or can be visualized using the VIOLIN [visualization function](#), which provides a visual summary of any of the output files. Figure 3-4 shows an example of this visualization for the Total output, summarizing the scores and the classification break down. Part of this function is a filtering value, which allows the user to visualize VIOLIN output based on specific criteria. The user can either use a thresholding value for the Total score or Evidence score, summarizing the output only of LEEs with such a score above a specific value, or the user can threshold a certain

percentage, summarizing the output of the top X% of LEEs, based on the Total score. This visualization allows the user to make decisions on how they wish to use the VIOLIN output.

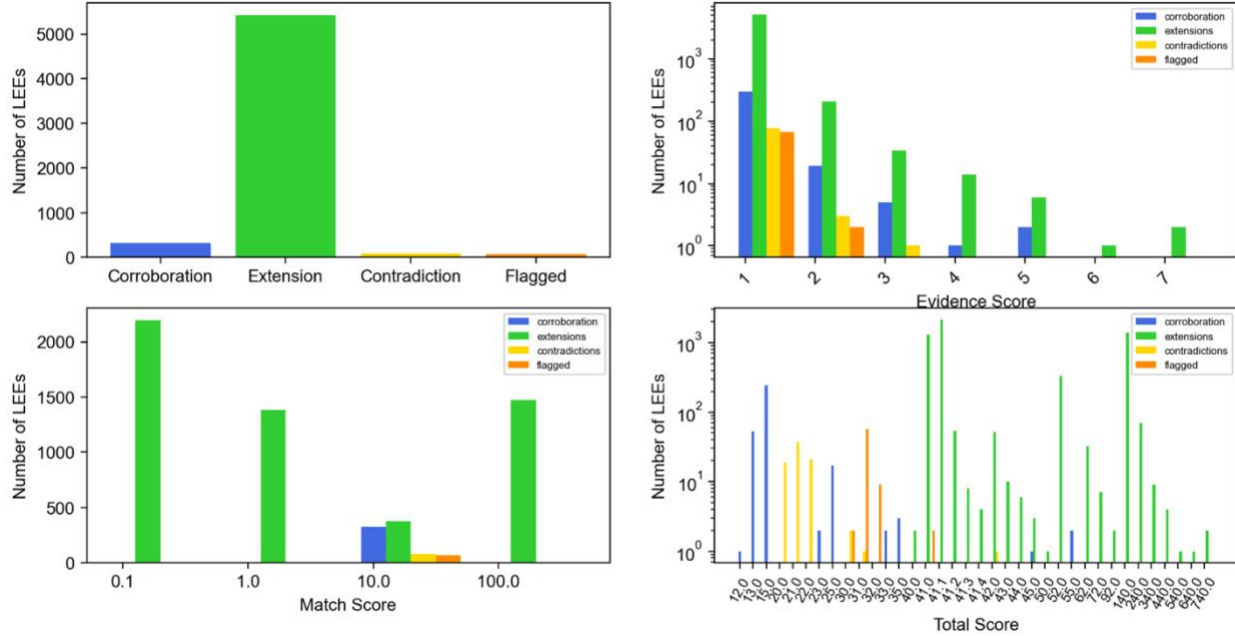


Figure 3-4. Example output from the VIOLIN visualization function, showing the classification distribution, and the distributions of the Evidence, Match, and Total scores.

From this visualization, the user can surmise that the majority of the LEEs are extensions, from the top left graph. From the bottom left graph, they can discover that about half of the LEEs from the set are connected to the baseline model system, knowing that the Match Scores represent the categorizations defined in Section 3.2.2. Approximately 2,300 LEEs are completely unconnected while approximately 3,000 LEEs are connected to the baseline model system by either the source node ($S_M = 1$), the target node ($S_M = 100$), or by both nodes ($S_M = 1$). This distribution could be caused by either a small baseline model network, or by LEEs assembled from papers/databases which were not chosen with close enough relevance to the model.

The top right graph of Figure 3-4 shows the distribution of Evidence scores across the LEE set. The highest Evidence score being 7, the user could conclude that the LEEs set was varied, and LEEs did not often repeat. The Total score distribution, shown in the lower right graph of Figure 3-4, helps illustrate why we would not want the Kind score to be the leading factor in the Total score calculation. While there is a pretty clear divide between the extensions and the other three categories, the corroborations, contradictions, and flagged are more mixed. If the Kind score were the leading factor in the Total score, there would be no mixing, there would only be discrete sections of each category. This would mean the user would have to choose from each category which LEE to use. By having the Match and Evidence scores as the leading factors of the total score, there is a more continuous distribution, which allows for automatic LEE thresholding based on the Total score. The Evidence score and Total score distributions suggest that the contradictions and flagged interactions are not particularly useful to this user's needs, possibly due to machine reading errors. We investigate the prevalence of machine reading errors further in Section 4.6.

4.0 VIOLIN Outcomes

In this chapter, we show the outcomes of the VIOLIN tool on initial benchmark input. We first describe the Models and LEE sets curated for initial use of VIOLIN, as well as for investigations carried out in Chapters 5.0 and 6.0 (Section 4.1). We then show the initial results of VIOLIN’s judgement of LEE sets against two different baseline models (Section 4.2). We then illustrate how modeling goals guide classification strategies (Section 4.4), and how the path finding function leads to the discovery of loops (Section 4.5). We end the chapter with a deeper investigation into contradictions and flagged LEEs, and their potential for identifying machine reading errors (Section 4.6).

To demonstrate VIOLIN’s capabilities in this chapter, we conducted several experiments, using two baseline models, and seven LEE sets. The main differences between the models are size and complexity, and the difference between reading sets is size and the method used to create them, with two sets created as “negative sets” to be deliberately irrelevant to the model.

4.1 Models and LEE Construction

In total, we curated five models of various systems and sources. The first model we used is a discrete model of biological pathways in the melanoma SkMel-133 cell line. This model (we will refer to it as model A) was created using the model in [122] by removing the immune cells and incorporating mutations of the melanoma cell line used in Korkut et al. [123], in which the authors performed DNA, RNA, and protein analysis on primary and metastatic melanomas from

331 patients. This model was in the BioRECIPES representation format [13], and included the connection types for all model interactions. The second model we used is a model of the circuitry that controls the differentiation of naïve T cells into regulatory (Treg) and helper (Th) cells, described in [124]. This model (we will refer to it as model B) was manually created to investigate the early steps of T cell activation, and the suppression or generation of regulatory T cells (Treg). This model was also already in the BioRECIPES format but did not include the connection type attribute. In this case, we set at input that the model connection types should be assumed to be “direct,” under the reasoning that baseline model interactions will generally be validated and well understood.

Our third baseline model (model C) was found on NDEx [69], from the publication of Sandhya et al. [125]. This model is a network of the BDNF pathway as it may relate to the regulatory processes involved in major depression disorder. This model was represented on the NDEX database as a node-edge list, which was made compatible with VIOLIN through its internal transformation functions. This model did not include the connection type attribute – it was added manually using the evidence text. Our fourth baseline model (model D) is of the pancreatic cancer cell model presented in [122]. It already existed in the BioRECIPES format, so it did not need transformation. Though it did not include the connection type attribute, and in this case the default connection type was again assumed to be “direct.” Our last model (model G) is of the Glioblastoma Multiforme system, as presented in [33]. This was a model automatically assembled and verified from the relevant literature and represented in the BioRECIPES format. Again, the connection types were not available in the model, and so were assumed to all be “direct.” Full descriptions of all models used in this work can be found in Appendix A.

For these models, we assembled a number of LEE sets. Three of our LEE sets (R_{B0} , R_{C0} , R_{D0}) were assembled by using the REACH machine reader to extract interactions from the reference papers for the model publications, and we denote this method with the subscript 0. We then have LEE sets where were created with the REACH tool, indicated with single number subscripts, R_{A1} , R_{B2} , R_{A3} , etc. R_{A1} was created using a general search query in REACH. R_{A2} was created using the more specific REACH Explorer tool [121], where papers were selected using 13 protein queries. For each protein, we first selected top 10 papers that were returned by the REACH Explorer. In the case where 10 papers were not found, we obtained additional papers using the Fetch tool [121]. This ultimately led to 125 papers, from which REACH extracted 6305 interactions. The remaining such denoted LEE sets were assembled by querying terms related to the associated baseline model using the REACH Explorer tool [121], and then using INDRA [2] to access the REACH machine reading engine [76]. A Full list of queries can be found in Appendix B.

In Chapter 6.0, we investigate the use of the FLUTE tool [106] in conjunction with VIOLIN, and those LEE sets which are filtered by FLUTE are denoted by the #.1 subscript (e.g. $R_{A2.1}$). For LEE sets $R_{A2.0.1}$ and $R_{B0.0.1}$, we manually removed all LEEs involving chemicals or biological processes. We call this manual filtering (1), and label it with #.0.1 subscript. For LEE R_{A2} , we also took the resultant $R_{A2.0.1}$ LEE set and had an expert manually judge the LEEs for redundancy to the model network and for obvious errors. We call this manual filtering (2) and label it with a #.0.1.1 subscript. We finally created two LEE sets, R_{B*1} and R_{B*2} and used them as “negative” sets. They were created by the same method of using REACH explorer, INDRA, and REACH, but their query terms included elements and pathways specifically not relevant for the T cell model. This was an important test of the accuracy of VIOLIN, as we would expect that most

if not all LEEs from these sets should be judged as extensions, with any exceptions being either machine reading errors or ubiquitous biological interactions.

We show the metrics of our LEE sets and models in Figure 4-1. Figure 4-1A shows the model metrics and the LEE set notation scheme for VIOLIN inputs used in Chapters 4.0, 5.0, and 6.0. Figure 4-1B shows a sample of query terms for a selection of LEE sets. The full list of query terms and curation methods for all LEE sets can be found in Appendix B. Figure 4-1C shows the number of papers used to assemble each LEE set, and the resultant number of LEEs. Across the LEE sets without additional filtering, we found there to be an average of 26 LEEs per paper. In the following chapter, results will be shown from VIOLIN output judging LEE sets against models A and B.

Our models A and B, along with the associated inputs, were used for the initial investigation of VIOLIN, testing how well-assembled models and LEE sets are judged in VIOLIN. We expect these inputs to produce a good distribution of classifications, allowing us to investigate the outcomes of corroboration, extension, contradiction, and flagged judgements (Sections 4.2-4.5). We also use two specific LEE inputs, R_{A1} and $R_{A2.0.1.1}$, to further investigate how VIOLIN's judgement compares to an expert identification of erroneous or redundant LEEs (Section 4.6). Here we expect the contradiction and flagged classification to help identify machine reading errors.

With our Models C and D, we use the reference input (R_{C0} and R_{D0}) to test the efficacy of VIOLIN's corroboration and extension judgement (Sections 5.4 and 5.5). Here we would expect the majority of model interactions to be corroborated by LEEs sourced from the model references, and we also expect to be able to recover removed interactions as extensions.

Models A, B, and C are also used to investigate the outcomes of using VIOLIN with the FLUTE tool, testing how both judgement and filtering can reduce the number of LEEs used to

extend a model.. Finally, Model G and its associated output are used to test how VIOLIN can be used to affect input for model extension tools such as CLARINET (Section 6.2.2). Here, we expect that VIOLIN will alter the size and entities of the clusters, as judgement is used to focus the scope of the desired extensions (in this case, extensions that are already connected to the model network).

(A)

Baseline model	Model System	# Nodes	# Edges	LEE set				
				Machine reading from references	Machine reading from queries	+ Manual filtering (1)	+ Manual filtering (2)	+ FLUTE filtering
A	Skel-133 melanoma	255	325	none	R _{A1}	none	none	none
				none	R _{A2}	R _{A2.0.1}	R _{A2.0.1.1}	R _{A2.1} , R _{A2.1.1} , R _{A2.1.1.1}
				none	R _{A3}	none	none	R _{A3.1}
				none	R _{A4}	none	none	none
B	T-cell	61	89	R _{B0}	none	R _{B0.0.1}	none	R _{B0.1} , R _{B0.1.1}
				none	R _{B2}	none	none	R _{B1.1}
				none	R _{B3}	none	none	R _{B3.1}
B*				none	R _{B*1}	none	none	none
				none	R _{B*2}	none	none	none
C	BDNF pathway	72	82	R _{C0}	none	none	none	R _{C0.1}
				none	R _{C1}	none	none	R _{C1.1}
				none	R _{C2}	none	none	R _{C2.1}
D	Pancreatic Cancer	236	314	R _{D0}	none	none	none	none
G	Glioblastoma multiforme	237	366	none	R _{G1}	none	none	none
				none	R _{G2}	none	none	none

(B)

LEE set	Description
R _{A2}	MEK, ERK, AKT, GSK3, P70RSK, S6, CDK4, 4EBP1, YB1, SRC, CHK2, MTOR, PI3K
R _{A3}	MAPK/ERK pathway
R _{A4}	RPS6K1
R _{B1}	PTEN
R _{B3}	T-cell, PTEN, AKT, FOXO
R _{B*1}	Breast Cancer, DNA repair, Autophagy, Cancer
R _{B*2}	DNA repair, BRCA1, ADAM17, inflammation

(C)

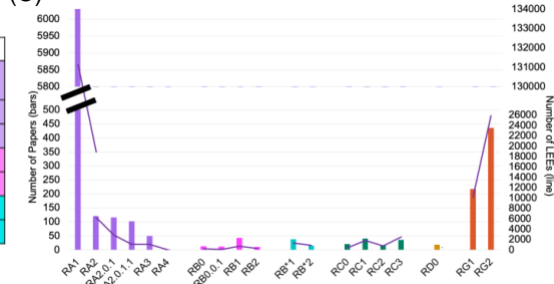


Figure 4-1. Metrics for our tested VIOLIN inputs(A) model metrics and LEE set curation methods, (B) example query terms or methods for each of our LEE sets, and (C) content of each LEE set, showing the number of papers and total number of LEEs per set.

4.2 Classification of Extracted Events in Context of Baseline Model

We show in Figure 4-2 the classification results of the LEE sets introduced in the previous section: (1) four sets classified with respect to the melanoma model (model A), R_{A1} , R_{A2} , R_{A3} , R_{A4} , and (2) three sets classified with respect to the T cell model (model B), R_{B1} , R_{B2} , R_{B3} . While the distribution of LEEs among the four main classification categories defined in Sections 3.1.2-3.1.5 is similar for most LEE sets, with extensions being far more prevalent than other categories, the distribution of classification subcategories varies.

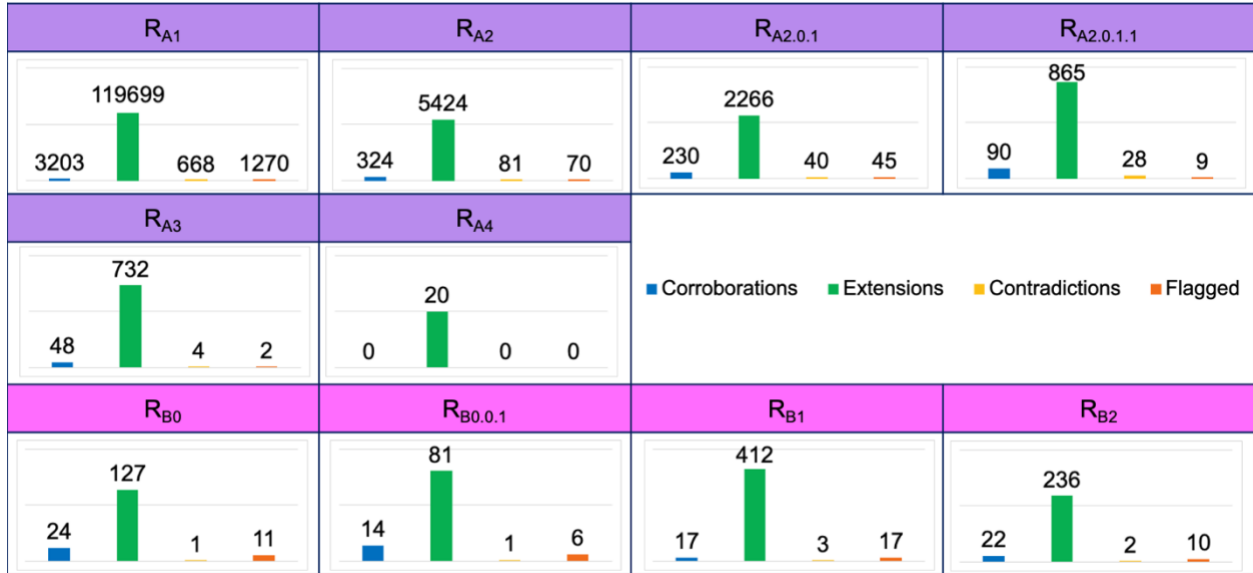


Figure 4-2. Classification distribution of our tested VIOLIN inputs

In the model A case study, the distributions of classification subcategories for the LEE sets R_{A1} and R_{A2} are similar, while the distributions for sets R_{A3} and R_{A4} differ. This is due to both the size and context of the LEE sets. R_{A3} contains 1106 LEEs, while R_{A4} has only 21 LEEs (20 of which are unique). The query for R_{A3} , the MAPK/ERK pathway, is also critical for the modeled system as changes in this pathway are found in many cancers. MAPK and ERK alone are involved

in 15 of the melanoma cell baseline model interactions. In contrast, the query for R_{A4} , RPS6KB1, is a common kinase, but is only involved in two baseline model interactions. There are 10 hanging extensions in VIOLIN output for the R_{A4} set, among 20 extensions, caused by interactions involving entities which are either common in biological processes or interact with RPS6KB1: TERT, ERK, JNK, RAS, E2F, autophagy. On the other hand, all three LEE sets in the model B case study (R_{B1} , R_{B2} , and R_{B3}) have similar distribution of classification subcategories, due to the queries chosen; not only are queries for these LEE sets closely related to the naïve T cell differentiation context, but the queries also overlap with each other, all including T-cell, PTEN, AKT and FOXP3 terms. Sets R_{B2} and R_{B3} even share 18 LEEs from the same paper.

The corroboration subcategory distribution is similar for all LEE sets, except R_{A4} . The subclassification breakdown is shown in Figure 4-3. The most prevalent corroboration subcategory, weak type 3, includes LEEs with nodes that match source and target nodes of a multi-edge path (not a single edge) in the model. The second largest subclassification of corroborations is weak type 2 (when an LEE matches the direction and sign of a baseline model interaction, and does not contradict any other attributes), due to many of the tested LEE sets not having any additional interaction attributes. For example, LEE sets R_{A1} , R_{A2} , and their subsets include only location attribute information, and thus, for these LEE sets VIOLIN compares only the location attribute to the model.

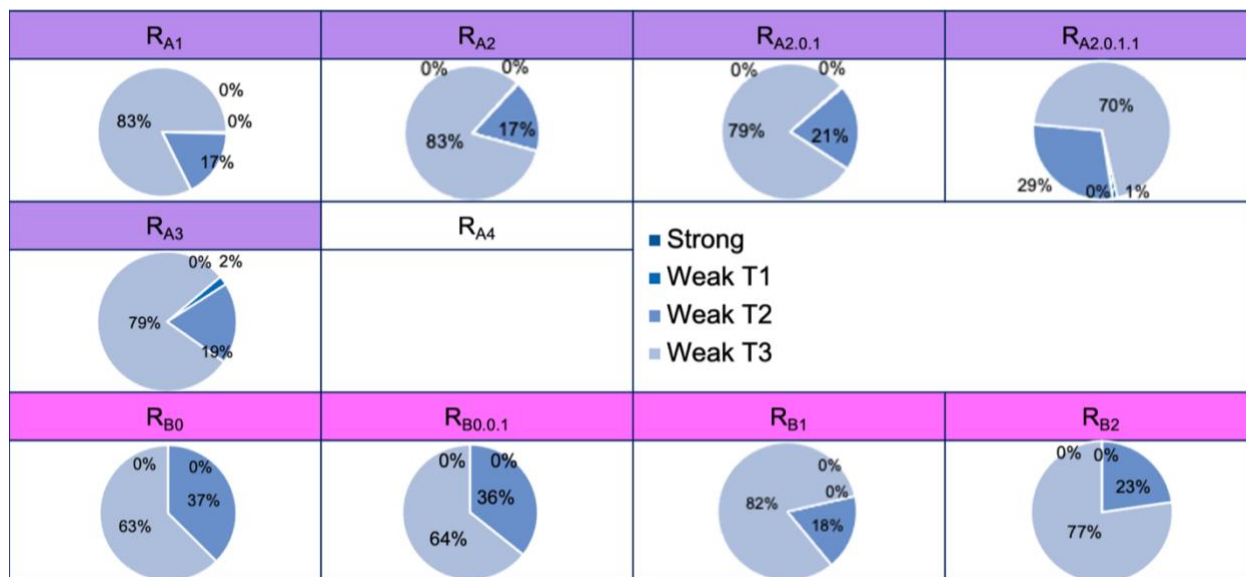


Figure 4-3. The corroboration subclassification judgements of the tested VIOLIN input

As can be seen in Figure 4-4, the extension subcategory judgement distributions, the extension subcategory distribution highlights the prevalence of the hanging subcategory in all LEE sets, confirming that, with the selection of relevant literature, the majority of LEEs are connected to the baseline model. LEE set $R_{A2.0.1.1}$, which was obtained through curation of the original set R_{A2} , has the highest number of internal extensions.



Figure 4-4. The extension subclassification judgements of the VIOLIN input.

The next most common subcategory of extensions in the VIOLIN output are full extensions, also illustrated in Figure 4-4. These results suggest that even when using the machine readers on papers that match the context of the baseline model, not all LEEs will be relevant, or even connected to the baseline model network. This is especially emphasized with 41% of extensions in set R_{A1} being full extensions, and also to an extent with LEEs in set $R_{A2.0.1.1}$ (selected through curation for their relevance to baseline model) where approximately 24% of the extensions do not connect to the original model network directly, though they may connect to an element which does. We also note here that some of the full extensions are result of erroneous grounding of entities by machine readers, which can lead to elements being identified as “new” instead of matching a baseline model element.

Of the contradictions, the predominant subcategory is a sign contradiction, except in the $R_{A2.0.1.1}$ set, which has highest percent of attribute contradictions. Figure 4-5 shows the subclassification breakdowns for individual LEE sets, excepting R_{A4} , as there were no

contradictions identified from this LEE set. As will be discussed later in Section 4.4, for reading sets R_{A1} , R_{A2} , $R_{A2.0.1}$, and $R_{A2.0.1.1}$ only, the location attribute is available. If this attribute were “empty”, or we had chosen not to consider it, all LEEs classified as attribute contradictions in Figure 4-5 would have instead been classified as either weak corroborations or specifications. This illustrates how the user’s classification choices affect the final VIOLIN output (this will be further discussed in Section 5.3). Direction contradictions make up either the smallest number of contradictions, or are not found in the contradictions at all, as shown for our R_B sets, most likely due to the overall very few contradictions in these sets.

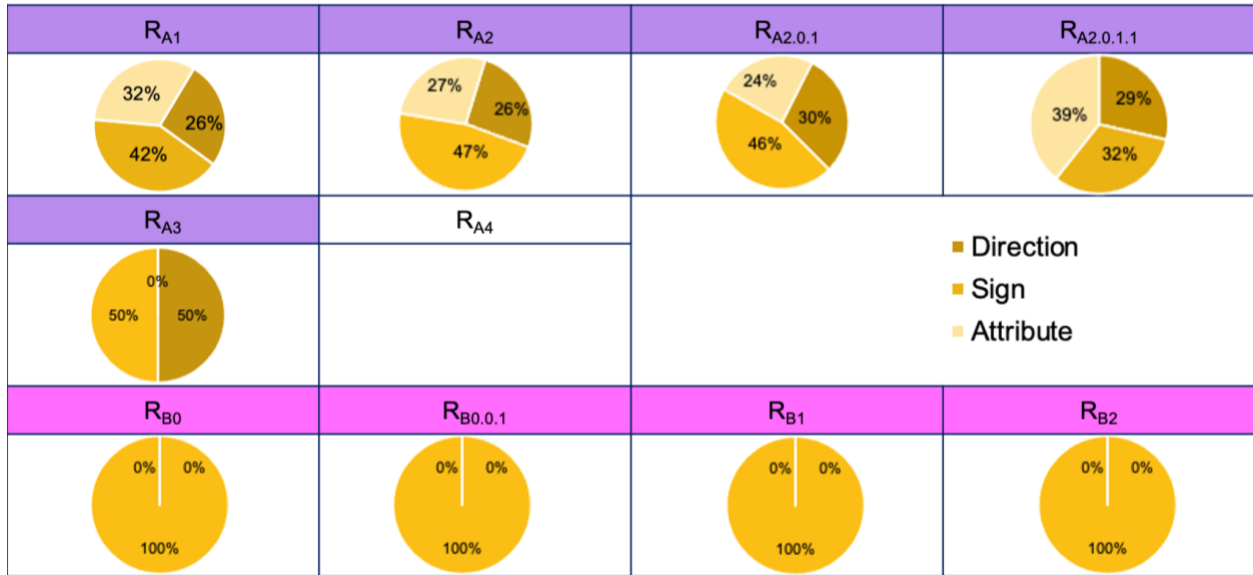


Figure 4-5. The contradiction subclassification judgements of the VIOLIN input.

The predominant subcategory of the flagged LEEs is the type 2 subcategory, an indirect LEE which corresponds to a path with one or more mismatched attributes. Figure 4-6 shows the subclassification breakdowns for individual LEE sets, excepting R_{A4} , as there were no LEEs flagged from this set. This is expected, since the machine readers default to $a^{connectiontype'} =$

“indirect” and self-regulations are not common, reducing the probability for flagged type 1 and 3 subcategories.



Figure 4-6. The flagged subclassification judgements of the VIOLIN input.

For the two “negative” LEE sets, R_{B*1} and R_{B*2} , since they were extracted from breast cancer-related literature, different from the context of the T cell baseline model, we expected few to no corroborations, few to zero contradictions, and only extensions in the VIOLIN output. The VIOLIN judgements of these two sets are illustrated in Figure 4-7. The R_{B*1} category distribution shows this perfectly, with only extensions in the VIOLIN output, and with 98% of those extensions as full extensions, not connecting to the baseline model in any way. The remaining 2% of extensions are hanging extensions and involve proteins which are found in many biological processes, such as the TGF-beta protein, and therefore, have one of the nodes in the baseline model. The R_{B*2} set includes 99% extensions, however there is also a small number of corroborations in this set (0.3%). This is due to the “inflammation” search term included in the query for the R_{B*2} set, which is also part of the immune system response. The remaining LEEs were flagged type 2.

Similar to R_{B*1} , 93% of the R_{B*2} extensions are full, and 7% are hanging extensions involving common biological entities (e.g., “mTOR promotes cell growth”).

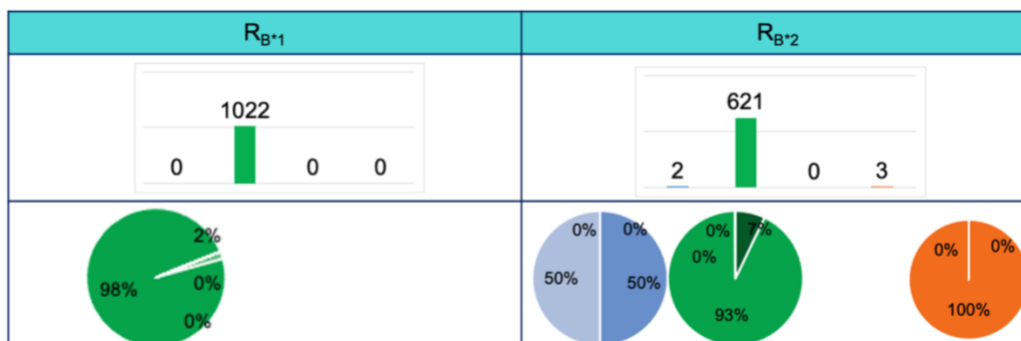


Figure 4-7. Classification breakdown and subclassification judgements for our two “negative” LEE sets

4.3 Automated Curation of Extracted Events

We explored the influence of curation on VIOLIN’s output, and we also compared the VIOLIN output for manually and automatically curated LEE sets.

We consider LEE sets R_{A1} , R_{A3} , R_{A4} , R_{B1} , and R_{B3} as “raw” reading output, since they are obtained directly from the machine reader without manual curation and removal of LEEs. These sets are the fastest to assemble, but the most likely to have reading errors or irrelevant interactions. Even just observing the R_{A1} LEE classification in Figure 4-4, the percentage of full extensions is much higher than in the classifications of the LEE sets R_{A2} , R_{A3} , R_{A4} , which are more closely tailored to the baseline model context.

As described in Section 4.1, the selection of papers from which sets R_{A2} and R_{B2} were extracted was guided by experts. In particular, the R_{A2} set was created using a larger number of

query terms, while R_{B2} set was created using citations of the paper that describes the baseline model. Since the LEE sets $R_{A2.0.1}$, $R_{A2.0.1.1}$, and $R_{B2.0.1}$ were obtained by manually curating and filtering the original sets, R_{A2} and R_{B2} , thus relying on expert knowledge and manual input, they have a smaller number of LEEs and required a longer time to create. Figure 4-3 and Figure 4-5 show that this careful selection for papers or LEEs that are relevant to the baseline model produces a higher percentage of corroborations and contradictions. Figure 4-4 shows that there is plenty of room for extending the melanoma baseline model. Moreover, in the $R_{A2.0.1.1}$ LEE set, which was specifically created to be relevant to the model, most of the LEEs were classified as extensions. Figure 4-2 shows the difference in VIOLIN output between sets obtained using guided selection of papers, without any manual curation (R_{A2}) and one with manual curation ($R_{A2.0.1.1}$).

In comparing the classification outcomes of LEE set R_{A2} to the manually curated $R_{A2.0.1.1}$, we found that the corroborations and contradictions had the highest retention percentage (Figure 4-3 and Figure 4-5). As expected, all strong corroborations were retained. Of the extensions, full extensions had the lowest retention, suggesting that the full extensions are the least likely to be useful. Furthermore, in some cases full extensions result from grounding errors, when one or both of the LEE nodes are improperly grounded and classified as “new” to the model.

4.4 Classification Strategies Guided by Modeling Goals

As can be concluded from Figure 3-3, a number of decisions, mostly related to interaction attributes, need to be made in order to inform VIOLIN’s classification algorithm. These decisions are not fixed and can often vary depending on the modeled system, the goals of modeling, or user’s preferences. VIOLIN allows users to choose how some of the classification decisions are made.

While some of the interaction attributes are critical for the classification (e.g., sign, as discussed in Section 3.1.3), and thus need to be considered, we were interested in exploring how much the decisions related to other attributes will affect the classification outcome. In Figure 4-8, we show the percentage of LEEs in several reading outputs where $a^{connectiontype}' = \text{“direct”}$, LEEs where $a^{location}'$ is non-“empty”, and LEEs where $a^{mechanism}'$ is non-“empty”. Although the information about novel direct interactions, interaction location, and mechanism is in general very useful for modeling, the numbers in Figure 4-8 suggest that machine readers have a hard time extracting such information. Hence the depth at which VIOLIN can make comparisons is constrained due to machine reader limitations, leading to rare strong corroborations and specifications.

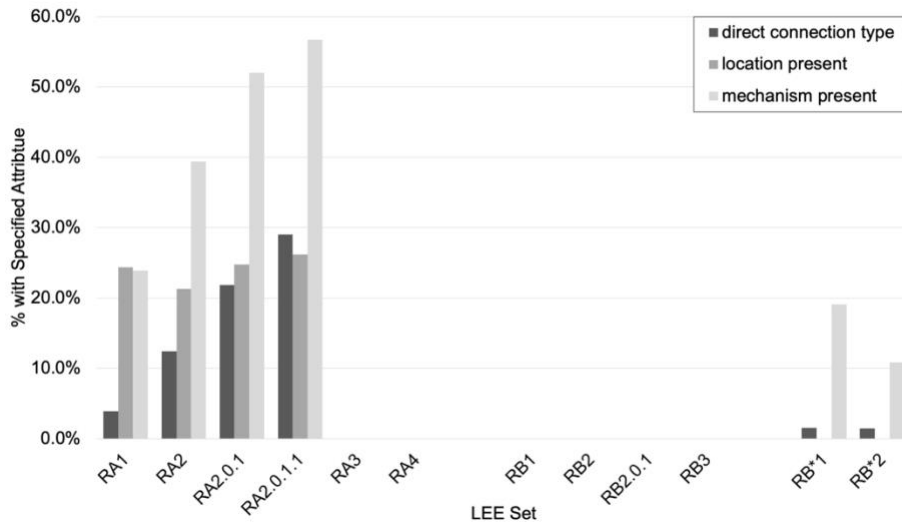


Figure 4-8. Overview of attributes for different scenarios: attribute availability within the LEE sets observing what percentage had a “Direct” connection type (dark grey), a present location attribute (mid grey), and a present mechanism attribute (light grey).**Pathfinding Discovery of Loops**

When enough information is available, machine readers can include the connection type attribute $a^{connectiontype}'$ as part of an LEE, indicating whether the LEE represents direct or indirect interaction between its source and target nodes. In models, if the connection type is

available, VIOLIN will use that information to set the $a^{connectiontype'}$ attribute and will use it in the classification. If there is no information about the connection type for model interactions, in our studies, we used $a^{connectiontype="direct"}$ as default. We made such choice as most baseline models are manually assembled by experts, and the likelihood of interaction ambiguity is low. However, if the omission of the connection type information is due to the lack of knowledge about the modeled system, the user can choose to set indirect connection type as a default.

With the $a^{connectiontype'}$ attribute, VIOLIN can more accurately make judgements of LEEs with respect to a model. In Figure 3-3, we show how the classification outcomes differ due to the connection type. For an LEE that satisfies the necessary condition to match a baseline model interaction, the two interactions have the same connection type, and there is no mismatch in the remaining attributes, we can assume that they are the same interaction, and VIOLIN classifies this LEE as a strong corroboration of the model. If all the conditions are the same, except that the LEE has a direct connection type and the model interaction has an indirect connection type, VIOLIN classifies this LEE as a specification, because a known direct relationship between two elements is less ambiguous. In the converse case, if the LEE is indirect but the model interaction is direct, VIOLIN classifies this LEE as a weak corroboration, because an indirect interaction is not as specific as a direct relationship. Without the $a^{connectiontype'}$ attribute, these distinctions would be lost, and there would be more false corroborations or fewer specifications.

Another important benefit of using $a^{connectiontype'}$ attribute is that it enables identification of paths, feed-forward and feedback loops in the baseline model, and between the baseline model and LEE sets. VIOLIN searches for paths when both the source and target from an LEE are found in the model, but there is no corresponding model interaction. As of right now, VIOLIN only judges the shortest path between the source and target. When searching for paths in the model,

there are several possible outcomes. The first is that an indirect LEE has a matching path in the model, which can be interpreted as a corroboration of that path. This is a neutral outcome, as it neither adds new information to the model, nor does it suggest errors in the model or in machine reading. In particular, such LEEs are classified as weak corroboration type 3, and as discussed previously and shown in Figure 4-3, this is the most prevalent corroboration subcategory in the studies that we conducted. Next, when the source and target node of a direct LEE match the beginning and end nodes of a path in the model, this indicates that there are potentially more direct relationships between model nodes and adding such an LEE to the model forms a feed-forward loop. This case is identified as an internal extension, and while it is the second-least common type of extension after specifications (Figure 4-4), it is important to be able to identify it when it appears.

Finally, if there is an indirect LEE whose attributes mismatch a path in the model, this is flagged for review, although it is possible that the LEE is just representing a path different from the one that exists in the model. In such cases, VIOLIN allows users to decide how to treat the flagged LEE, that is, whether to include the conflicting indirect LEE as a baseline model extension or reject it to avoid complicating the model. These conflicting indirect LEEs are flagged, with type 2 subcategory, and as seen in Figure 4-6, they are the most prevalent flagged subcategory in our studies. In Chapter 5.0, we investigate how these attributes influence the classification scheme.

4.6 Reading Errors and VIOLIN output

As described in Section 2.5.2.3, four types of errors are typically found in the machine reading output of biomedical literature, namely sign, direction, grounding, and omission errors.

Sign errors are most common in phosphorylation interactions, as machine readers make the assumption that phosphorylation is always a positively regulating interaction. From Figure 4-5, which shows a majority of contradictions are sign contradictions (with the exception of $R_{A2.0.1.1}$ LEE set), it follows that contradictions which are phosphorylation interactions are likely candidates for reading errors.

Omission errors and grounding errors can both lead to wrong LEE elements. In turn, this can result in VIOLIN either inducing that an element is “new” with respect to the baseline model when there is a model element matching the true element, or in matching the element with a model element when the true element should be declared “new”. In the former case, VIOLIN will classify the LEE as an extension. However, in the latter case, VIOLIN may be able to single out reading errors through the contradiction and flagged classification categories.

To explore VIOLIN’s ability to identify reading errors, for the LEE sets R_{A1} and $R_{A2.0.1.1}$, we manually judged the presence of machine reading errors in the contradiction and flagged output, and for the flagged classification we additionally assigned all non-erroneous LEEs to an appropriate classification. We wanted to investigate the potential of these classifications to identify machine reading errors, as well as justify the use of the flagged classification. We chose the R_{A1} set as it is the most general and would give us an idea of the overall prevalence of machine reading errors, and we chose $R_{A2.0.1.1}$ because it would show us how many machine reading errors are able to slip through even expert judgement. We illustrate the outcomes of this investigation in Figure 4-9.

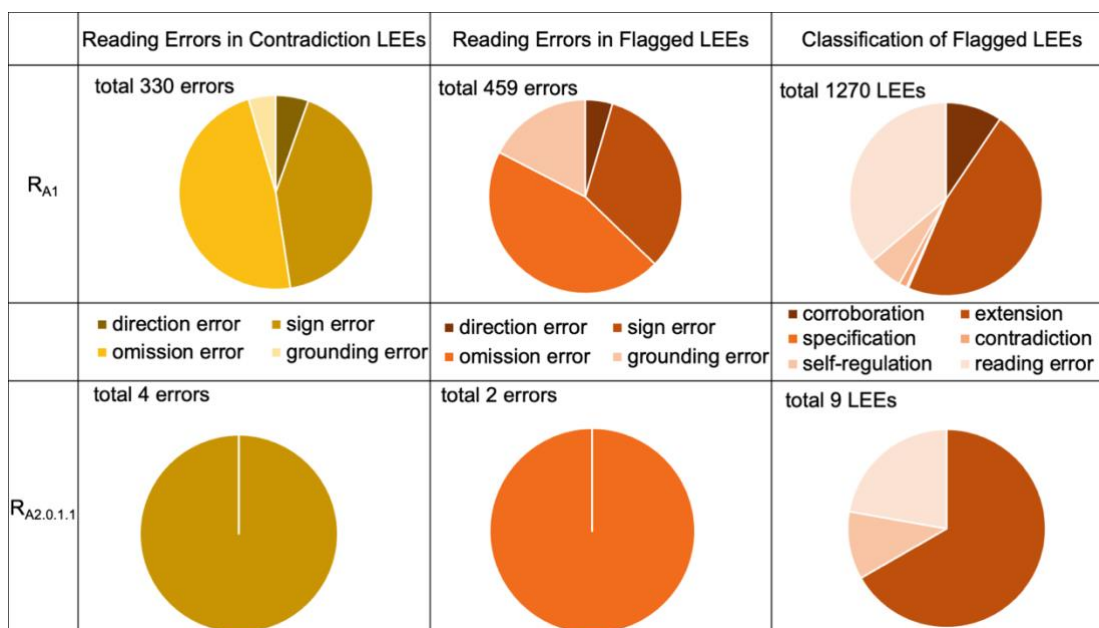


Figure 4-9. Prevalence of machine reading errors in subcategories of contradiction and flagged categories, found manually in two LEE sets: R_{A1} and $R_{A2.0.1.1}$.

As expected, the manually curated set $R_{A2.0.1.1}$ had fewer errors compared to the non-curated set R_{A1} . The most common reading error found in both contradictions and flagged interactions within the R_{A1} set was an omission error (Figure 4-9). None of the reading errors from the $R_{A2.0.1.1}$ contradictions were classified as omission errors, though all of the flagged reading errors were omissions (Figure 4-9). We found that many omission errors come from “negative language” in the text such as “We couldn’t find that...”, “...did not regulate...”, “...failed to regulate...”, “...was regulated by [one protein] but not [the other]”. The other patterns we found in the omission reading errors were hypothesis statements, and element adjectives. The machine readers did not seem to distinguish between “*mutated* [entity]”, “*wild-type* [entity]”, and “[entity]” without descriptors. VIOLIN is capable of identifying these errors by classifying them as contradictions when both elements from the interaction are already in the model. When either the

regulator or regulated is not in the model, the interaction is automatically classified as an extension, and would require manual inspection to identify an omission error.

Grounding errors made up 5% of the R_{A1} contradictions, 17% of the R_{A1} flagged, and 0% of the $R_{A2.0.1.1}$ contradictions or flagged (Figure 4-9). We found two distinct grounding errors in the VIOLIN output. First were LEEs which contained short abbreviations. For example, m2 macrophages were grounded to the PubChem ID for phthalic acid. Second, interactions where pH regulates the activity of an element almost always produced reading errors, such as “acidic extracellular pH causes rapid intracellular acidification and il-1beta-inducing effects” [126]. In these cases, pH is grounded as pancreatic prohormone.

The second most common reading error in the R_{A1} contradictions and flagged interactions were sign errors (42% LEEs in contradictions and 33% LEEs in flagged output), and the least common were direction errors (5% LEEs in both the contradictions and in the flagged) (Figure 4-9). The $R_{A2.0.1.1}$ output contained four sign errors only in the contradictions and no direction errors in either the contradictions or flagged output (Figure 4-9). These errors almost exclusively result from vague language, such as interaction that describes the inhibition of another inhibitory interaction. Another common sign error came from phosphorylation interactions. Phosphorylation is assumed to be a positive regulatory process, but as is the case of TSC2 phosphorylation by AKT, it can also be an inhibitory interaction. This supports our assertion that contradictions which are phosphorylation interactions are likely machine reading errors.

Figure 4-9 also shows the outcome of the manual classification of the flagged LEEs. The mixed distribution of classifications from the flagged LEEs shown in Figure 4-9 supports the need for this classification – the LEEs are too varied to be able to fully automate their classification without creating misclassified LEEs, though VIOLIN has been created with capability to classify

these LEEs as one of the other categories if the user chooses. While R_{A1} flagged LEEs produced the greatest number of errors, the R_{A1} contradictions had a greater percentage of reading errors, 49% compared to the total number of contradictions. The most common types of errors found were sign and omission errors. This information can help guide users in choosing LEEs for consideration.

5.0 Further VIOLIN Evaluation

We investigated the VIOLIN output from changing the classification parameters and input. In the previous chapter, we showed that VIOLIN can be effective on two different systems with a variety of LEE sets. We also showed the utility of the contradiction and flagged classifications, and how they can be used to identify reading errors. From these initial results, we found that it was important to further investigate the choices we made in the initial VIOLIN implementation. Here, we investigate how the Match Score is affected by different values (Section 5.1), how the Kind score is affected by different values (Section 5.2), how the classification scheme affects VIOLIN outcomes (Section 5.3), corroboration outcomes (Section 5.4), and extension outcomes (Section 5.5).

For our investigations into corroboration and extension classifications, we used models C (the BDNF pathway) and D (pancreatic cancer system), described in Section 4.1, and their reference LEE sets, R_{C0} and R_{D0} . We also altered our input baseline models, removing specific interactions creating models C' and D', to investigate how well VIOLIN could recreate pathways from the reference literature (Section 5.5). Along with providing the context for our corroboration and extension testing, these inputs further show that VIOLIN is applicable to a wide range of systems and model types.

5.1 Match Score Testing

5.1.1 Match Score Parameters

To test the Match Score parameters, we used model A and LEE set R_{A2} as our input. Model A, already validated, acts as an ideal case for comparison, as we know that model interactions have been judged for biological accuracy. The LEE set R_{A2} is a large set, and a good representation of typical user input: curated to be highly relevant and lots of candidate information. Our classification scheme investigation makes use of several of the LEE sets introduced in Chapter 4.0, judged against models A and B.

In Section 3.2.2, we defined the Match Score values as follows: $S_{M1} = 10$ (both LEE entities are found in the model), $S_{M2} = 1$ (LEE introduces a new regulated element), $S_{M3} = 100$ (LEE introduces a new regulator element), and $S_{M4} = 0.1$ (neither LEE entity is also found in the model). In other words, we ranked the Match cases as follows: $M3 > M1 > M2 > M4$. This scoring is based on the formula $S_{Mx} = 0.1 \times Y^x$, where x is from the set $\{0, 1, 2, 3\}$ and Y is the evidence score required to move an LEE's score from one ranking to another. In the case of our presented Match Score, $Y=10$, so an LEE must have an Evidence Score of 10 (i.e. be found in the LEE 10 times) for its score to increase to the next highest rank. We present this as a set of default scoring values for general model extension, showing preference towards those LEEs with introduce new regulators or new information and interactions between existing model nodes. This default also works as a type of filtering method; LEEs with higher evidence scores are generally assumed to be a higher quality, and this formula considers that factor in determining the Match Score.

To test the impact that the score values have on the judgement outcome, we tested 20 different sets of values to represent these criteria, shown in Table 5-1. These values were chosen on several different methods. Value sets B-I were calculated by the formula $SM = Z^x$, where x is the ranking, and Z is an integer from 2 to 9. Value sets J-N were given as strict ranking (1,2,3,4), and then multiplied by factors 5, 10, 25, or 100. Value sets O-R were created as “controlled randomness” sets, maintaining the ranking order, but altering the differences between ranking values. And finally value sets S and T are for different ranking systems, where for S: $M1 > M3 > M2 > M3$, and for T: $M3 = M1 > M2 > M4$. Value set S represents the use case where the user desires to extend the model downstream of the input nodes, and value set T represents the use case where the user wants to add as many new nodes to the model as possible.

For this investigation, we calculated the Total Score as

$$S_T = S_E \times S_M, \quad 5-1$$

to better analyze how the tested score values affected Match Score outcomes independent from other factors. Since we want the Match Score of an LEE to be affected by its Evidence Score, we include this factor, instead of considering the Match Score alone.

Table 5-1. Match value sets used to test the Match Score

Element in Model	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
Both	10	4	9	16	25	36	49	64	81	3	15	30	75	300	7	13	19	79	4	10
Source	100	8	27	64	125	216	343	512	729	4	20	40	100	400	11	19	31	31	3	100
Target	1	2	3	4	5	6	7	8	9	2	10	20	50	200	5	7	11	37	2	100
Neither	0.1	1	1	1	1	1	1	1	1	1	5	10	25	100	3	3	3	3	1	1

5.1.2 Match Score Results

From our tested Match Score value sets, we took the top 100 scored LEEs (calculated using the total score definition in Equation 5-1) and compared the LEEs in this subset pairwise for each value set. Figure 5-1 shows the percent overlap for each pair of outputs. From this, we found an exact overlap between value sets A, F, G, H, and I, and upon closer inspection, we find the VIOLIN outputs were identical for each of these value sets, excepting the exact total score values. Similarly, we found the output for value sets J, K, L, M, and N to be identical to each other. Value set S has the least amount of overlap with any other value set, which is to be expected, since it was chosen with a different use case and ranking in mind.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
A	100	75	98	82	98	100	100	100	100	68	68	68	68	68	70	70	70	54	40	58
B	75	100	76	76	75	75	75	75	75	82	82	82	82	82	84	84	84	61	54	58
C	98	76	100	82	96	98	98	98	98	68	68	68	68	68	70	70	70	54	40	58
D	82	76	82	100	83	82	82	82	82	68	68	68	68	68	70	70	70	54	40	58
E	98	75	96	83	100	98	98	98	98	68	68	68	68	68	70	70	70	55	40	58
F	100	75	98	82	98	100	100	100	100	68	68	68	68	68	70	70	70	54	40	58
G	100	75	98	82	98	100	100	100	100	68	68	68	68	68	70	70	70	54	40	58
H	100	75	98	82	98	100	100	100	100	68	68	68	68	68	70	70	70	54	40	58
I	100	75	98	82	98	100	100	100	100	68	68	68	68	68	70	70	70	54	40	58
J	68	82	68	68	68	68	68	68	68	100	100	100	100	100	98	94	94	71	72	72
K	68	82	68	68	68	68	68	68	68	100	100	100	100	100	98	94	94	71	72	72
L	72	74	72	72	72	72	72	72	72	100	100	100	100	100	98	94	94	71	72	72
M	68	82	68	68	68	68	68	68	68	100	100	100	100	100	98	94	94	71	72	72
N	68	82	68	68	68	68	68	68	68	100	100	100	100	100	98	94	94	71	72	72
O	70	84	70	70	70	70	70	70	70	98	98	98	98	98	100	94	94	72	70	72
P	70	84	70	70	70	70	70	70	70	94	94	94	94	94	94	100	100	67	70	66
Q	70	84	70	70	70	70	70	70	70	94	94	94	94	94	94	100	100	67	70	66
R	54	61	54	54	55	54	54	54	54	71	71	71	71	71	72	67	67	100	52	56
S	40	54	40	40	40	40	40	40	40	72	72	72	72	72	70	70	70	52	100	54
T	58	58	58	58	58	58	58	58	58	72	72	72	72	72	72	66	66	56	54	100

Figure 5-1. Number of interactions in common in the top 100 scored (greater than 50% overlap is in orange, greater than 90% overlap is in green)

We then analyzed the scoring distributions of our tested Match Score value sets, using the total score defined in Equation 5-1, and then normalizing all total score values for ease of comparison. Figure 5-2 shows the Total Score distributions for each Match Value set. The plot was normalized to account for the wide range of values and allow for easier comparison between Value sets. The figure shows how tightly these distributions are, with value sets S and T showing the greatest outliers.

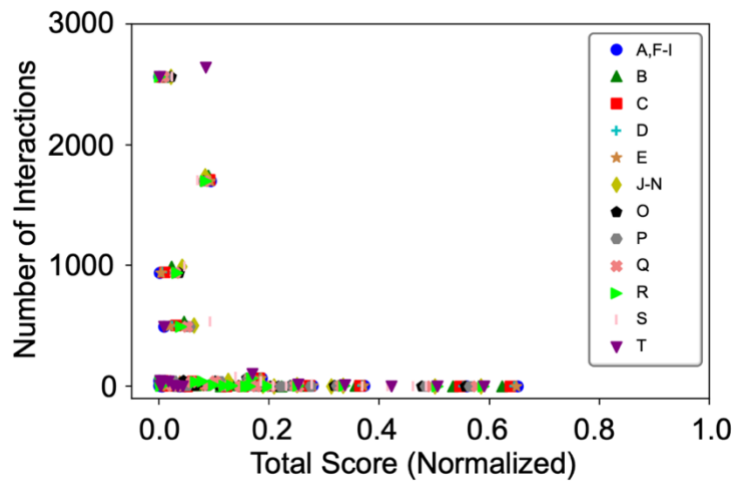


Figure 5-2. Normalized total scores for each of our tested Match Level parameters. Parameters A,F-I were found to be identical to each other, and parameters J-N were found to be identical to each other. Kind Score

Testing

5.2.1 Kind Score Parameters

Following from the Match Score parameterization in Section 5.1, we used model A and LEE set R_{A2} for our Kind Score parameterization as well. The Kind Score values were previously defined in the following way: $S_{K1.1} = 2$, $S_{K1.2} = 1$, $S_{K2.1} = 40$, $S_{K2.2} = 30$, $S_{K3} = 10$, and $S_{K4} = 20$. Again, these values were chosen for the default use case of general model extension, showing

preference towards extensions and flagged interactions, which may be added to the baseline model, and then towards contradictions, which may indicate changes need to be made to the baseline model, and finally corroborations, which generally aren't needed for model extension, unless there exist some novel interactions in the baseline which it would be helpful to validate during extension.

Here, we tested six score value sets shown in Table 5-2, based off the results of the Match Score parameterization, shown in Section Match Score Testing. Value sets A and B are strict rankings, multiplied by a factor of 10 or 5, representing the median Match value orders of magnitude. Value sets C and D use the same ranking order, but those categories of higher importance to model extension were given scores based on the higher median Match value, and the corroborations are devalued to the lower median Match Value. Value sets E and F created categories, based on whether the LEE category is “more” or “less” useful. Value set E judges the categories in pairs: strong corroborations are more useful than weak, extensions are more useful than specifications, etc. Value set F judges the categories overall, deciding that extensions and contradictions are the most useful for model extension, specifications and flagged are of middling usefulness, and corroborations are least useful.

We based these values on the median values of the Match Score so that the Kind Score can enhance the Total Score, but not dominate it. For this investigation, we calculated the Total score as defined in Equation 3-1.

Table 5-2. Kind value sets used to test the Kind Score

Category	A	B	C	D	E	F
Strong Corroboration	20	10	2	2	20	1
Weak Corroboration	10	5	1	1	10	1
Extension	60	30	40	50	20	10
Specification	50	25	30	25	10	50
Contradiction	30	15	10	10	10	10
Flagged	40	20	20	20	20	5

5.2.2 Kind Score Results

We also took the top 100 scored LEEs from our tested Kind Score value sets, this time using the standard VIOLIN Total Score, defined in Equation 3-1; the pairwise comparisons are shown in Figure 5-3A. For the Kind Score, there is more than 60% overlap between any two tested value sets. The value sets with the lowest overlap are E and F, and this can be explained by the reduced number of category values in these sets – two possible values in set E and three in set F, compared to the six values in sets A-D. Figure 5-3B shows the distributions over the Total Score, and the full distributions match the trends of the top 100 scored. In this case we plotted the Total

Score distribution using the raw values, as the range in values was not as wide as for the Match Score outcomes.

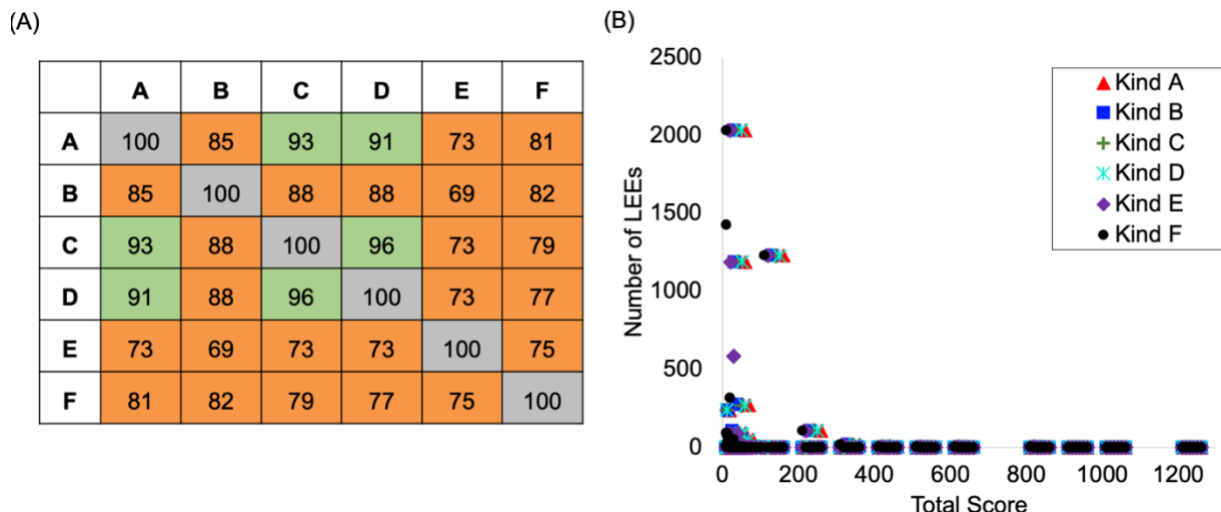


Figure 5-3. Results from the Kind Score Parameterization: (A) shows the overlap in the top scored (> 90 in green, >50 in orange) (B) shows the total score distribution. Classification Scheme testing

Besides the classification scheme that we showed in Figure 3-3, we created two additional classifications schemes; all three are shown in Figure 5-4. Scheme V1 was created assuming that a direction mismatch (i.e. the LEE source node is found in the baseline model as the *target* of the LEE target node) may possibly identify a feedback loop, and should be flagged for review, unless the additional attributes are also mismatched. This scheme also leaves the interpretation of paths compared to direct LEEs mainly up to user judgement, classifying a direct LEE with elements matching the source and target nodes of a path in the model as a flagged LEE. In the case of indirect LEEs, this scheme assumes that mismatched attributes indicated a contradictory LEE.

Scheme V2 made different assumptions about LEEs with correspond to model paths. Because of the difficulty with which machine readers identify direct LEEs (in that the language

usually has to be very clear and easily identified), and taking into account that biologically, this case may identify a feed-forward loop or provide more accurate information on the interaction between two elements, we chose to judge these interactions as extensions instead of flagging them for manual review.

Scheme V3 made different assumptions on the meaning of mismatched attributes: for direct LEEs, attribute mismatches were assumed to be contradictory unless *all* attributes mismatched, because in this last case the argument can be made that with so few similarities, the LEE is likely a completely different interaction than presented in the model, and so should be flagged for review. In the case of indirect LEEs which correspond to source and target nodes of a direct model interaction, this scheme made the assumption that sign attribute mismatches should be flagged as potential feed-forward loops, but that a direction mismatch should be a contradiction unless the additional attributes (e.g. location) were also mismatches. This decision was made on the idea that if the element locations of the LEE differed from the corresponding model interaction, it's possible

(though not certain) to be a completely new interaction. This scheme defined five separate subcategories of flagged LEEs, the greatest of any of the three schemes.

Scheme V1

Flagged1: Mismatched Direction and non-contradictory Other Attributes with a Direct connection type in the model

Flagged2: Direct LEE with corresponding Path in Model

Scheme V2

Flagged1: Mismatched Direction and non-contradictory Other Attributes with a Direct connection type in the model

Flagged2: An LEE with a corresponding path which has one or more Mismatched Attributes

Flagged3: An LEE which is a self-regulation based on the definition of model element

Scheme V3

Flagged1: LEE with corresponding Direct LEE where *all* attributes are mismatched

Flagged2: An LEE with a corresponding path which has one or more Mismatched Attributes

Flagged3: An LEE which is a self-regulation based on the definition of model element

Flagged4: Indirect LEE with Direct MI and mismatched direction and Other Attributes

Flagged5: Indirect LEE with Direct MI and mismatched sign

MATCH COMPARISON		OUTCOME	
present/absent in both, match	+/-M	Contradiction	
absent in LEE, present in MI	-+	Extension/Specification	
present in LEE, absent in MI	+/-	Strong/Weak Corroboration	
present in both, mismatch	++X	Flagged	
NA	-----		

LEE	MI	Direction	Sign	Other attributes	Classification Scheme 1	Classification Scheme 2	Classification Scheme 3	LEE	MI	Direction	Sign	Other attributes	Classification Scheme 1	Classification Scheme 2	Classification Scheme 3
Direct	Indirect	M	M	+/-M				Indirect	Indirect	M	M	+/-M			
		M	M	-+						M	M	-+			
		M	M	+/-						M	M	+/-			
		M	M	++X						M	M	++X			
		M	X	-----						M	X	-----			
		X	-----	-----						X	-----	-----			
	Direct	M	M	+/-M					Direct	M	M	+/-M			
		M	M	-+						M	M	-+			
		M	M	+/-						M	M	+/-			
		M	M	++X						M	M	++X			
		M	X	-----						M	X	-----			
		X	M	+/-M						X	M	+/-M			
		X	M	-+						X	M	-+			
		X	M	+/-						X	M	+/-			
		X	M	++X						X	M	++X			
		X	X	+/-M						X	X	+/-M			
		X	X	-+						X	X	-+			
		X	X	+/-						X	X	+/-			
		X	X	++X						X	X	++X			
	Path	-----	-----	-----					Path	M	M	+/-M			
	NO	-----	-----	-----					NO	M	M	-+			
										M	M	+/-			
										M	M	++X			
										M	X	-----			
										X	-----	-----			
										NO	-----	-----			

Figure 5-4. The outcome chart of our three tested classification schemes compared side-by-side, including the different flagged subclassification definitions. Like Figure 3-3, this chart shows the possible outcomes of comparing an LEE to an interaction (MI) from a baseline model.

We ultimately found that scheme V1 identified too many indirect LEEs as contradictions compared to paths found in the model, and scheme V3 was too biased against attribute mismatches. Our default scheme V2 strikes a balance between these two schemes, making decisive judgement where possible (and likely) without making too many assumptions about the contents of the LEEs.

Figure 5-5 shows the prevalence of direct LEEs in each VIOLIN category for four LEE sets across our three classification schemes. With less than 1% of corroborations being direct, we can interpret this as LEE connection type having little correlation to the relevance of the LEE. Our

set RA2.0.1.1 has zero direct corroborations, but is arguably the most relevant LEE set, due to its curation method. However, we also note in Figure 5-5 the percentage of direct extensions increases as the LEE set becomes more relevantly curated, suggesting that direct LEEs may more often be *useful* to the model.

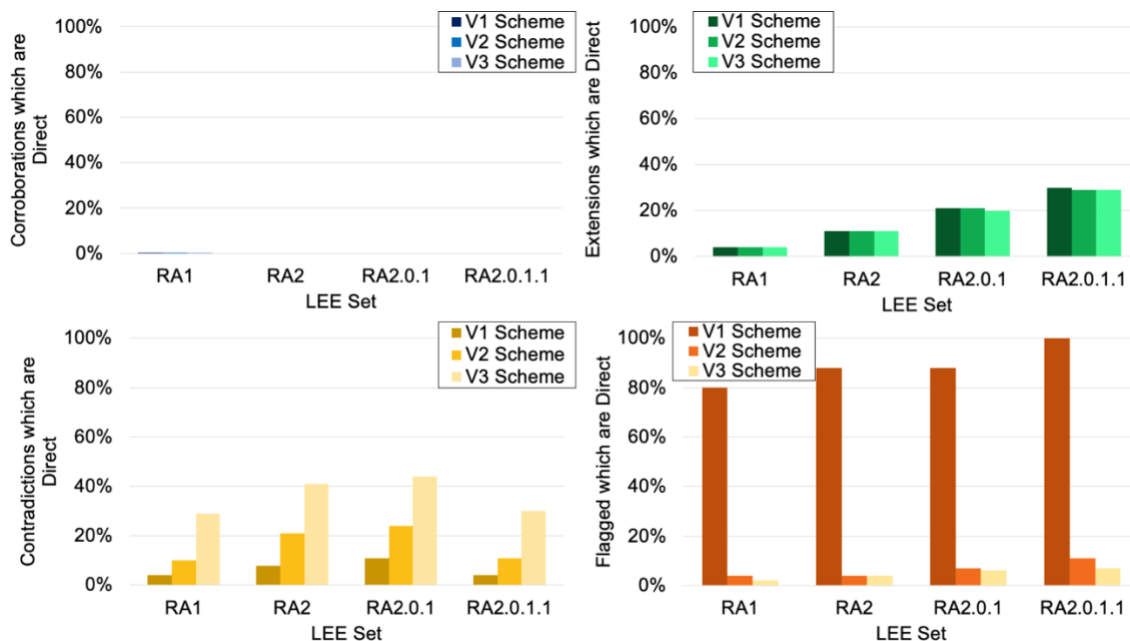


Figure 5-5. The presence of direct LEEs across the main VIOLIN classifications, comparing our three tested classification schemes

The results for the contradictions and flagged LEEs shown in Figure 5-5 demonstrate the importance of the classification scheme choices, and how they relate to facilitating the judgement on paths and the identification of loops. Figure 5-6 looks specifically at the contradictions and flagged LEEs, since those subclassifications are what required the most consideration in creating the classification scheme. This figure illustrates how the judgement choices led to the number of identified contradictions and flagged LEEs. Scheme V1 assumed too many contradictions, but was also not taking into account what is known or common about direct interactions. Scheme V3 relied

too much on manual judgement. We developed scheme 2 based on what is known or common amongst biological interactions (for example the existence of feed-forward loops), while still allowing for interpretation in the flagged LEEs.

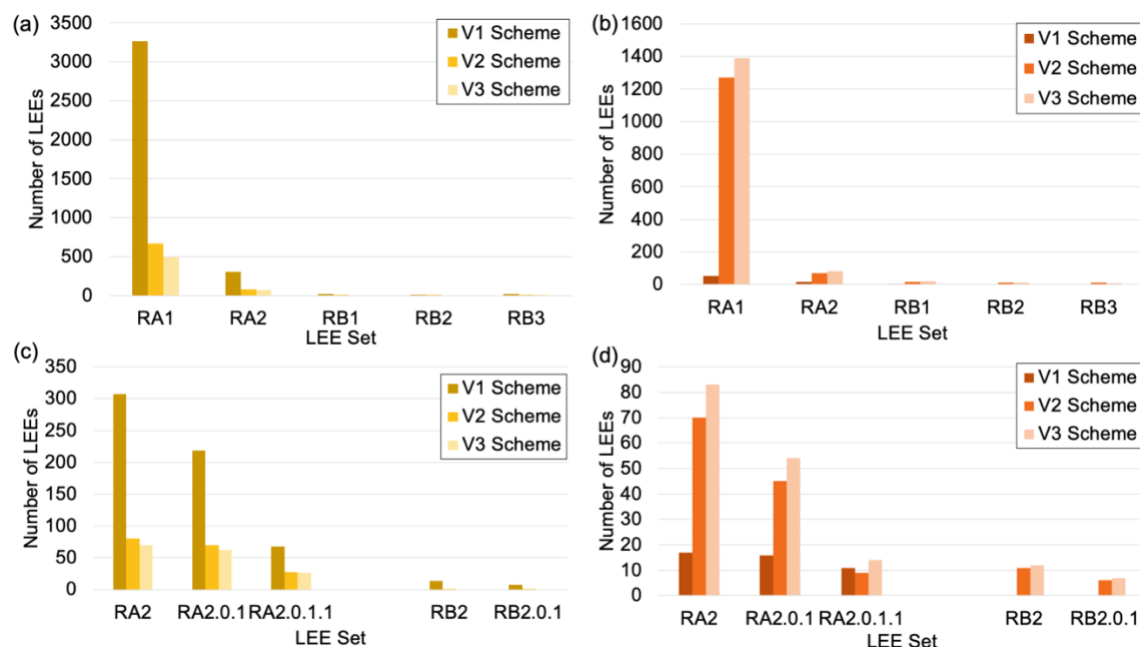


Figure 5-6. The number of LEEs found in contradictions and flagged classifications across our 3 classification schemes: (a) shows the number of contradictions in the “raw” sets, (b) shows the number of flagged in the “raw” set, (c) shows the number of contradictions comparing the filtering methods to their associated “raw” LEE sets, and (d) shows the number of flagged LEEs, comparing the filtering methods to their associated “raw” LEE sets

Figure 5-7 shows the occurrence of phosphorylations across each category for our three classification schemes compared to Figure 4-5, which shows a majority of contradictions are sign

contradictions (with the exception of RA_{2.0.1.1} LEE set), it follows that contradictions which are phosphorylation interactions are likely candidates for reading errors.

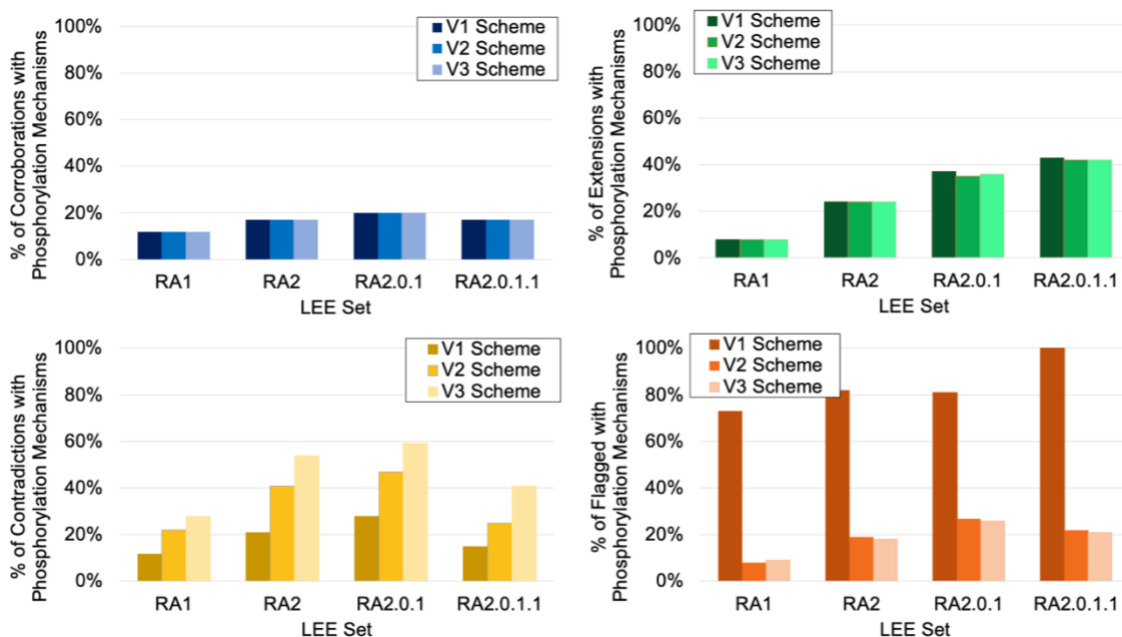


Figure 5-7. The presence of phosphorylations across the main VIOLIN classification, comparing our tested classification schemes.

5.3.1 VIOLIN Speed

One of VIOLIN's biggest assets is its speed. We investigated the processing time of VIOLIN inputs for each of our classification schemes. Figure 5-8 shows the average processing time of VIOLIN, broken down into LEE set and model input, LEE classification, and LEE output. The smallest LEE sets shows the slowest input timing (Figure 5-8A), which can be attributed to a baseline time required to open the LEE files. Model input, and even model conversion into the directed graph used in the VIOLIN path-finding methods, takes only a fraction of a second (Figure 5-8B) for both of our models, showing that VIOLIN can handle models of various sizes. The

average classification time for the default scoring scheme is ~1000 LEEs/sec (Figure 5-8C). Assuming an expert can judge an LEE in 30 seconds, that puts VIOLIN at 30,000 times faster than an expert. A typical user would take closer to 60 seconds to judge a single LEE, making VIOLIN 60,000 times faster in its judgement. The timing outcomes are even better when considering the time it would take an average user to find a path in the model to correspond to a given LEE. Once the LEEs are classified, VIOLIN quickly assembles the information into user-friendly spreadsheets (Figure 5-8D).

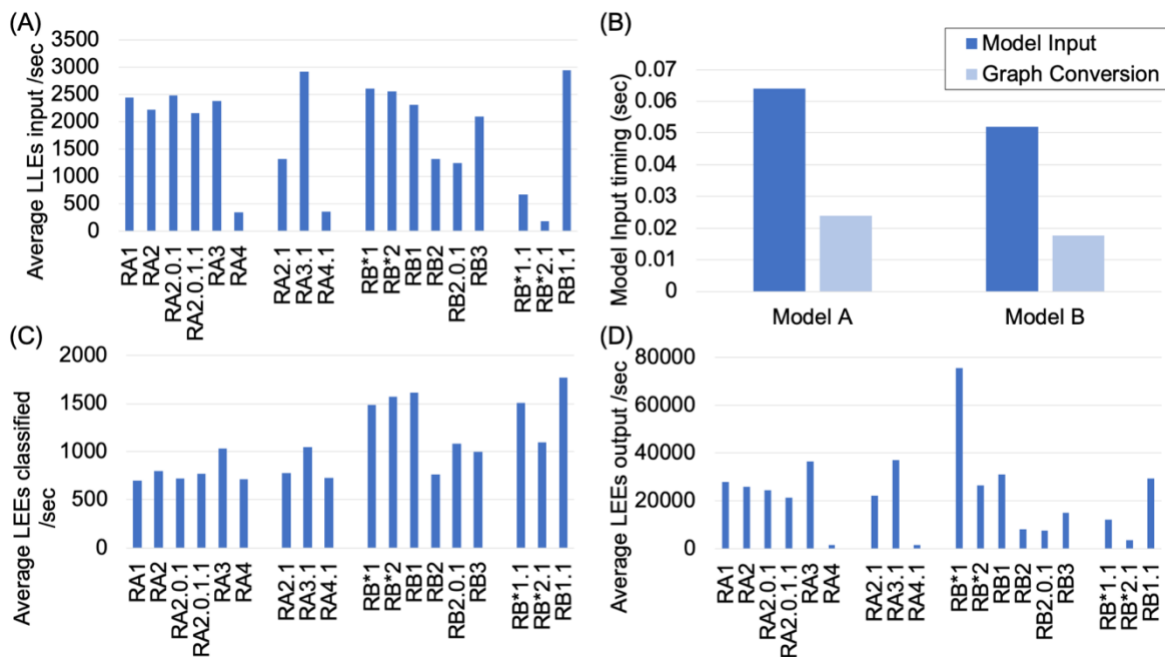


Figure 5-8. The average processing time of VIOLIN.

(A) shows the input rate for each LEE set, (B) shows the time it takes to input out models and convert them into directed graphs for path finding, (C) shows the classification rate for each LEE set, and (D) shows the output rate for each LEE set

In general, three major factors of LEE processing speed are: the classification scheme, the number of attributes compared, and model size. In Figure 5-9, classification scheme V2 shows the

fastest LEE processing time. This is due to how paths and flagged interactions are treated. In classification V1, paths are judged more strictly by their attributes, determining what sub-classification of corroboration or contradiction they are. In V2, a mismatched attribute between the path and the LEE is a flagged type 2 case, regardless of which attribute is mismatched. Classification V3 is slowed down by the increase in flagged sub-categories defined in Figure 5-4, as shown in Figure 5-9. Model A is bigger than model B, meaning that the LEE sets have to be compared to a larger knowledge base, and when VIOLIN is searching for paths, it takes longer to find a path through a larger model. The R_B LEE sets also have no attributes other than element type, which means VIOLIN is skipping much of the attribute comparison steps, making the LEE judgement time even faster.

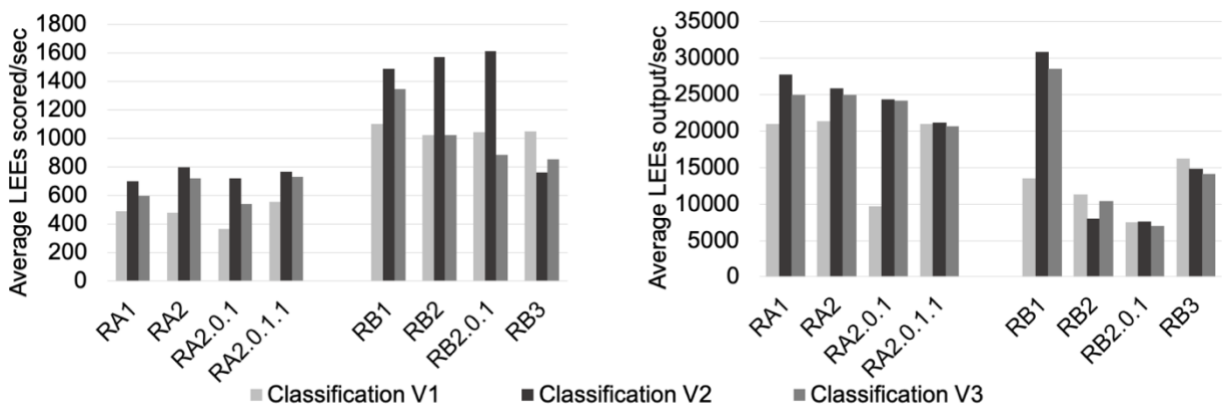


Figure 5-9. Average processing times across classification schemes.

Showing the average scoring time (left) and output time (right) for classification schemes V1, V2, and V3.

While VIOLIN classification does not occur one classification category at a time, we can draw conclusions about how the number of computational steps correlates to classification categories. For example, a hanging or full extension (where one or both nodes are new to the model system) requires the fewest computational steps towards classification, while those LEEs that

require checking the model for paths result in the greatest number of computational steps. Pathfinding occurs when both LEE nodes are found in the model, an LEE has an indirect connection type, and there is no corresponding direct interaction in the model.

5.4 Corroboration Testing

To investigate VIOLIN's corroborative abilities, we made use of those LEE sets created from references from the model publications: R_{C0} was created using the references from [125] used to build the model; of the 83 total references, only 28 were accessible by machine readers, and LEE set R_{D0} was created using 19 of the references from [122]. The full details of these LEE sets are included in Appendix B.

Figure 5-10 shows the classification distributions of these two sets. Both sets contain a majority of extensions, which is unsurprising. The amount of information available from a given paper may be more than is needed for the model assembly. In the case of the model C, the model itself was created to relate to major depressive disorder [125], but the BDNF pathway was implications in many biological functions. Interactions from these other pathways were ignored in creating model C. Many grounding errors were found in the reading set, relating to the TRKB protein. This was a central protein to model C assembly [36], and so this grounding error could have created several false extensions. Finally, many of the references used to create model C were not accessible by machine readers, and so the LEE set doesn't not contain all of the information used to create the model. Our LEE set R_{C0} represents 34% of the reference literature for model C, and corroborates, 17% of the total model interactions.

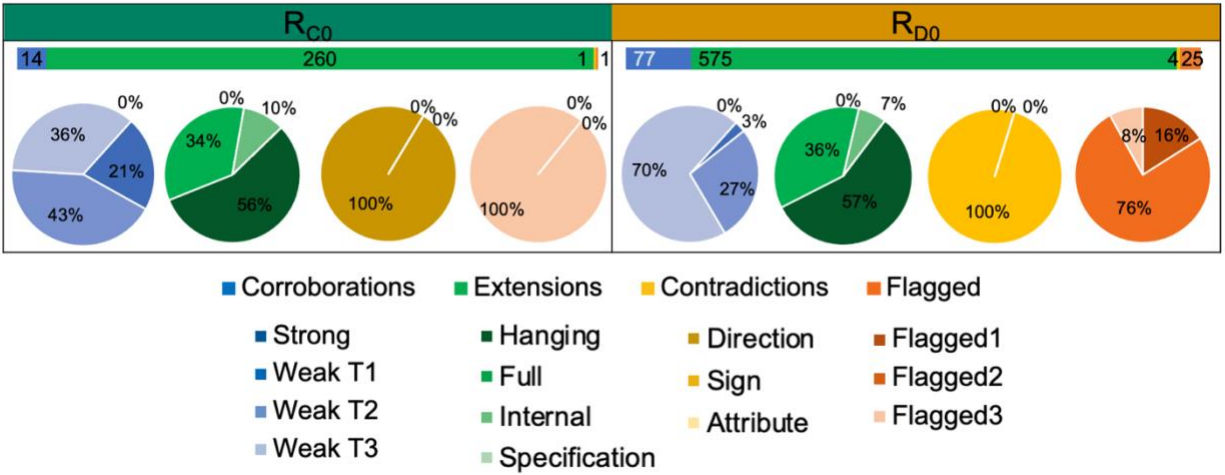


Figure 5-10. Classification Distributions for our two corroborative sets; the LEE sets were judged against the original models C and D, respectively

Similarly, LEE set R_{D0} was created from literature relevant to, but not exclusive to the pancreatic cancer cell. References for this model included general cancer environment data [127] and the ERK pathway [128], which participates in many biological processes. The authors of [122] also used sources outside of the 19 papers used to create R_{D0} in building model D. 25% of the model D interactions were corroborated by the R_{D0} set.

For both LEE sets, we found that the contradictions were caused by reading errors. The flagged LEE from the R_{C0} set was of type 3, or a self-regulation, which was not considered useful by the model creators. The majority of the flagged LEEs in R_{D0} were of type 2, meaning that the LEE corresponded to a path in the model. If we assume all such LEEs to be weak corroborations of the more detailed model path, then these LEEs corroborate a further 6% of model D interactions.

The extensions found in these reading sets give us information on just how much information is generated by machine readers, and also on the choices that experts make during

model assembly. Understanding these factors can help guide a user through utilizing VIOLIN as a corroborative tool.

5.5 Extension Testing

We used again our LEE sets R_{C0} and R_{D0} to test the extension utility of VIOLIN. In this case, we altered our input models, strategically removing interactions, and investigating how easily VIOLIN identifies the missing connections through the assembled information. From model C, we removed the protein NTRK2 which, connects to several other proteins, either directly or through a pathway, and we call this modified model C'. We will investigate if we can recover specifically the connections from BDNF to NTRK1 and NTRK2. From model D, we made use of the modified version presented in [28], where the authors removed the pathway from KRAS to cell proliferation, which we will call modified model D', and we will investigate whether or not this pathway can be recovered.

Figure 5-11 shows how these changes to the models affect VIOLIN outcomes. As expected, the number of corroborations decreases, and the number of extensions increases. Within the extensions, we are able to recover 5 of the 24 connections to NTRK2 originally found in model C, either from LEEs representing the model interaction as originally presented or from an LEE representing the endpoints of what was a model path. This is a promising outcome considering our LEE set R_{C0} represents less than half of the literature used to assemble model C.

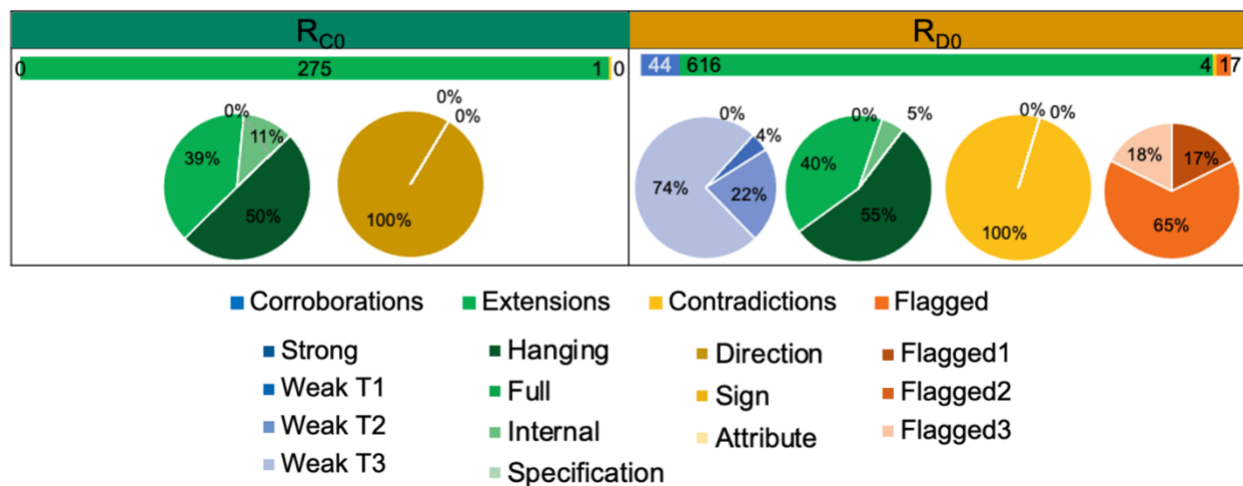


Figure 5-11. Classification Distributions for our two corroborative sets; the LEE sets judged against the altered models C' and D', respectively

From R_{D0} extensions, we were able to recover the full KRAS-proliferation pathway as shown in Figure 5-12. The difference between the pathway steps highlights how different sources represent biological interactions and pathways differently. In which case, it can be important for the user to identify the veracity of an identified pathway. We investigate this more deeply in Chapter 6.0.



Figure 5-12. Representation of the KRAS-Proliferation pathway. (A) Shows the path as it was originally represented in model D and (B) shows how the path was recreated in model D' by the extensions found in R_{D0}

6.0 VIOLIN Variability

In this chapter, we investigate the full utility of the VIOLIN tool. Our presented default values in Section 3.2 are for the general extension use case, but we investigate how different Match Score rankings can be applied to other modeling use cases (Section 6.1). We also investigate how VIOLIN integrates with other automated modeling tools as part of a larger framework, as shown in Figure 6-1. Using filtering methods such as FLUTE [106], we can use expertly curated databases to select LEEs which have verified for accuracy (Section 6.2.1). The output from VIOLIN can be used as input for automated model extension tools such as CLARINET [111] (Section 6.2.2). These tools help automate the model building and extension process, greatly reduce the need for human curation and assembly of knowledge.

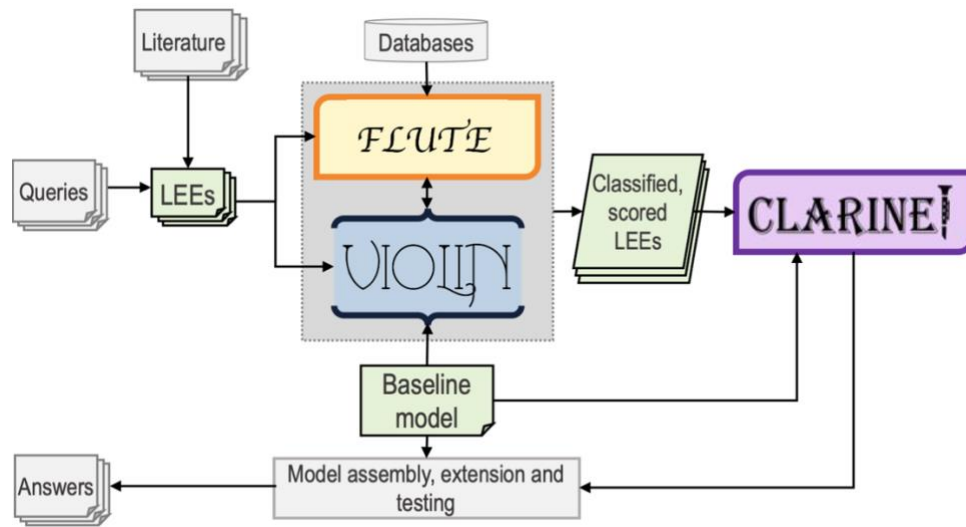


Figure 6-1. How VIOLIN fits in to a larger modeling framework with other automated modeling tools.

To investigate potential default scoring values for other modeling use cases, we used the inputs from the scoring parametrization of Chapter 5.0: model A and LEE set R_{A2} . To investigate

how VIOLIN would integrate with the FLUTE tool, we used a sampling of our benchmark inputs: choosing baseline models A (melanoma), B (T-cells), and C (BDNF pathway), and then choosing 3 LEE sets for each, choosing those that were assembled in the manner of a typical user. We did also include the LEE set $R_{A2.0.1.1}$ for this section, to investigate how an automated filter would compare to expert judgement. And then for the final section of this chapter, we used baseline model G (glioblastoma multiforme), and two associated LEE sets to investigate how VIOLIN output could be used in automated model extension methods.

6.1 Use Cases for Other Scoring Values

We investigated the potential of four default Match Score value sets: our original default value set for general extension, a value set for breadth-first extension (seeking to enlarge the model network), a value set for depth-first extension (seeking to increase the detail of the baseline model), and model validation (for a previously unvalidated baseline model). We outline the value sets for these cases in Table 6-1.

Table 6-1. Match Score values for different use cases

Elements in Model	General Extension	Breadth Extension	Depth Extension	Validation
Both	10	10	100	100
Target	100	100	10	1
Source	1	100	10	1
Neither	0.1	1	0.1	1

In Figure 6-2, we show the outcomes of these different use cases. For the Match Score schemes representing our four use cases, we compared both the total score trends and the scoring values. In the top 100 scored, there was less than 50% overlap between any two Match schemes except for depth extension and model validation. Figure 6-2A shows the pairwise comparison for each use case. This makes sense because both of these use cases rely on LEEs where both the source and target are in the model, which is reflected in the scheme values in Table 6-1. Figure 6-2B shows the overall scoring distributions for each Match Scheme, and these shows more variation than that of the Match and Kind Score value sets, which differed in value but not in ranking.

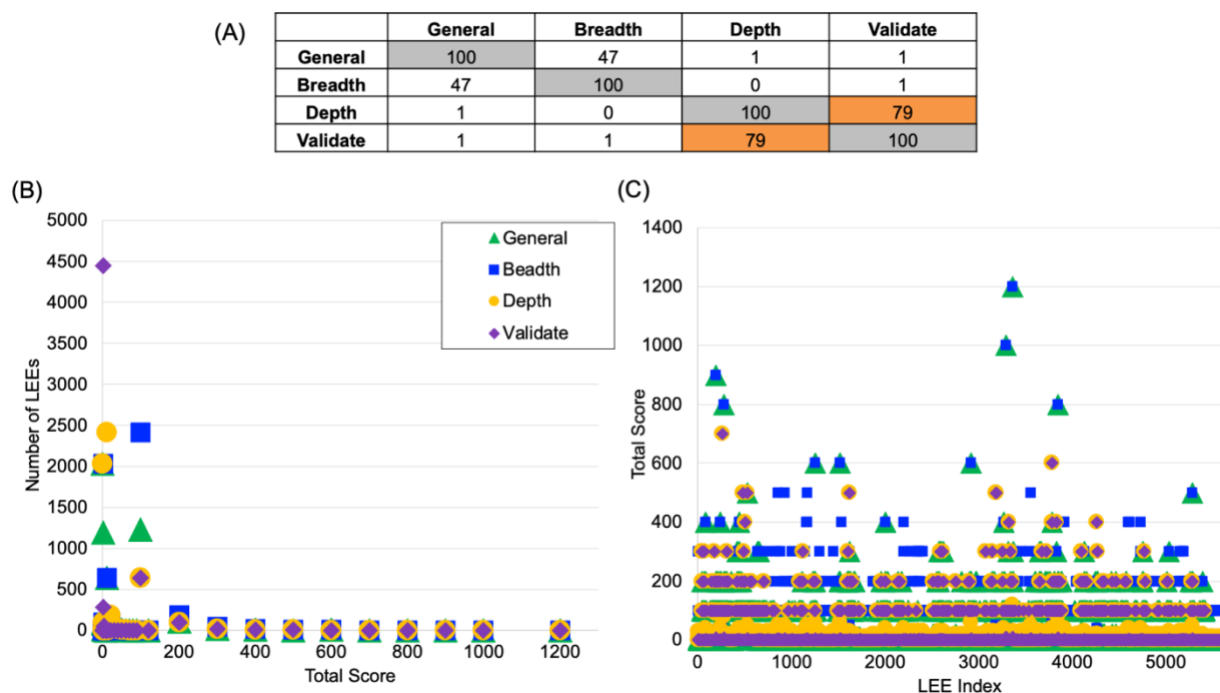


Figure 6-2. VIOLIN outcomes of our different modeling use cases. (A) shows the overlap between the top 100 scored LEEs, (greater than 50 overlap is in orange, greater than 90 overlap is in green). (B) shows the Total Score distribution for each approach, and (C) shows how the Total score for a specified LEE differs between the use cases.

Finally, before judging with VIOLIN, every LEE in the set was given an index number, and we used these indices to compare how the LEE was scored for each Match scheme. The results of this comparison are shown in Figure 6-2C. This further shows how the judgement outcomes are influenced by the user's choices.

6.2 Integration into Larger Framework

6.2.1 Integration with Filtering Methods

FLUTE (FiLter for Understanding True Events) [106], is a tool that can use public databases to judge the veracity of machine reading output. Figure 6-3 shows the pipeline of FLUTE, which takes a set of literature extracted events (or LEEs), uses information from public databases like STRING [55], STITCH [57], and Gene Ontology [7] to score these LEEs, and taking a thresholding value from the user, outputs those LEEs which meet the specified criteria. We took 12 of our LEE sets across three model systems (models A, B, and C) and filtered them with the FLUTE tool before judging the LEEs with VIOLIN. FLUTE kept an LEE in the set if it finds it in at least one of the supported databases.

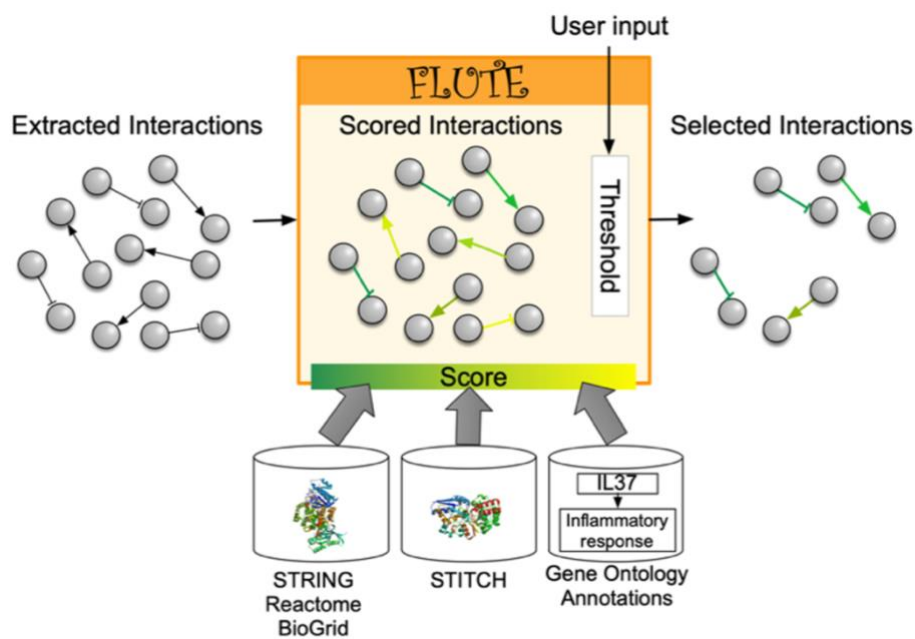


Figure 6-3. Schematic of the FLUTE pipeline from [19].

We also investigated how many LEEs remain in a set after applying a specified filtering criteria, represented as the number of remaining LEEs relating to the number in the original LEE set (retention percentage). LEE sets $R_{A2.0.1}$ and $R_{B2.0.1}$ were created from LEE sets R_{A2} and R_{B2} , respectively, by removing those LEEs representing PCIs or PBIs, and $R_{A2.0.1.1}$, was created by removing $R_{A2.0.1}$ of redundant or irrelevant LEEs, as outlined in Appendix B. We designate the associated FLUTE-filtered LEE sets as $R_{A2.1}$ and $R_{B2.1}$, respectively.

We initially found that using FLUTE to filter and LEE set removes many interactions, in some instances, more than half. Figure 6-4A shows that even for our $R_{A2.0.1.1}$ set, which was already manually judged for usefulness, was reduced by 61% when filtered. Figure 6-4B. shows how the filtering propagates through each classification subcategory, giving the average retention percentage. Overall, corroborations have the highest retention, and extensions have the lowest. This we would expect, since extensions are most likely to be novel or unverified interactions.

Within the subcategories, strong corroborations and specifications have the lowest retention, though we should note that these classifications rely on the comparison of additional attributes, of which only three of our LEE sets have available, as described in Section 4.4.

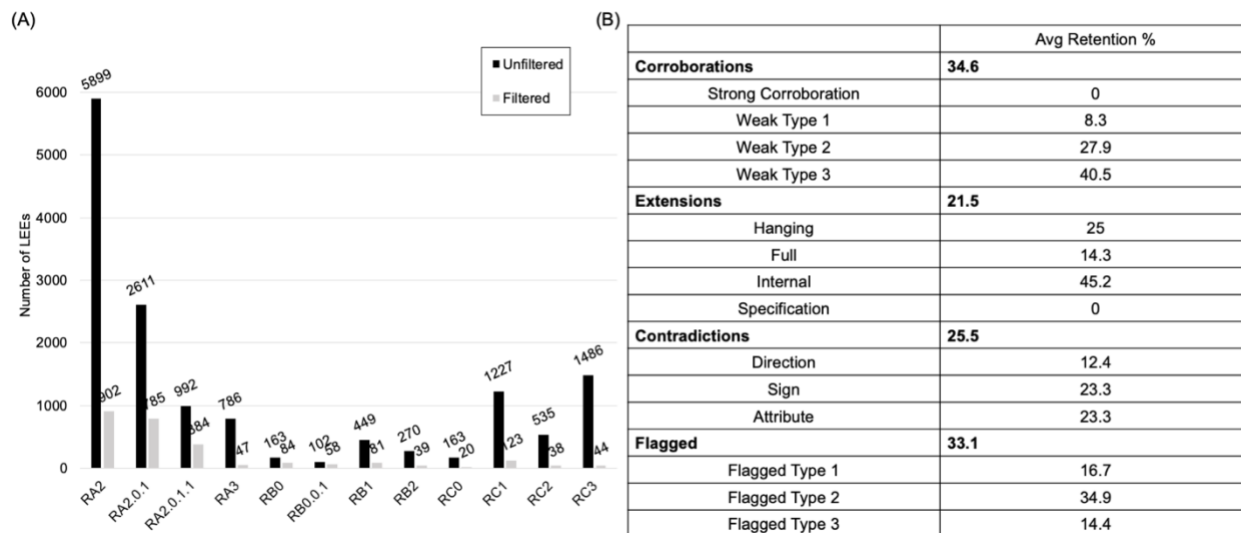


Figure 6-4. VIOLIN outcomes in conjunction with FLUTE filtering methods. (A) shows the number of LEEs for each set, with and without filtering. (B) shows the average retention rate for each classification category across all 12 LEE sets.

Figure 6-5 shows the VIOLIN output results for each LEE in both the unfiltered and filtered sets. Figure 6-5A and Figure 6-5B show how few strong corroborations and specifications were found even in the unfiltered sets. Similarly, the corroboration subcategory with the next lowest retention, weak type 1, also relies on additional attribute comparisons, as described in the Section 3.1.2.

The attribute contradiction subcategory was only calculated based on those three LEE sets which were comparing an additional attribute (location), and this small sample size may have artificially inflated the retention percentage. Figure 6-5C shows that direction and attribute

contradictions were found for only three of our LEE sets, and of the sign contradictions, no LEE judged as such from our BDNF-related LEE sets was retained.

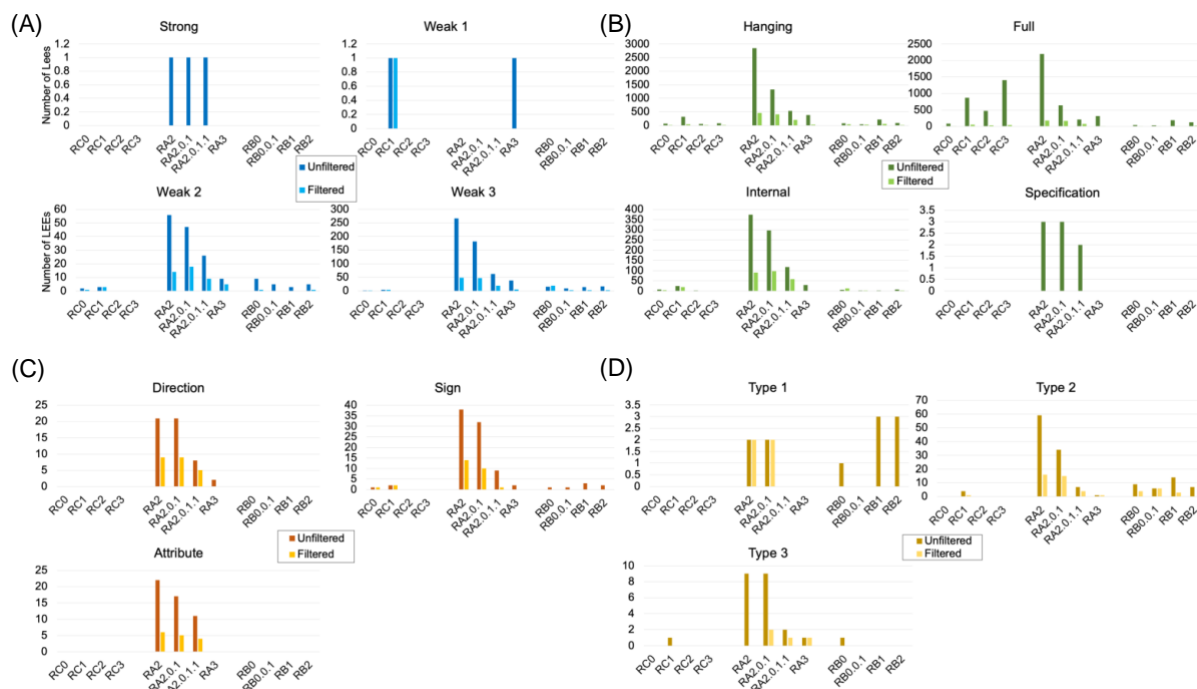


Figure 6-5. VIOLIN output results for each LEE set, broken down by classification category. (A) shows the subclassification categories of corroborations, (B) of extensions, (C) of contradictions, and (D) of flagged.

The corroboration category with the highest retention, weak type 3, occurs when the source and target nodes of an LEE matches the source node and end target node of a path in the model, as defined in Section 2.1.2. Figure 6-5D shows that of the flagged subcategories, type 2 is the only subcategory which retains at least one LEE for every LEE set filtered. Flagged type 2 describes an LEE whose source and target nodes match the source and end target of a path in the model, but there are one or more mismatched attributes (compared attribute, regulation sign, or regulation direction). We initially chose this distinction because this case could identify a feed forward of feedback loop, it could be caused by a machine reading error, or it may be a contradiction, some

new information identifying a more powerful direct interaction between two elements. The judgement requires expert knowledge to make, and so is flagged for further review.

Figure 6-6 shows the retention percentage of different filtering methods for LEE sets RA₂ and RB₂. In Figure 6-6A, we show that removing LEEs which involve chemicals or biological processes is the least stringent. The contradictions show the highest retention percentage in this method due to the components of modeled systems. Only 10% of model A nodes, and 5% of model B nodes are either chemicals or biological processes (model B in fact contains no nodes that represent biological processes), and so we would expect very few if not zero contradictions involving chemicals or biological processes.

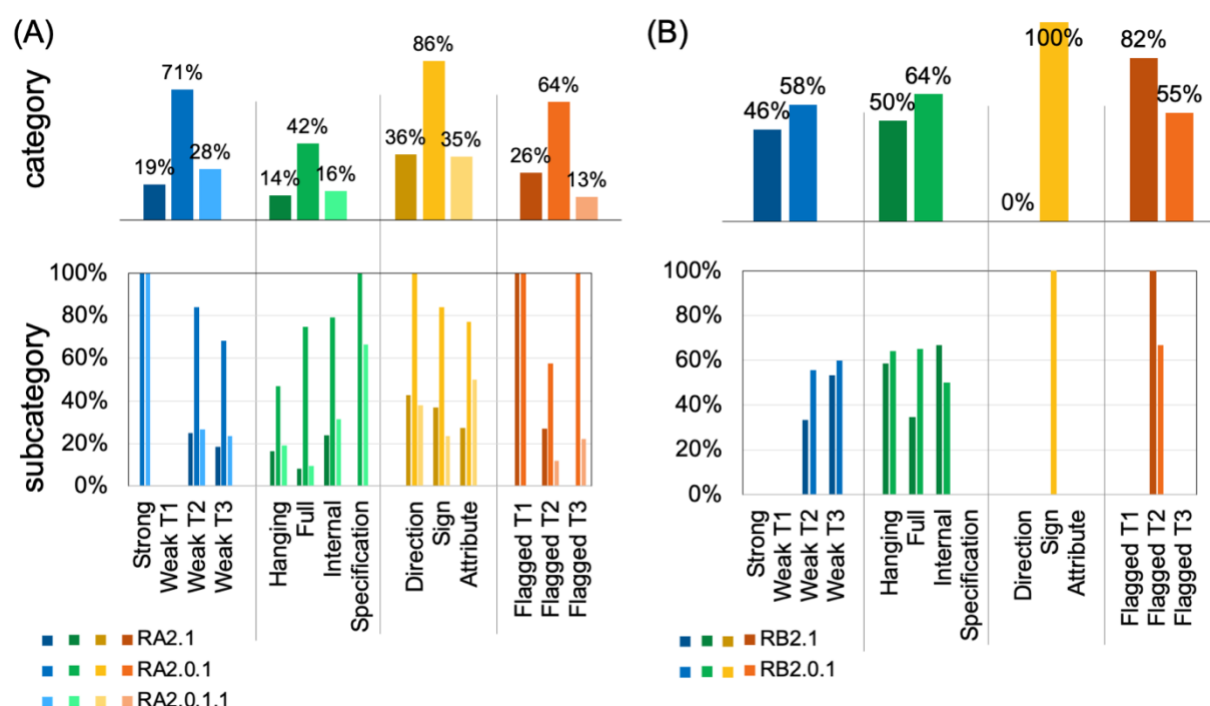


Figure 6-6. Showing the retention percentages across the subclassification categories. (A) Shows retention in RA2.1, RA2.0.1, and RA2.0.1.1 from RA2 and (B) shows retention in RB2.1 and RB2.0.1 from RB2.

Figure 6-6 also highlights the differences between automatic filtering with FLUTE (RA_{2.1}) and manual curation (RA_{2.0.1.1}). The similarities between these retention percentages support the

expertise of our manual curator, and the differences give us some insight into the choices of a VIOLIN user. FLUTE filtered out a higher percentage of weak corroboration type 3 LEEs, those indirect LEEs which correspond to a model path (Figure 6-6B). This could be attributed to databases not storing much information about paths between elements. FLUTE retained a higher percentage of flagged LEEs, specifically of flagged types 1 and 2 (Figure 6-6B). Flagged type 1 identifies potential feedback loops, which the expert did not find useful for the model. Flagged type 2 identifies potential feed-forward loops, of which the expert only judged some useful for the model. This distinction highlights the usefulness of VIOLIN's identification of feedback and feed-forward loops, as they may be biologically correct, but not desired for the model specifications. FLUTE had lower retention percentages than expert curation for specification and attribute contradiction subcategories (Figure 6-6B), which may be explained by a lack of attribute information in databases (in the case of our study, the compared attribute was interaction location).

The expert curator who selected the $R_{A2.0.1.1}$ set from R_{A2} chose to accept novel interactions as they could add previously unknown connections to the baseline model. On the other hand, FLUTE relies on the knowledge in databases, and therefore, it is more likely that novel LEEs are filtered out due to not yet being verified or added to databases.

6.2.2 Integration with a Model Extension Tool

CLARINET (CLARIfying NETworks) [111] is a tool for automatically extending models using information from the published literature. CLARINET takes a baseline model and a set of LEEs, and from the LEEs looks at not only how the LEEs relate to the model, but how they relate to each other, and from this CLARINET generates candidate extension clusters, which are then

ranked, and from that the user can find the best clusters to use to extend the baseline model (Figure 6-7).

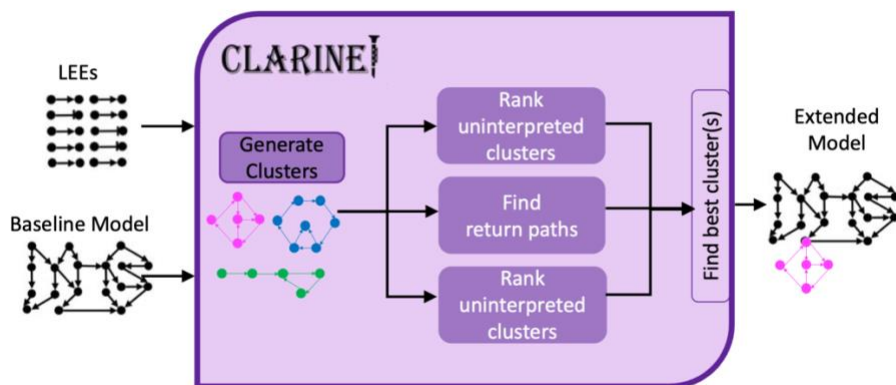


Figure 6-7. Schematic of the CLARINET framework

To investigate how VIOLIN output can be used with this tool, we introduce our final benchmarking input: a model of glioblastoma multiforme (GBM), which we call model G, and two associated LEE sets, R_{G1} and R_{G2} . These LEE sets were created by querying lists of proteins related to the GBM system through REACH Explorer, and then assembling machine reading output from the resultant papers using INDRA. The full details of the model are found in 7.0, and the details of the LEE sets are found in Appendix B. For each LEE set, we judged them first with VIOLIN, and then used the VIOLIN output as input for the CLARINET tool. In total, we created four sets of input from each LEE set: the raw input from the unjudged LEE set, the unique LEEs from the set, making use of VIOLIN’s Evidence Score calculation and duplicate LEE identification, the extensions from the judged VIOLIN output, and the “connected” extensions, i.e., the hanging, internal, and specification extensions from the VIOLIN output. Table 6-2 shows

the number of LEEs in each set of input. Of our set R_{G1} only about 60% of the LEEs are unique, and similarly from R_{G2} , only about 65% of LEEs are unique.

Table 6-2. Number of LEEs for Each CLARINET Input

LEE Set	Raw	Unique	Extensions	Connected Extensions
R_{G1}	10131	6214	6064	1779
R_{G2}	25857	17034	16718	4905

Figure 6-8 shows the results of VIOLIN judgement on these two reading sets. From Figure 6-8A we find that a vast majority of LEEs are judged as extensions. In Figure 6-8B, we see that the majority of these extensions are full. However, they may still connect to the model system, if one or both the nodes from a full extension LEE are found in a hanging LEE elsewhere in the set, or even if there is a path between several LEEs from a full extension LEE to the model system.

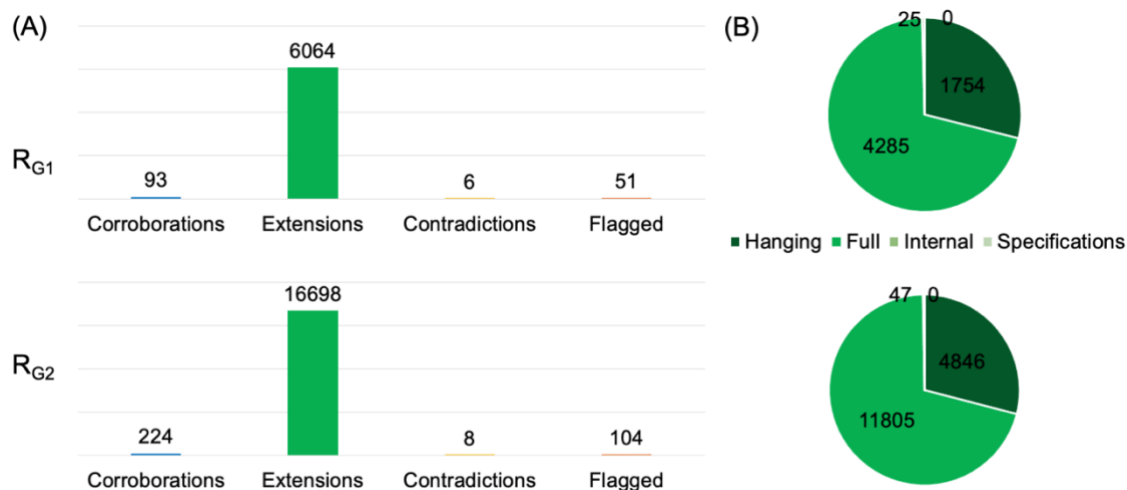


Figure 6-8. VIOLIN judgement of R_{G1} and R_{G2} . (A) shows the classification distribution, and (B) shows the subclassification distribution of the extensions.

And this is what drives our investigation with CLARINET. Figure 6-9 shows the resultant clusters generated by each of our input sets. As expected, the cluster sizes decrease in size as the input sets get smaller. In addition to the sizes, the cluster centers changed as the composition of the input set changed. For both R_{G1} and R_{G2} , one of the clusters from the raw input was not retained in the unique LEE input, and in the case of R_{G2} , a new cluster is introduced. This suggests that the frequency of an LEE has an influence on its use to the system. We investigated this with the Evidence Score we defined in Section 3.2.1, that the usefulness or relevance of an LEE is related to its frequency. Though we must note that in some cases, a high Evidence Score can be an indication of how ubiquitous an interaction is, or the focus of the source text (i.e., a paper on a focused topic will likely repeat a biological interaction).

For LEE set R_{G1} , clusters centered around Akt are identified in the raw, unique LEE, and extensions input, as shown in Figure 6-9, but the clusters generated from the connected extensions center introduce two completely new centers, PI3K and PTEN proteins. For set R_{G2} , A cluster centered around ERK is not identified in the raw set, but one is identified for the unique, extensions, and connected extensions inputs.

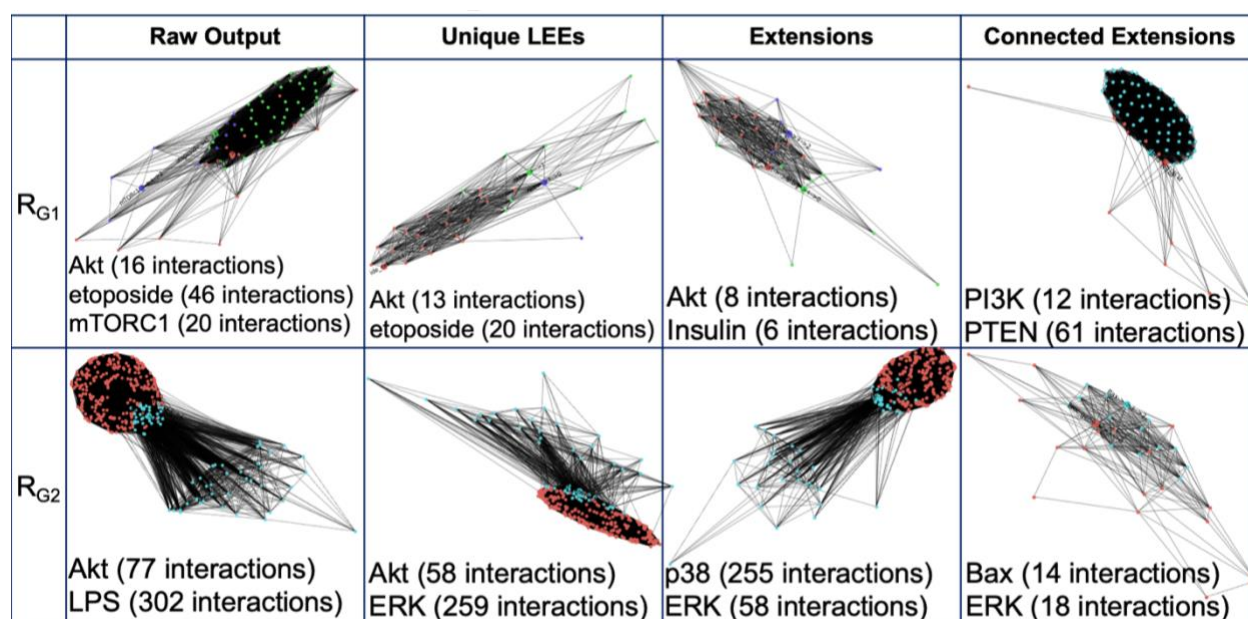


Figure 6-9. Cluster outputs from CLARINET for each of our input sets

These results tell us about how clusters can be generated with CLARINET, and how different modeling goals may determine which input set is most relevant. Figure 6-9 shows that etoposide and LPS, two chemicals used in cancer treatments and are *not* found in the baseline model G , are identified as significant addition to from the raw and unique output. The clusters from the extensions and connected extensions input are centered around nodes already found in the model. This suggests that using VIOLIN to first judge an LEE set can narrow the scope of extension – if one wanted a depth-first extension as described in Section 6.1, considering only connected extensions, or even taking a narrower focus of just internal extensions would allow for a tightly controlled extension. Using the full set of extensions as input would allow for more breadth in the extension of the baseline model.

The benefit of using the raw and unique outputs lies in the other VIOLIN classifications, namely the corroborations. By having corroborations in the candidate extension LEEs,

CLARINET can assemble clusters that connect in a more abstract way through interactions already found in the model.

7.0 Conclusions & Future Work

In this work, we introduced a tool for the fast, automated judgement of literature information as it relates to a baseline model, taking machine reading output and a baseline model and producing a classified output with numerical scores. In Chapter 1.0, we identified the current computational methods used in biological model building, and how they fit into the model-building process of question identification, information curation, model assembly, model extension, model testing and analysis, and eventual answering of the question. We highlighted how these tools improve and automate the modeling process, and also where they fall short of user need. We showed how the information available from the literature and from public databases is too vast for easy human curation. We also proposed how the VIOLIN tool can address these needs, and improve the speed and accuracy of comparing new information to an existing baseline.

In Chapter 2.0, we outlined the process by which VIOLIN carries out this task. We assembled benchmark baseline model and LEE input from various sources and systems to represent typical user input. Our multi-tier scoring method classifies information on a deeper level than other tools currently available. Our implementation of VIOLIN is supported by extensive online documentation, open source tutorials, and example input. In Chapter 4.0, we showed how the novel path-finding method of VIOLIN allows for the identification of potential feed-forward and feedback loops. We also evaluated the stability of our scoring metrics and our classification scheme choices, making sure they hold up against a variety of inputs. We developed VIOLIN in the context of the already available information databases and tools, making sure that VIOLIN is compatible within the larger network of model-building resources. We introduced methods of using VIOLIN for multiple modeling goals, and for using VIOLIN within a larger modeling

framework, integrating VIOLIN with a filtering tool that can be used to further increase the relevance and usefulness of LEEs considered, and showing how VIOLIN output can be utilized as input for automated model extension tools.

Our results showed the effectiveness of VIOLIN on a variety of systems. In Our developed tool was applied to models of melanoma, T cell differentiation, major depressive disorder, pancreatic cancer, and glioblastoma, created by varying methods and varying levels of detail. Our results also showed how machine reading output, and in particular machine reading errors, propagate through VIOLIN judgement. In Chapter 5.0, we showed further evaluation of VIOLIN's methods, parametrizing the Kind and Match scores, comparing multiple classification judgement schemes, and investigating the outcomes of VIOLIN's identification of corroborations and extensions, and in Chapter 6.0, we showed the effectiveness of using VIOLIN for multiple modeling goals, and it's potential to integrate with other modeling tools.

7.1 Future Work

As future work, our first priority is to further increase the compatibility of VIOLIN, adding the functionality to transform between the user-friendly spreadsheet formats presented here and the commonly used SBML [77], SBML-qual [78], and BEL [1] representation formats. Adding this functionality will allow VIOLIN to take input from virtually all the commonly used modeling tools available, expanding its applications further. Currently in development are methods in progress to transform between the BioRECIPES [13] format and the SBML and SBML-qual [77, 78] formats, but the challenges for this are differences between the two formats, such as the more mechanistic details usually presented in SBML. The ultimate goal would be to integrate into

VIOLIN the capability to take as input the major representation formats (BioRECIPES, BEL, SBML, SBML-qual, node-edge list), and compare them to machine reading output, user-assembled information, or even other models. To that end, we also aim extend the utility of VIOLIN towards the level 3 comparison described in Section 1.2, showing use cases of VIOLIN comparing a small baseline model to a larger, validated network or knowledge base.

We also plan to further investigate further capabilities of the Evidence Score. In this work, we implemented the Evidence Score based on the presence of recurring LEEs. However, we would like to investigate how we could create a more complex Evidence Score calculation taking into consideration such information as the evidence publication date or evidence source (journal versus database versus expert knowledge).

Expanding on the integration methods we presented in Chapter 6.0, we hope to further integrate VIOLIN as part of a fully automated model building and extension system. The VIOLIN tool can be applied to increasing understanding of diseases, identifying effectiveness of disease treatments, and even be applied on the larger modeling scale outside of biology.

Finally, to increase the utility of VIOLIN, we hope to suggest further scoring scheme input values for various use cases and implement a graphic user interface for VIOLIN use. This would make VIOLIN accessible to not only a wider range of modeling uses, but to a wider range of users, requiring less knowledge of programming syntax.

Appendix A Input Model Descriptions

Melanoma cell model (A). A discrete model of the melanoma SkMel-133 cell line was manually created based on the work from Korkut et al. [123], in which the authors performed DNA, RNA, and protein analysis on primary and metastatic melanomas from 331 patients. Their investigation focused on four specific genomic subtypes: mutant BRAF, mutant RAS, mutant NF1, and triple wild type. The goal of both the wet lab experiments and the model analysis was to investigate potential immunotherapies. Korkut et al. were able to identify additional subtypes that may benefit from MAPK and RTK inhibition therapies. The model consists of 255 nodes and 325 edges, with 54 input nodes (those which do not have regulators) and 91 output nodes (those which do not regulate other elements). This model includes PPIs, chemical regulation, the mRNA-gene-protein cascade, and regulations of and by biological processes such as apoptosis. Specific interaction mechanisms were not available for this model. We used the BioRECIPES [13] tabular format to represent the model.

Naïve T cell differentiation model (B). A model of the circuitry that controls the differentiation of naïve T cells into regulatory (Treg) and helper (Th) cells was published in Hawse et al. [124]. This model, with 61 nodes and 89 edges, 10 input elements, and 4 output elements, was manually created and confirmed by experimental results, investigating cellular response signals, such as extracellular stimulation. Model creation focused on the early steps of T cell activation, and the suppression or generation of regulatory T cells (Treg). The model connects the Akt-mTOR pathway to Treg development and shows the T cell receptor (TCR) pathways, including the activation and inhibition of FOXP3, leading to different T cell fates. In contrast to the melanoma cell model, this model includes almost exclusively protein-protein interactions, with

very few (three) chemical nodes. Most of the elements are assumed to have two main levels of activity, and are therefore represented with Boolean variables, and their update rules are logic functions. This model was also represented in the BioRECIPES tabular format.

Major depressive disorder model (C). A model of the brain-derived neurotrophic factor (BDNF) pathway as it relates to Major Depressive Disorder (MDD), published on NDEx [69] by Sandhya et al. [125]. BDNF has been linked to energy metabolism, memory, Alzheimers, and Huntington's Disease. This model consists of 72 nodes, 82 edges, 23 input elements, and 31 output elements, and created from expert literature. This model represents both the BDNF/TrkB and BDNF/p75NTR signaling pathways, creating a network of protein-protein interactions. The goal of this model was to assemble a network pathway to better direct scientific investigations of this system. It was published as a node-edge list and transformed by VIOLIN for judgement.

Pancreatic cancer cell model (D). A discrete model of the signaling pathways, metabolism, and microenvironment of pancreatic cancer was created by Telmer et al. [122], assembled from expert literature. This model was simulated over time and initialized with processes such as apoptosis, autophagy, immune response, etc. to investigate possible treatment therapies. This model, with 236 nodes, 316 edges, 99 input nodes, and 73 output nodes, was assembled from 19 core papers, which in turn cited 2,158 paper, was assembled manually from the information in these references. Model development was centered around the Hallmarks of Cancer [127], as was also a foundation for model building tools and methodologies. Like the melanoma model, this model includes PPIs and PBIs, as well as cell cycle progression. It was represented in the BioRECIPES format.

Glioblastoma multiforme model (G). A model network of Glioblastoma multiforme, assembled by Holtzaple et al [23]. This model was created by a combination of manual assembly,

automated assembly, automated extension, and automated verification methods, using tools such as FLUTE [106], INDRA [2], PCnet [61], as well as relevant papers and databases. This model has 237 nodes, 366 edges, 59 input nodes, and 90 output nodes. The goal of this model was to showcase a workflow for using open-source resources for network building and verification to understand the signaling pathways of GBM.

Appendix B Input LEE Sets

Appendix Table 1. LEE set parameters

LEE Set	Model System	Curation Method	Query Terms	Number of LEEs
R _{A1}	Skel-133 melanoma	REACH	General system query	131445
R _{A2}		REACH Explorer & Fetch tools	MEK, ERK, AKT, GSK3, P70RSK, S6, CDK4, 4EBP1, YB1, SRC, CHK2, MTOR, PI3K	6305
R _{A2.1}		FLUTE	R _{A2} filtered with FLUTE	902
R _{A2.0.1}		Manual	R _{A2} removed of PCIs and PBIs	2876
R _{A2.1.1}		FLUTE	R _{A2.0.1} filtered with FLUTE	785
R _{A2.0.1.1}		Manual	R _{A2.0.1} removed of redundant or irrelevant LEEs	1102
R _{A2.1.1.1}		FLUTE	R _{A2.0.1.1} filtered with FLUTE	384
R _{A3}		INDRA	MAPK/ERK pathway	1106
R _{A4}		INDRA	RPS6K1	21
R _{B0}	T-cell differentiation	INDRA	Machine Reading output from Hawse et al. [34] references	188
R _{B0.1}		FLUTE	R _{B0} filtered with FLUTE	84
R _{B0.0.1}		Manual	R _{B2} removed of PBIs and contradictions	117
R _{B0.1.1}		FLUTE	R _{B0.0.1} filtered with FLUTE	58
R _{B1}		INDRA	PTEN	711
R _{B1.1}		FLUTE	R _{B1} filtered with FLUTE	81
R _{B2}		INDRA	T-cell, PTEN, AKT, FOXO	293

Appendix Table 1. (continued)

R _{B2.1}	T-cell differentiaion	FLUTE	R _{B2} filtered with FLUTE	39
R _{B*1}		INDRA	Breast Cancer, DNA repair, Autophagy, Cancer	1336
R _{B*2}		INDRA	DNA repair, BRCA1, ADAM17, inflammation	865
R _{C0}	BDNF pathway	INDRA	References from Sandhya et al. [36]	380
R _{C0.1}		FLUTE	R _{C0} filtered with FLUTE	20
R _{C1}		INDRA	BDNF, NTRK, MDD, NFkB, TRK	1830
R _{C1.1}		FLUTE	R _{C1} filtered with FLUTE	123
R _{C2}		INDRA	MEF2A, MEF2C	789
R _{C2.1}		FLUTE	R _{C2} filtered with FLUTE	38
R _{C3}		INDRA	JUN, c-JUN	2521
R _{C3.1}		FLUTE	R _{C3} filtered with FLUTE	44
R _{D0}	Pancreatic cancer cell	INDRA	References from Telmer et al. [33]	706
R _{G1}	Glioblastoma multiforme	INDRA	TERT, TP53, ATRX, EGFR, PTEN, IDH1, PDGFRA, NF1, NEFL, GABRA1, SYT1, SLC12A5, RB, PI3K/AKT, MGMT, citrate, daxx	10131
R _{G2}		INDRA	Sos, VEGFA, CDK1, MCHR, MCHR2, ADCY1, MEK, CAM, ERK, ECM, ARF, MAPK, Raf, Ras, TGF, TGFa, IGF-1, p48, p21, GADD45, POLK, PLC, Shc	25857

Appendix C Example Typical Baseline Model

Here we illustrate how one of our baseline models (model C) was taken from NDEx, saved into a spreadsheet file, and then transformed upon input within VIOLIN. Many models on the NDEx database can be viewed as a node-edge list spreadsheet, as shown in Appendix Figure 1.

Source Node	Interaction	Target Node	directed	pmid	ndex:citation
ADAM17	controls-state-change-of	NGFR	true	20421303	pmid:20421303
AKT1	controls-phosphorylati...	BAD	true	10558990	pmid:10558990
AKT1	controls-phosphorylati...	YBX1	true	22699894	pmid:22699894
BDNF	interacts-with	CPE		15664176	pmid:15664176
BDNF	interacts-with	NGFR		8	8
BDNF	interacts-with	NGF		7961712	pmid:7961712
BDNF	interacts-with	NTF3		7961712	pmid:7961712
BDNF	interacts-with	NTRK2		3	3
BDNF	interacts-with	SORT1		15987945	pmid:15987945
CDH2	in-complex-with	CTNNB1			
CDH2	interacts-with	NTRK2		18664491	pmid:18664491
CDK5	controls-phosphorylati...	NTRK2	true	17341134	pmid:17341134
CDK5	interacts-with	NTRK2		17341134	pmid:17341134
CDKL5	interacts-with	RAC1		20861382	pmid:20861382
CDKR1	interacts-with	NTRK2		17341134	pmid:17341134
CREB1	interacts-with	CRTC1		20639200	pmid:20639200

Total Items: 82

Appendix Figure 1. Model C as it is presented on the NDEx database

While NDEx does not currently support the ability to download this spreadsheet directly, it's very easy to copy into a CSV or Excel file, as shown in Appendix Table 2. The node types and standardized IDs are found in a separate “node” spreadsheet on the NDEx database.

Appendix Table 2. The model information copied from NDEx to a spreadsheet that can be saved as an input file for VIOLIN

Source Node	Source Type	Source ID	Target Node	Target Type	Target ID
ADAM17	Protein	P78536	NGFR	Protein	P08138
AKT1	Protein	P31749	BAD	Protein	Q92934
AKT1	Protein	P31749	YBX1	Protein	P67809
BDNF	Protein	P23560	CPE	Protein	P16870

Finally, this node-edge list is easily transformed in VIOLIN to the BioRECIPES format, shown in Appendix Table 3. In this case, we used the evidence text for each model edge to manually determine and add the connection type attributes to this model. And since this model did not give specific variable names to each node, the common element name was used as the variable names, which are used in the VIOLIN path-finding function.

Appendix Table 3. Model C transformed into the BioRECIPES format

Element Name	Element Type	Element IDs	Variable	Positive Regulators	Positive Connection Type	Negative Regulators	Negative Connection Type
ADAM17	Protein	P78536	ADAM17				
AKT1	Protein	P31749	AKT1				
BAD	Protein	Q92934	BAD	AKT1,RPS6KA3	D,D		
BDNF	Protein	P23560	BDNF	NTRK2	D		

Bibliography

- [1] C. T. Hoyt *et al.*, "Re-curation and rational enrichment of knowledge graphs in Biological Expression Language," *Database: The Journal of Biological Databases and Curation*, vol. 2019, 2019/06/21/ 2019, doi: 10.1093/database/baz068.
- [2] B. M. Gyori, J. A. Bachman, K. Subramanian, J. L. Muhlich, L. Galescu, and P. K. Sorger, "From word models to executable models of signaling networks using automated assembly," (in en), *Molecular Systems Biology*, vol. 13, no. 11, p. 954, 2017/11// 2017, doi: 10.15252/msb.20177651.
- [3] M. Ostaszewski *et al.*, "COVID-19 Disease Map, building a computational repository of SARS-CoV-2 virus-host interaction mechanisms," (in en), *Scientific Data*, vol. 7, no. 1, p. 136, 2020/12// 2020, doi: 10.1038/s41597-020-0477-8.
- [4] V. Singh *et al.*, "RA-map: building a state-of-the-art interactive knowledge base for rheumatoid arthritis," (in en), *Database*, vol. 2020, p. baaa017, 2020/01/01/ 2020, doi: 10.1093/database/baaa017.
- [5] T. U. Consortium, "UniProt: a worldwide hub of protein knowledge," *Nucleic Acids Research*, vol. 47, no. D1, pp. D506-D515, 2018, doi: 10.1093/nar/gky1049.
- [6] PubChem. PubChem Database [Online] Available: <https://pubchem.ncbi.nlm.nih.gov/>
- [7] M. Ashburner *et al.*, "Gene Ontology: tool for the unification of biology," (in en), *Nature Genetics*, vol. 25, no. 1, pp. 25-29, 2000/05// 2000, doi: 10.1038/75556.
- [8] "PubMed Central," *Reference Reviews*, vol. 19, no. 3, pp. 37-38, 2005 2005, doi: doi:10.1108/09504120510587797.
- [9] W. C. Baldwin, B. Sauser, and R. Cloutier, "Simulation Approaches for System of Systems: Events-based versus Agent Based Modeling," (in en), *Procedia Computer Science*, vol. 44, pp. 363-372, 2015 2015, doi: 10.1016/j.procs.2015.03.032.
- [10] G. Cugola, E. Di Nitto, and A. Fuggetta, "Exploiting an event-based infrastructure to develop complex distributed systems," in *20th International Conference on Software Engineering*, 1998 1998, Kyoto, Japan: IEEE Comput. Soc, pp. 261-270, doi: 10.1109/ICSE.1998.671135. [Online]. Available: <http://ieeexplore.ieee.org/document/671135/>
- [11] S. Magnenat, P. Rétornaz, M. Bonani, V. Longchamp, and F. Mondada, "ASEBA: A Modular Architecture for Event-Based Control of Complex Robots," *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 2, pp. 321-329, 2011/04// 2011, doi: 10.1109/TMECH.2010.2042722.

- [12] W. S. Hlavacek, J. R. Faeder, M. L. Blinov, R. G. Posner, M. Hucka, and W. Fontana, "Rules for Modeling Signal-Transduction Systems," (in en), *Science Signaling*, vol. 2006, no. 344, pp. re6-re6, 2006/07/18/ 2006, doi: 10.1126/stke.3442006re6.
- [13] K. Sayed, C. A. Telmer, A. A. Butchy, and N. Miskov-Zivanov, "Recipes for Translating Big Data Machine Reading to Executable Cellular Signaling Models," in *Machine Learning, Optimization, and Big Data*, vol. 10710, G. Nicosia, P. Pardalos, G. Giuffrida, and R. Umeton Eds. Cham: Springer International Publishing, 2018, pp. 1-15.
- [14] R.-S. Wang, "Ordinary Differential Equation (ODE), Model," in *Encyclopedia of Systems Biology*, W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota Eds. New York, NY: Springer New York, 2013, pp. 1606-1608.
- [15] R. Machne, A. Finney, S. Muller, J. Lu, S. Widder, and C. Flamm, "The SBML ODE Solver Library: a native API for symbolic and fast numerical analysis of reaction networks," (in en), *Bioinformatics*, vol. 22, no. 11, pp. 1406-1407, 2006/06/01/ 2006, doi: 10.1093/bioinformatics/btl086.
- [16] S. Hoops *et al.*, "Ordinary Differential Equations (ODEs) Based Modeling," in *Computational Immunology*: Elsevier, 2016, pp. 63-78.
- [17] T. Chen, H. L. He, and G. M. Church, "Modeling gene expression with differential equations," (in eng), *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pp. 29-40, 1999 1999.
- [18] S. Ando, E. Sakamoto, and H. Iba, "Evolutionary modeling and inference of gene network," (in en), *Information Sciences*, vol. 145, no. 3-4, pp. 237-259, 2002/09// 2002, doi: 10.1016/S0020-0255(02)00235-9.
- [19] H. Cao, L. Kang, Y. Chen, and J. Yu, "Evolutionary Modeling of Systems of Ordinary Differential Equations with Genetic Programming," (in en), *Genetic Programming and Evolvable Machines*, vol. 1, no. 4, pp. 309-337, 2000/10/01/ 2000, doi: 10.1023/A:1010013106294.
- [20] L. A. Chylek, L. A. Harris, C.-S. Tung, J. R. Faeder, C. F. Lopez, and W. S. Hlavacek, "Rule-based modeling: a computational approach for studying biomolecular site dynamics in cell signaling systems: Computational approach for studying biomolecular site dynamics in cell signaling systems," (in en), *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, vol. 6, no. 1, pp. 13-36, 2014/01// 2014, doi: 10.1002/wsbm.1245.
- [21] L. A. Chylek, L. A. Harris, J. R. Faeder, and W. S. Hlavacek, "Modeling for (physical) biologists: an introduction to the rule-based approach," *Physical Biology*, vol. 12, no. 4, p. 045007, 2015/07/16/ 2015, doi: 10.1088/1478-3975/12/4/045007.
- [22] L. A. Harris *et al.*, "BioNetGen 2.2: advances in rule-based modeling," (in en), *Bioinformatics*, vol. 32, no. 21, pp. 3366-3368, 2016/11/01/ 2016, doi: 10.1093/bioinformatics/btw469.

- [23] M. A. Marchisio, M. Colaiacovo, E. Whitehead, and J. Stelling, "Modular, rule-based modeling for the design of eukaryotic synthetic gene circuits," *BMC Systems Biology*, vol. 7, no. 1, p. 42, 2013/05/27/ 2013, doi: 10.1186/1752-0509-7-42.
- [24] D. T. Gillespie, "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions," (in en), *Journal of Computational Physics*, vol. 22, no. 4, pp. 403-434, 1976/12// 1976, doi: 10.1016/0021-9991(76)90041-3.
- [25] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *The Journal of Physical Chemistry*, vol. 81, no. 25, pp. 2340-2361, 1977/12/01/ 1977, doi: 10.1021/j100540a008.
- [26] M. L. Blinov, J. R. Faeder, B. Goldstein, and W. S. Hlavacek, "BioNetGen: Software for rule-based modeling of signal transduction based on the interactions of molecular domains," *Bioinformatics*, vol. 20, pp. 3289-3291, 2004, doi: 10.1093/bioinformatics/bth378.
- [27] Y. Cao, D. T. Gillespie, and L. R. Petzold, "The slow-scale stochastic simulation algorithm," (in en), *The Journal of Chemical Physics*, vol. 122, no. 1, p. 014116, 2005/01// 2005, doi: 10.1063/1.1824902.
- [28] M. W. Sneddon, J. R. Faeder, and T. Emonet, "Efficient modeling, simulation and coarse-graining of biological complexity with NFsim," (in en), *Nature Methods*, vol. 8, no. 2, pp. 177-183, 2011/02// 2011, doi: 10.1038/nmeth.1546.
- [29] J. D. Schwab, S. D. Kühlwein, N. Ikonomi, M. Köhl, and H. A. Kestler, "Concepts in Boolean network modeling: What do they all mean?," *Computational and Structural Biotechnology Journal*, vol. 18, pp. 571-582, 2020 2020, doi: <https://doi.org/10.1016/j.csbj.2020.03.001>.
- [30] J. Saez-Rodriguez, L. G. Alexopoulos, M. Zhang, M. K. Morris, D. A. Lauffenburger, and P. K. Sorger, "Comparing signaling networks between normal and transformed hepatocytes using discrete logical models," (in eng), *Cancer Research*, vol. 71, no. 16, pp. 5400-5411, 2011/08/15/ 2011, doi: 10.1158/0008-5472.CAN-10-4453.
- [31] I. Albert, J. Thakar, S. Li, R. Zhang, and R. Albert, "Boolean network simulations for life scientists," (in eng), *Source Code for Biology and Medicine*, vol. 3, p. 16, 2008/11/14/ 2008, doi: 10.1186/1751-0473-3-16.
- [32] C. A. Telmer *et al.*, "Computational modeling of cell signaling and mutations in pancreatic cancer," *Systems Biology*, preprint 2021/06/09/ 2021. Accessed: 2021/07/06/17:26:21. [Online]. Available: <http://biorxiv.org/lookup/doi/10.1101/2021.06.08.447557>
- [33] E. Holtzapple, B. Cochran, and N. Miskov-Zivanov, "Automated verification, assembly, and extension of GBM stem cell network model with knowledge from literature and data," *Systems Biology*, preprint 2021/07/05/ 2021. Accessed: 2021/08/16/15:21:44. [Online]. Available: <http://biorxiv.org/lookup/doi/10.1101/2021.07.04.451062>

- [34] K. Murphy, "Dynamic bayesian networks: representation, inference and learning," University of California, Berkley, 2002.
- [35] V. Mihajlovic and M. Petkovic, "Dynamic Bayesian Networks: A State of the Art," Technical Report Series, 2001.
- [36] N. Miskov-Zivanov, D. Marculescu, and J. R. Faeder, "Dynamic behavior of cell signaling networks: model design and analysis automation," in *the 50th Annual Design Automation Conference*, 2013 2013, Austin, Texas: ACM Press, p. 1, doi: 10.1145/2463209.2488743. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2463209.2488743>
- [37] M. Afzal *et al.*, "Knowledge-Based Query Construction Using the CDSS Knowledge Base for Efficient Evidence Retrieval," (in en), *Sensors*, vol. 15, no. 9, pp. 21294-21314, 2015/08/28/ 2015, doi: 10.3390/s150921294.
- [38] K. Sayed, Y.-H. Kuo, A. Kulkarni, and N. Miskov-Zivanov, "DiSH simulator: Capturing dynamics of cellular signaling with heterogeneous knowledge," in *2017 Winter Simulation Conference (WSC)*, 2017/12// 2017, Las Vegas, NV: IEEE, pp. 896-907, doi: 10.1109/WSC.2017.8247841. [Online]. Available: <http://ieeexplore.ieee.org/document/8247841/>
- [39] C. A. Telmer *et al.*, "Dynamic system explanation: DySE, a framework that evolves to reason about complex systems - lessons learned," in *AIDR '19: Artificial Intelligence for Data Discovery and Reuse 2019*, 2019/05/13/ 2019, Pittsburgh Pennsylvania: ACM, pp. 1-10, doi: 10.1145/3359115.3359123. [Online]. Available: <https://dl.acm.org/doi/10.1145/3359115.3359123>
- [40] S. Kauffman, C. Peterson, B. Samuelsson, and C. Troein, "Random Boolean network models and the yeast transcriptional network," (in en), *Proceedings of the National Academy of Sciences*, vol. 100, no. 25, pp. 14796-14799, 2003/12/09/ 2003, doi: 10.1073/pnas.2036429100.
- [41] I. Shmulevich, E. R. Dougherty, and Z. Wei, "From Boolean to probabilistic Boolean networks as models of genetic regulatory networks," (in en), *Proceedings of the IEEE*, vol. 90, no. 11, pp. 1778-1792, 2002/11// 2002, doi: 10.1109/JPROC.2002.804686.
- [42] I. Shmulevich, E. R. Dougherty, S. Kim, and W. Zhang, "Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks," (in en), *Bioinformatics*, vol. 18, no. 2, pp. 261-274, 2002/02/01/ 2002, doi: 10.1093/bioinformatics/18.2.261.
- [43] C. Xiao, Y. Jin, J. Liu, B. Zeng, and S. Huang, "Optimal Expert Knowledge Elicitation for Bayesian Network Structure Identification," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1163-1177, 2018/07// 2018, doi: 10.1109/TASE.2017.2747130.
- [44] L. Bottou *et al.*, "Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising," *Journal of Machine Learning Research*, vol. 14, no. 65, pp. 3207-3260, 2013 2013.

- [45] T. N. Beran and C. Violato, "Structural equation modeling in medical research: a primer," (in en), *BMC Research Notes*, vol. 3, no. 1, p. 267, 2010/12// 2010, doi: 10.1186/1756-0500-3-267.
- [46] S. Wright, "The Relative Importance of Heredity and Environment in Determining the Piebald Pattern of Guinea-Pigs," (in en), *Proceedings of the National Academy of Sciences*, vol. 6, no. 6, pp. 320-332, 1920/06// 1920, doi: 10.1073/pnas.6.6.320.
- [47] R. Oughtred *et al.*, "The BioGRID interaction database: 2019 update," (in en), *Nucleic Acids Research*, vol. 47, no. D1, pp. D529-D541, 2019/01/08/ 2019, doi: 10.1093/nar/gky1079.
- [48] L. Licata *et al.*, "SIGNOR 2.0, the SIGnaling Network Open Resource 2.0: 2019 update," *Nucleic Acids Research*, vol. 48, no. D1, pp. D504-D510, 2020 2020, doi: 10.1093/nar/gkz949.
- [49] E. G. Cerami *et al.*, "Pathway Commons, a web resource for biological pathway data," (in en), *Nucleic Acids Research*, vol. 39, no. Database, pp. D685-D690, 2011/01/01/ 2011, doi: 10.1093/nar/gkq1039.
- [50] A. Fabregat *et al.*, "The Reactome pathway Knowledgebase," (in en), *Nucleic Acids Research*, vol. 44, no. D1, pp. D481-D487, 2016/01/04/ 2016, doi: 10.1093/nar/gkv1351.
- [51] H. Hermjakob *et al.*, "IntAct: an open source molecular interaction database," *Nucleic Acids Research*, vol. 32, no. Database issue, pp. D452-D455, 2004/01/01/ 2004, doi: 10.1093/nar/gkh052.
- [52] S. Peri *et al.*, "Development of Human Protein Reference Database as an Initial Platform for Approaching Systems Biology in Humans," (in en), *Genome Research*, vol. 13, no. 10, pp. 2363-2371, 2003/10// 2003, doi: 10.1101/gr.1680803.
- [53] G. R. Mishra, "Human protein reference database--2006 update," (in en), *Nucleic Acids Research*, vol. 34, no. 90001, pp. D411-D414, 2006/01/01/ 2006, doi: 10.1093/nar/gkj141.
- [54] D. Szklarczyk *et al.*, "STRING v10: protein–protein interaction networks, integrated over the tree of life," (in en), *Nucleic Acids Research*, vol. 43, no. D1, pp. D447-D452, 2015/01/28/ 2015, doi: 10.1093/nar/gku1003.
- [55] D. Szklarczyk *et al.*, "The STRING database in 2021: customizable protein–protein networks, and functional characterization of user-uploaded gene/measurement sets," (in en), *Nucleic Acids Research*, vol. 49, no. D1, pp. D605-D612, 2021/01/08/ 2021, doi: 10.1093/nar/gkaa1074.
- [56] C. von Mering, "STRING: known and predicted protein-protein associations, integrated and transferred across organisms," (in en), *Nucleic Acids Research*, vol. 33, no. Database issue, pp. D433-D437, 2004/12/17/ 2004, doi: 10.1093/nar/gki005.

- [57] M. Kuhn *et al.*, "STITCH 4: integration of protein–chemical interactions with user data," (in en), *Nucleic Acids Research*, vol. 42, no. D1, pp. D401-D407, 2014/01// 2014, doi: 10.1093/nar/gkt1207.
- [58] A. J. Williams *et al.*, "Open PHACTS: semantic interoperability for drug discovery," (in eng), *Drug Discovery Today*, vol. 17, no. 21-22, pp. 1188-1198, 2012/11// 2012, doi: 10.1016/j.drudis.2012.05.016.
- [59] G. Joshi-Tope, "Reactome: a knowledgebase of biological pathways," (in en), *Nucleic Acids Research*, vol. 33, no. Database issue, pp. D428-D432, 2004/12/17/ 2004, doi: 10.1093/nar/gki072.
- [60] M. Kanehisa, Y. Sato, M. Kawashima, M. Furumichi, and M. Tanabe, "KEGG as a reference resource for gene and protein annotation," (in en), *Nucleic Acids Research*, vol. 44, no. D1, pp. D457-D462, 2016/01/04/ 2016, doi: 10.1093/nar/gkv1070.
- [61] J. K. Huang *et al.*, "Systematic Evaluation of Molecular Networks for Discovery of Disease Genes," *Cell Systems*, vol. 6, no. 4, pp. 484-495.e5, 2018 2018, doi: 10.1016/j.cels.2018.03.001.
- [62] M. Glont *et al.*, "BioModels: expanding horizons to include more modelling approaches and formats," (in en), *Nucleic Acids Research*, vol. 46, no. D1, pp. D1248-D1253, 2018/01/04/ 2018, doi: 10.1093/nar/gkx1023.
- [63] R. S. Malik-Sheriff *et al.*, "BioModels-15 years of sharing computational models in life science," (in eng), *Nucleic acids research*, vol. 48, no. D1, pp. D407-D415, 2020/01/08/ 2020, doi: 10.1093/nar/gkz1055.
- [64] N. Le Novère, "BioModels Database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems," (in en), *Nucleic Acids Research*, vol. 34, no. 90001, pp. D689-D691, 2006/01/01/ 2006, doi: 10.1093/nar/gkj092.
- [65] A. Chatr-aryamontri *et al.*, "MINT: the Molecular INTeraction database," (in en), *Nucleic Acids Research*, vol. 35, no. Database, pp. D572-D574, 2007/01/03/ 2007, doi: 10.1093/nar/gkl950.
- [66] Y. Pita-Juárez *et al.*, "The Pathway Coexpression Network: Revealing pathway relationships," (in eng), *PLoS computational biology*, vol. 14, no. 3, p. e1006042, 2018/03// 2018, doi: 10.1371/journal.pcbi.1006042.
- [67] D. Türei, T. Korcsmáros, and J. Saez-Rodriguez, "OmniPath: guidelines and gateway for literature-curated signaling pathway resources," (in eng), *Nature Methods*, vol. 13, no. 12, pp. 966-967, 2016/11/29/ 2016, doi: 10.1038/nmeth.4077.
- [68] F. Büchel *et al.*, "Path2Models: large-scale generation of computational models from biochemical pathway maps," (in eng), *BMC systems biology*, vol. 7, p. 116, 2013/11/01/ 2013, doi: 10.1186/1752-0509-7-116.

- [69] D. Pratt, J. Chen, D. Welker, J. Kuentzer, B. Demchak, and T. Ideker, "NDEx , the Network Data Exchange," *Cell Systems*, vol. 1, no. 4, pp. 302-305, 2015, doi: 10.1016/j.cels.2015.10.001.
- [70] A. R. Pico, T. Kelder, M. P. van Iersel, K. Hanspers, B. R. Conklin, and C. Evelo, "WikiPathways: Pathway Editing for the People," (in en), *PLoS Biology*, vol. 6, no. 7, p. e184, 2008/07/22/ 2008, doi: 10.1371/journal.pbio.0060184.
- [71] T. Helikar *et al.*, "The Cell Collective: Toward an open and collaborative approach to systems biology," *BMC Systems Biology*, vol. 6, no. 1, p. 96, 2012 2012, doi: 10.1186/1752-0509-6-96.
- [72] F. Belleau, M.-A. Nolin, N. Tourigny, P. Rigault, and J. Morissette, "Bio2RDF: Towards a mashup to build bioinformatics knowledge systems," (in en), *Journal of Biomedical Informatics*, vol. 41, no. 5, pp. 706-716, 2008/10// 2008, doi: 10.1016/j.jbi.2008.03.004.
- [73] A. Naldi *et al.*, "Cooperative development of logical modelling standards and tools with CoLoMoTo," (in eng), *Bioinformatics (Oxford, England)*, vol. 31, no. 7, pp. 1154-1159, 2015/04/01/ 2015, doi: 10.1093/bioinformatics/btv013.
- [74] A. Niarakis *et al.*, "Setting the basis of best practices and standards for curation and annotation of logical models in biology-highlights of the [BC]2 2019 CoLoMoTo/SysMod Workshop," (in eng), *Briefings in Bioinformatics*, vol. 22, no. 2, pp. 1848-1859, 2021/03/22/ 2021, doi: 10.1093/bib/bbaa046.
- [75] G. Ferguson and J. F. Allen, "TRIPs: an integrated intelligent problem-solving assistant," presented at the Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence, Madison, Wisconsin, USA, 1998.
- [76] M. A. Valenzuela-Escárcega *et al.*, "Large-scale automated machine reading discovers new cancer-driving mechanisms," (in eng), *Database : the journal of biological databases and curation*, vol. 2018, p. bay098, 2018, doi: 10.1093/database/bay098.
- [77] M. Hucka *et al.*, "The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models," *Bioinformatics*, vol. 19, no. 4, pp. 524-531, 2003/03/01/ 2003, doi: 10.1093/bioinformatics/btg015.
- [78] C. Chaouiya *et al.*, "SBML qualitative models: a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools," *BMC Systems Biology*, vol. 7, no. 1, p. 135, 2013/12/10/ 2013, doi: 10.1186/1752-0509-7-135.
- [79] T. Slater, "Recent advances in modeling languages for pathway maps and computable biological networks," (in en), *Drug Discovery Today*, vol. 19, no. 2, pp. 193-198, 2014/02/01/ 2014, doi: 10.1016/j.drudis.2013.12.011.

- [80] E. Demir *et al.*, "BioPAX – A community standard for pathway data sharing," *Nature biotechnology*, vol. 28, no. 9, pp. 935-942, 2010/09// 2010, doi: 10.1038/nbt.1666.
- [81] J.-L. Yu and S. R. J. Jang, "A mathematical model of tumor-immune interactions with an immune checkpoint inhibitor," (in en), *Applied Mathematics and Computation*, vol. 362, p. 124523, 2019/12// 2019, doi: 10.1016/j.amc.2019.06.037.
- [82] "BioNetGen at Virtual Cell: Sample models."
- [83] "BEL.bio · BEL.bio."
- [84] L. Lahti, "A brief overview on the BioPAX and SBML standards for formal presentation of complex biological knowledge," *arXiv:1109.4919 [cs, q-bio]*, 2011/09/22/ 2011.
- [85] R. Scheines, P. Spirtes, C. Glymour, C. Meek, and T. Richardson, "The TETRAD Project: Constraint Based Aids to Causal Model Specification," (in en), *Multivariate Behavioral Research*, vol. 33, no. 1, pp. 65-117, 1998/01// 1998, doi: 10.1207/s15327906mbr3301_3.
- [86] C. Glymour, K. Zhang, and P. Spirtes, "Review of Causal Discovery Methods Based on Graphical Models," *Frontiers in Genetics*, vol. 10, p. 524, 2019/06/04/ 2019, doi: 10.3389/fgene.2019.00524.
- [87] P. O. Hoyer, S. Shimizu, A. J. Kerminen, and M. Palviainen, "Estimation of causal effects using linear non-Gaussian causal models with hidden variables," (in en), *International Journal of Approximate Reasoning*, vol. 49, no. 2, pp. 362-378, 2008/10// 2008, doi: 10.1016/j.ijar.2008.02.006.
- [88] G. Lacerda, P. L. Spirtes, J. Ramsey, and P. O. Hoyer, "[1206.3273] Discovering Cyclic Causal Models by Independent Components Analysis."
- [89] M. Kalisch, M. Mächler, D. Colombo, M. H. Maathuis, and P. Bühlmann, "Causal Inference Using Graphical Models with the R Package pcalg," (in en), *Journal of Statistical Software*, vol. 47, no. 11, 2012 2012, doi: 10.18637/jss.v047.i11.
- [90] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*, 2 ed. (Adaptive Computation and Machine Learning series). Cambridge, MA, USA: A Bradford Book (in en), 2001, p. 568.
- [91] P. L. Spirtes, C. Meek, and T. S. Richardson, "Causal Inference in the Presence of Latent Variables and Selection Bias," 2013 2013, doi: 10.48550/ARXIV.1302.4983.
- [92] D. Colombo, M. H. Maathuis, M. Kalisch, and T. S. Richardson, "Learning high-dimensional directed acyclic graphs with latent and selection variables," *The Annals of Statistics*, vol. 40, no. 1, 2012/02/01/ 2012, doi: 10.1214/11-AOS940.
- [93] M. H. Maathuis, M. Kalisch, and P. Bühlmann, "Estimating high-dimensional intervention effects from observational data," *The Annals of Statistics*, vol. 37, no. 6A, 2009/12/01/ 2009, doi: 10.1214/09-AOS685.

- [94] M. Scutari, "Learning Bayesian Networks with the bnlearn R Package," (in en), *Journal of Statistical Software*, vol. 35, no. 3, 2010 2010, doi: 10.18637/jss.v035.i03.
- [95] G. Ferguson and J. F. Allen, "TRIPS: An Integrated Intelligent Problem-Solving Assistant," (in en), p. 6.
- [96] R. Sharp *et al.*, "Eidos, INDRA, & Delphi: From Free Text to Executable Causal Models," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, 2019 2019, Minneapolis, Minnesota: Association for Computational Linguistics, pp. 42-47, doi: 10.18653/v1/N19-4008. [Online]. Available: <http://aclweb.org/anthology/N19-4008>
- [97] C. T. Hoyt, A. Konotopez, and C. Ebeling, "PyBEL: a computational framework for Biological Expression Language," (in en), *Bioinformatics*, vol. 34, no. 4, pp. 703-704, 2018/02/15/ 2018, doi: 10.1093/bioinformatics/btx660.
- [98] G. A. P. C. Burns, P. Dasigi, A. de Waard, and E. H. Hovy, "Automated detection of discourse segment and experimental types from the text of cancer pathway results sections," (in en), *Database*, vol. 2016, p. baw122, 2016 2016, doi: 10.1093/database/baw122.
- [99] A. de Waard and H. Pander Maat, "Epistemic Modality and Knowledge Attribution in Scientific Discourse: A Taxonomy of Types and Overview of Features," 2012/07// 2012, Jeju Island, Korea: Association for Computational Linguistics, pp. 47-55. [Online]. Available: <https://aclanthology.org/W12-4306>
- [100] K. Hengeveld and J. L. Mackenzie, *Functional Discourse Grammar*. Oxford University Press, 2009.
- [101] W. J. Wilbur, A. Rzhetsky, and H. Shatkay, "New directions in biomedical text annotation: definitions, guidelines and corpus construction," (in en), *BMC Bioinformatics*, vol. 7, no. 1, p. 356, 2006/12// 2006, doi: 10.1186/1471-2105-7-356.
- [102] "JSON."
- [103] G. Wu, H. Huang, J. Garcia Abreu, and X. He, "Inhibition of GSK3 phosphorylation of beta-catenin via phosphorylated PPPSPXS motifs of Wnt coreceptor LRP6," (in eng), *PloS one*, vol. 4, no. 3, pp. e4926-e4926, 2009, doi: 10.1371/journal.pone.0004926.
- [104] G. McKenzie *et al.*, "Cellular Notch responsiveness is defined by phosphoinositide 3-kinase-dependent signals," (in en), *BMC Cell Biology*, vol. 7, no. 1, p. 10, 2006/12// 2006, doi: 10.1186/1471-2121-7-10.
- [105] G. Frisoni, G. Moro, and A. Carbonaro, "A Survey on Event Extraction for Natural Language Understanding: Riding the Biomedical Literature Wave," *IEEE Access*, vol. 9, pp. 160721-160757, 2021 2021, doi: 10.1109/ACCESS.2021.3130956.

- [106] E. Holtzapple, C. A. Telmer, and N. Miskov-Zivanov, "FLUTE: Fast and reliable knowledge retrieval from biomedical literature," *Database*, vol. 2020, 2020 2020, doi: 10.1093/database/baaa056.
- [107] P. Gawron *et al.*, "MINERVA—a platform for visualization and curation of molecular interaction networks," (in en), *npj Systems Biology and Applications*, vol. 2, no. 1, p. 16020, 2016/12// 2016, doi: 10.1038/npjsba.2016.20.
- [108] A. Liu, P. Trairatphisan, E. Gjerga, A. Didangelos, J. Barratt, and J. Saez-Rodriguez, "From expression footprints to causal pathways: contextualizing large signaling networks with CARNIVAL," (in eng), *NPJ systems biology and applications*, vol. 5, p. 40, 2019 2019, doi: 10.1038/s41540-019-0118-z.
- [109] D. Domingo-Fernández, S. Mubeen, J. Marín-Llaó, C. T. Hoyt, and M. Hofmann-Apitius, "PathMe: merging and exploring mechanistic pathway knowledge," *BMC Bioinformatics*, vol. 20, no. 1, p. 243, 2019/05/15/ 2019, doi: 10.1186/s12859-019-2863-9.
- [110] M. J. Alvarez *et al.*, "Network-based inference of protein activity helps functionalize the genetic landscape of cancer," *Nature genetics*, vol. 48, no. 8, pp. 838-847, 2016/08// 2016, doi: 10.1038/ng.3593.
- [111] Y. Ahmed, C. Telmer, and N. Miskov-Zivanov, "CLARINET: Efficient learning of dynamic network models from literature," *Bioinformatics Advances*, 2021 2021, doi: 10.1093/bioadv/vbab006.
- [112] N. L. Catlett *et al.*, "Reverse causal reasoning: applying qualitative causal knowledge to the interpretation of high-throughput data," (in eng), *BMC bioinformatics*, vol. 14, p. 340, 2013/11/23/ 2013, doi: 10.1186/1471-2105-14-340.
- [113] M. Ostaszewski *et al.*, "Community-driven roadmap for integrated disease maps," *Briefings in Bioinformatics*, vol. 20, no. 2, pp. 659-670, 2019/03/25/ 2019, doi: 10.1093/bib/bby024.
- [114] A. Iyappan *et al.*, "Towards a Pathway Inventory of the Human Brain for Modeling Disease Mechanisms Underlying Neurodegeneration," (in eng), *Journal of Alzheimer's disease: JAD*, vol. 52, no. 4, pp. 1343-1360, 2016/04/12/ 2016, doi: 10.3233/JAD-151178.
- [115] S. S. Aghamiri, V. Singh, A. Naldi, T. Helikar, S. Soliman, and A. Niarakis, "Automated inference of Boolean models from molecular interaction maps using CaSQ," *Bioinformatics*, vol. 36, no. 16, pp. 4473-4482, 2020/08/15/ 2020, doi: 10.1093/bioinformatics/btaa484.
- [116] Y. Ahmed, C. Telmer, and N. Miskov-Zivanov, "ACCORDION: Clustering and Selecting Relevant Data for Guided Network Extension and Query Answering," *arXiv:2002.05748 [q-bio]*, 2020/05/12/ 2020.
- [117] K.-W. Liang, Q. Wang, C. Telmer, D. Ravichandran, P. Spirtes, and N. Miskov-Zivanov, "Methods to Expand Cell Signaling Models Using Automated Reading and Model

- Checking," in *Computational Methods in Systems Biology*, vol. 10545, J. Feret and H. Koepl Eds. Cham: Springer International Publishing, 2017, pp. 145-159.
- [118] K. Sayed, K. N. Bocan, and N. Miskov-Zivanov, "Automated Extension of Cell Signaling Models with Genetic Algorithm," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018/07// 2018, Honolulu, HI: IEEE, pp. 5030-5033, doi: 10.1109/EMBC.2018.8513431. [Online]. Available: <https://ieeexplore.ieee.org/document/8513431/>
 - [119] K. Gilboy, K. Sayed, N. Sundaram, K. Bocan, and N. Miskov-zivanov, "A Faster DiSH : Hardware Implementation of a Discrete Cell Signaling Network Simulator," *IEEE*, vol. 15213, 2018.
 - [120] N. Miskov-Zivanov, P. Zuliani, E. M. Clarke, and J. R. Faeder, "Studies of biological networks with statistical model checking: application to immune system cells," (in en), p. 2, 2013 2013.
 - [121] M. A. Valenzuela-Escárcega, G. Hahn-Powell, M. Surdeanu, and T. Hicks, "A Domain-independent Rule-based Framework for Event Extraction," in *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, 2015 2015, Beijing, China: Association for Computational Linguistics and The Asian Federation of Natural Language Processing, pp. 127-132, doi: 10.3115/v1/P15-4022. [Online]. Available: <http://aclweb.org/anthology/P15-4022>
 - [122] C. A. Telmer *et al.*, "Computational modeling of cell signaling and mutations in pancreatic cancer," (in en), p. 8.
 - [123] R. Akbani *et al.*, "Genomic Classification of Cutaneous Melanoma," *Cell*, vol. 161, no. 7, pp. 1681-1696, 2015/06/18/ 2015, doi: <https://doi.org/10.1016/j.cell.2015.05.044>.
 - [124] W. F. Hawse *et al.*, "Cutting Edge: Differential Regulation of PTEN by TCR, Akt, and FoxO1 Controls CD4 ⁺ T Cell Fate Decisions," (in en), *The Journal of Immunology*, vol. 194, no. 10, pp. 4615-4619, 2015/05/15/ 2015, doi: 10.4049/jimmunol.1402554.
 - [125] V. K. Sandhya *et al.*, "A network map of BDNF/TRKB and BDNF/p75NTR signaling system," (in en), *Journal of Cell Communication and Signaling*, vol. 7, no. 4, pp. 301-307, 2013/12// 2013, doi: 10.1007/s12079-013-0200-z.
 - [126] G. Santoni, C. Cardinali, M. Morelli, M. Santoni, M. Nabissi, and C. Amantini, "Danger- and pathogen-associated molecular patterns recognition by pattern-recognition receptors and ion channels of the transient receptor potential family triggers the inflammasome activation in immune cells and sensory neurons," (in en), *Journal of Neuroinflammation*, vol. 12, no. 1, p. 21, 2015 2015, doi: 10.1186/s12974-015-0239-2.
 - [127] D. Hanahan and Robert A. Weinberg, "Hallmarks of Cancer: The Next Generation," (in en), *Cell*, vol. 144, no. 5, pp. 646-674, 2011/03// 2011, doi: 10.1016/j.cell.2011.02.013.

- [128] J. W. Ramos, "The regulation of extracellular signal-regulated kinase (ERK) in mammalian cells," (in eng), *The International Journal of Biochemistry & Cell Biology*, vol. 40, no. 12, pp. 2707-2719, 2008 2008, doi: 10.1016/j.biocel.2008.04.009.