



Article

Faithful Narratives from Complex Conceptual Models: Should Modelers or Large Language Models Simplify Causal Maps?

Tyler J. Gandee ¹ and Philippe J. Giabbanelli ^{2,*} ¹ Department of Computer Science & Software Engineering, Miami University, Oxford, OH 45056, USA² Virginia Modeling, Analysis, and Simulation Center (VMASC), Old Dominion University, Norfolk, VA 23435, USA

* Correspondence: pgiabban@odu.edu

Abstract

(1) Background: Comprehensive conceptual models can result in complex artifacts, consisting of many concepts that interact through multiple mechanisms. This complexity can be acceptable and even expected when generating rich models, for instance to support ensuing analyses that find central concepts or decompose models into parts that can be managed by different actors. However, complexity can become a barrier when the conceptual model is used directly by individuals. A ‘transparent’ model can support learning among stakeholders (e.g., in group model building) and it can motivate the adoption of specific interventions (i.e., using a model as evidence base). Although advances in graph-to-text generation with Large Language Models (LLMs) have made it possible to transform conceptual models into textual reports consisting of coherent and faithful paragraphs, turning a large conceptual model into a very lengthy report would only displace the challenge. (2) Methods: We experimentally examine the implications of two possible approaches: asking the text generator to simplify the model, either via abstractive (LLMs) or extractive summarization, or simplifying the model through graph algorithms and then generating the complete text. (3) Results: We find that the two approaches have similar scores on text-based evaluation metrics including readability and overlap scores (ROUGE, BLEU, Meteor), but faithfulness can be lower when the text generator decides on what is an interesting fact and is tasked with creating a story. These automated metrics capture textual properties, but they do not assess actual user comprehension, which would require an experimental study with human readers. (4) Conclusions: Our results suggest that graph algorithms may be preferable to support modelers in scientific translations from models to text while minimizing hallucinations.

Keywords: causalgraphs; graph algorithms; graph-to-text; simplification text summarization

Academic Editor: Laura Po

Received: 30 August 2025

Revised: 26 September 2025

Accepted: 3 October 2025

Published: 7 October 2025

Citation: Gandee, T.J.; Giabbanelli, P.J. Faithful Narratives from Complex Conceptual Models: Should Modelers or Large Language Models Simplify Causal Maps? *Mach. Learn. Knowl. Extr.* **2025**, *7*, 116. <https://doi.org/10.3390/make7040116>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The notion of *trust* in Modeling and Simulation (M&S) often revolves around technical constructs such as verification and validation [1–3]. A model that is demonstrated to be sufficiently accurate in an applicable domain can be considered adequate or ‘fit-for-purpose’. While the credibility of a model does contribute to trust and eventual adoption, practitioners have observed that trust is a multi-faceted construct that requires more than validation. Even if a model is comprehensive, validated, and built from appropriate datasets under the supervision of renowned subject-matter experts, end-users do not necessarily trust it and use it to inform practices. As pointed out by Harper and colleagues,

transparency, communication, and documentation are important enablers of trust [4]. There are two broad (and sometimes complementary) approaches to promoting transparency: either we are *transparent when building* a model (which is familiar to M&S practitioners working with participatory methods) or we are *transparent by explaining* a model that was built (e.g., post-hoc explainable AI to convey black-box models). In this paper, we focus on the transparency of *conceptual models* in the form of *causal maps* (also known as causal loop diagrams or systems maps), which consist of labeled nodes with a clear directionality (e.g., we can have more or less ‘rain’ but not more or less ‘weather’), connected by directed typed edges (e.g., more rain causes more floods).

Participatory approaches provide *transparent methods to build* a model by eliciting core assumptions and model dynamics from participants. However, there is no guarantee that the resulting model is transparent to other end-users, or even to the participants who were involved in building it. For example, we can build a small model with an individual within an hour where all constructs and relationships are easy to follow in a diagram, but once we scale this process to a *group* then the conceptual model can become complex and hard to visualize. The goal of transparency in assumptions and limitations [5] is challenging to achieve in this situation, as participants would struggle to know what *is* (assumptions) or *is not* (limitations) in a model that consists of hundreds of constructs and many more relationships. This situation is particularly problematic in the participatory modeling context, where participants are both model providers (the model synthesizes their views) and consumers (the model supports their decision-making activities). If the model-building process is transparent but the result is hard to interpret, then this imbalance can leave participants with the impression of being ‘used’ to create a model that they still do not really trust. Without transparency in the end-product, we lose the benefits of conceptual modeling activities, such as learning among participants, fostering engagement, or promoting buy-in from key actors [6].

Approaches focused on *explainability* examine how to convey a model (potentially with hundreds of constructs and relationships) to participants. While metrics such as LIME and SHAP are familiar AI approaches for models such as classifiers or regressors, they are not applicable in the context of explaining models such as causal maps. Instead, we turn to reporting guidelines for simulation studies such as TRACE (TRANSPARENT and Comprehensive model Evaluation), which includes model description [7], or recommendations from the ISPOR-SMDM task force to achieve transparency by providing nontechnical documentation that covers the model’s variables and relationships [8]. These recommendations are supported by empirical studies on decision-support systems, showing that “revealing the system’s internal decision structure (global explanation) and explaining how it decides in individual cases (local explanation)” positively affects a range of constructs such as trust [9]. However, as exemplified by a systematic survey of models for obesity, about half of peer-reviewed studies do not provide documentation on the overall model diagram, the algorithms used, or how data is processed [10]. The realization that documentation is important but either missing or inconsistent has prompted a line of research in *automatically generating explanations* for a model. The development and application of computational methods supporting documentation is an open problem. In 2021, Wang et al. approached the problem using a small set of predefined natural-language templates that provide contrastive model-based explanations (how does the inclusion of another parameter affect the measured outcomes?), scenario-based explanations (how does one scenario affect the outcomes?), and goal-oriented explanations (to achieve a target, which options to include and exclude?). For example, the model-based explanation starts with “Under scenario *s*, compared with model *B*, model *A* includes «options», but excludes «options». Such configuration differences improve «performance measures»...” [11]

With the emergence of Generative AI (GenAI) models such as GPT 3, we showed in 2022 that a causal map could be turned into text without templates, instead producing varied outputs [12]. While early works lost parts of the model in the translation process and required extensive manual examples for fine-tuning, we later showed that models could be reliably turned into text without loss [13] and by using only a handful of examples [14] (i.e., few shot prompting). The use of GenAI to explain models is now becoming increasingly common, as LLMs synthesize design documents and simulation logs to provide explanations to various end-users [15–17]. Despite these advances, current solutions can produce a large amount of text when explaining complex models. For instance, a causal map with 361 concept nodes and 946 edges could turn into a report of almost 10,000 words, spanning 92 paragraphs [13]. While such long reports echo standard practices in M&S, they are problematic as pointed out by Uhrmacher and colleagues: “textual documentation of simulation studies is notoriously lengthy [as one model] results easily in producing 30 pages [thus] means for succinct documentation are needed” [18]. The need for shorter outputs was emphasized in an earlier panel by Nigel Gilbert: noting that the UK Prime Minister at the time limited memos to two pages, Gilbert observed that “it is likely to be a tough job for a policy analyst to boil down the results of a policy simulation to two pages.” [19] Necessarily, if a report is shorter, then we cannot cover every aspect of the model. Transparency, thus, results in a trade-off, as the textual explanation must be simple and short yet sufficiently rigorous to cover what matters in a model [20]. Being selective by creating shorter reports can be beneficial. In particular, Mitchell argues that *not* disclosing certain aspects for transparency is *better* to build trust in a system [21]. That is, the indiscriminate disclosure of every internal detail (e.g., listing every parameter and rule) would lead to cognitive overload and paradoxically undermine trust. However, shorter reports can impact key measures in different ways: shortening a report inevitably reduces how much of the model’s content is represented (lower coverage), but it should avoid affecting how the retained content reflects the underlying model (faithfulness), particularly in high-risk domains such as policy or healthcare [22].

Reports can be shortened either by simplifying the model through explicit criteria and then using established text generation methods, or by passing the complete model to a text generator with an additional summarization task. Although text summarization [23] and model simplification [24] have been extensively studied, they have often appeared in separate strands of literature and follow different objective functions. For example, text summarization maximizes coverage and relevancy while minimizing redundancy [25], and model simplification may involve structural (e.g., prune weak links, eliminate transitive edges that can be inferred) or semantic objectives. Our main contribution is to experimentally compare the effects of model simplification and text summarization to generate shorter reports of conceptual models, motivated by the need to build trust through succinct documentation. We focus on measurable textual qualities of generated summaries (e.g., coverage, fluency, faithfulness, readability) rather than direct end-user comprehension. Our objective is accomplished through two specific aims:

- We formalize and implement algorithms to simplify causal models by structural compression (i.e., skip intermediates between concepts until reaching a branching or looping structure) and semantic pruning (i.e., remove the least-central concept nodes).
- We compare our model simplification algorithms with seven text summarization algorithms, including GPT-4, accounting for both extractive (select existing sentences) and abstractive (craft new sentences) strategies.

The remainder of this paper is organized as follows. In Section 2, we provide a brief background on how large conceptual models arise and how they can be transformed into text, along with text summarization methods. In Section 3, we describe the algorithms used

to apply our summarization models and reduce our graph to a desired size. Our results are provided in Section 4, and we compare all generated summaries with handwritten (ground truth) summaries of the conceptual model. Lastly, we discuss our findings in Section 5. To support replicability, we provide our source code on an open third-party repository at <https://osf.io/whqkd>, accessed on 26 September 2025.

2. Background

2.1. Large Causal Maps: Creation and Explanation Challenges

Societal challenges such as public health issues are often multidisciplinary and multilevel [26,27]. For example, suicide prevention efforts involve a variety of stakeholders ranging from subject-matter experts (e.g., social epidemiology, clinical psychology), individuals with lived experiences (e.g., survivors), and the broader community (e.g., family members) [28]. Stakeholder groups can have different views [29]: parents describe assessment procedures as rigid while staff favor of standardized protocols [30]. Working towards a shared solution, thus, requires an understanding of the *views held by each participant*. While approaches such as focus groups help to identify themes within and across participant groups [31], they do not fully capture the *mental model* of each individual with regard to the problem. Individuals may agree on aspects of the problem (e.g., shared key constructs may be highly ranked) while disagreeing on the course of action, because they have different views of causal implications.

Participatory modeling [32] externalizes the views of individuals into models, such as cognitive or causal maps. Externalizing how individuals perceive the structure and function of a system allows facilitators and modelers to identify leverage points and perform trade-off analyses [33,34]. The multilevel and multidisciplinary nature of complex problems, thus, calls for the construction of comprehensive models that pool the knowledge provided by several individuals [35]. The group-level model is expected to be comprehensive, as it should help to identify issues such as unintended side-effects from possible interventions. Consequently, the group-level model has a much larger number of nodes and edges than the individual models (Figure 1), and it contains structures that are important for the dynamics of a problem (e.g., feedback loops). Typically, the nodes and edges in the group-level map are obtained by a simple aggregation process that takes the union of nodes and edges across individual maps. Since individuals would have some constructs in common, the size of the group-level map grows slower than the number of participants. Eventually, we may reach a plateau or saturation (e.g., see the accumulation curves in Figure 5 from [36]) whereby adding knowledge from an individual map leads to a negligible change in the group-level map [37,38]. Variations of the process may further filter the content of the map, for instance to avoid having ‘outlier’ constructs that were only mentioned by one individual and create noise in the overall model [39]. Other variations may only allow participants from using terms with a pre-determined list, which, thus, bounds the number of nodes in the group-level map [40]. While imposing such a constraint makes it simpler to combine maps (e.g., freely choosing terms means that the same idea may use different words and calls for semantic equivalence algorithms [41,42]), it artificially forces participants to think alike and it prevents the identification of new concepts.

There is no ‘typical’ size for a group-level map as it depends on the problem, the participants, and the methods (e.g., how maps are aggregated, whether terms are limited to a predefined list). In obesity research, the map developed by the Provincial Health Services (PHSA) of Canada has 98 nodes and 174 edges [43] while the Foresight Obesity Map has 108 nodes and 304 edges [44]. While neither would qualify as a ‘large’ graph by any means, they become nonetheless too large to be conveniently explored as a diagram. For instance, the Foresight Obesity Map was derided as looking “more like a spilled plate

of spaghetti than anything of use to policymakers” [44], with several articles insisting on its complexity [45,46] instead of focusing on its intended role of a decision-support tool. Even specialized software to interact with such conceptual models did not fully solve the problem, as a usability study documented the struggles of experts in interacting with the PHSA map via interactive visualizations [47]. Directly attempting to deal with large maps illustrates Mitchell’s point: a complete disclosure of low-level details can be overwhelming and a counter-productive use of transparency if the intent is to build trust. Obesity researchers realized that the Foresight “map is so complex that some have worried that its use would lead to despair and retreat from the problem”, leading to a *simplification* of the map using network science techniques [48].

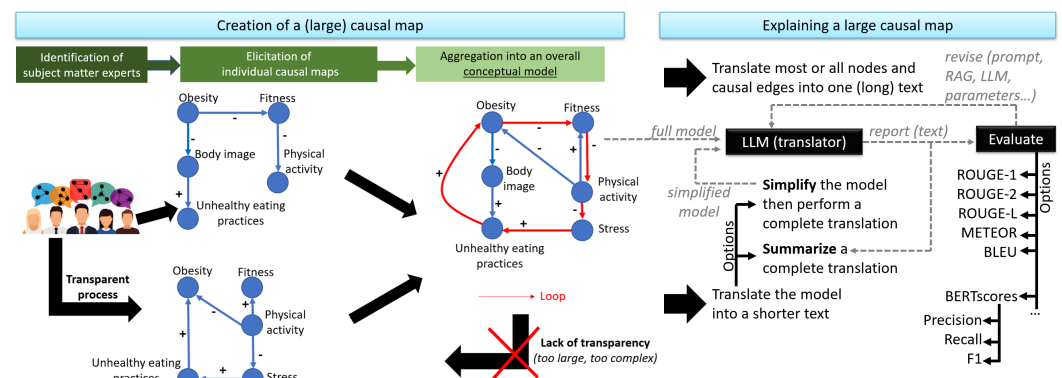


Figure 1. Large causal models can be produced in several ways, such as by aggregating the (smaller) mental models of several individuals. The aggregate model is larger than the individual ones and may exhibit new structures such as loops, which makes it harder to understand. Efforts in model-to-text translations via LLMs have focused on translating most or all of a model, whereas we examine how to produce a smaller report via model simplification or text summarization.

On the one hand, the use of Large Language Models presents opportunities to transform conceptual models into a format that is accessible to a broad audience, including those with visual impairments: textual reports. On the other hand, the *volume of text* remains a challenge and prevents the practical deployment of model-to-text as part of standards practices in participatory modeling. Transforming a large map into a report of 30 pages would just displace the problem: instead of facing one ‘spaghetti diagram’, participants would now be cognitively burdened by massive reports that make it difficult to focus on the key takeaways from the model. Consequently, this paper focuses on generating *shorter* reports, echoing how obesity researchers reduced large maps to help guide the conversations on policy development.

2.2. From Causal Maps to Text

The recent years have seen growing interest and multiple studies on transforming graphs into text [49–52], but these emerging methods may be inapplicable to conceptual models or require an intermediate transformation (e.g., into a text-attributed graph [53]). Even if method uses LLMs for some conceptual modeling tasks [54], it does not follow that it can be used for the translations of *causal maps* of interest in this paper. This section, thus, starts by briefly comparing the characteristics of different conceptual model, so the reader can appreciate how other LLM-based model-to-text transformations are related to our work. Causal maps, process models such as BPMN, and software models such as UML belong to the broader family of conceptual modeling approaches, but they differ in purpose, formalism, and intended use. Causal maps, also known as causal loop diagrams, are primarily used to externalize mental models and represent the stakeholders’ perceived cause-effect relationships among variables in a system. Causal maps consist of a labeled,

typed, directed graph. The nodes have labels that should be interpretable within the application context and that should have a clear direction of change. For example, weather lacks clarity and directionality (what does it mean to have more weather?), but the amount of rain would be acceptable. Edges are directed to indicate that a concept has a causal effect onto another one, and types specify whether the effect is a causal increase or decrease. Causal maps support participatory modeling and stakeholder engagement, for instance by comparing perspectives between individuals or groups, or identifying leverage points for interventions in a socio-environmental system. In contrast, process models such as BPMN are used to describe the sequential flow of activities, decisions, and events within organizational processes. These models are highly formalized, with strict syntax and semantics, and are often used for operational optimization, documentation, or automation of workflows. Several recent works have transformed BPMN diagrams into text [55–58]. Similarly, software models like UML serve to specify the structure and behavior of software systems. They include multiple diagram types, such as class diagrams and sequence diagrams, and are commonly integrated into software development lifecycles, often with support for formal analysis or code generation. Some studies have also used LLMs to transform UML diagrams into text, for instance to automate the process of providing feedback to students [59] or to help learners in interpreting models [60]. In sum, while BPMN and UML are typically used in technical or execution-oriented contexts with a high degree of formalism, causal maps are expected to be more interpretive and flexible to support collaborative understanding in complex, often interdisciplinary domains.

Early experiments with GPT-3 showed that asking LLMs to translate an *entire* model into text could lead to inaccurate outputs. Thus, LLMs are ‘spoon-fed’ with *parts of a model* to generate sentences that should be *fluent* (i.e., grammatically correct and coherent), *faithful* (i.e., accurately reflecting the model’s structure and semantics), and *cover* the input (i.e., include all relevant concepts and relations from the given model fragment) [12]. Our early work (Figure 2, top) identified how large the parts should be to achieve high scores with LLMs available at the time [12]. The *decomposition* of a model into smaller parts was achieved by a modified breadth-first search that originated from a node, selected a limited amount of its neighbors (to avoid a long list of connections) and avoided more than two hops. This strategy resembles the construction of local samples by Zhang et al. [61]. Since some nodes and edges were not selected in the sample, this information loss in the input structure resulted in information loss in the text. This lossy decomposition was not unique to our work, as other graph-to-text studies also dropped some of the content. For example, Li and colleagues noted that the input graph may not fit in the decoder and they pruned ‘irrelevant’ nodes [62].

While simplification can be a desirable feature, conflating the generation and simplification of text within a pre-processing step is problematic because it blurs the distinction between faithful model-to-text generation and deliberate simplification for communication purposes. Pre-processing routines are meant only to format the input for the LLM; if they also perform simplification, then the LLM’s output may reflect a distorted representation of the causal map rather than a faithful translation. This makes it impossible to attribute omissions or distortions to the LLM versus the pre-processing pipeline, undermines evaluation (since fidelity and coverage cannot be cleanly measured), and risks introducing hidden biases about which constructs are preserved or discarded. In other words, decomposing a model into manageable input chunks should maintain fidelity and completeness, while simplification should remain a *separate*, transparent step that can be explicitly justified, documented, and evaluated. Separating pre-processing and model-to-text generation from simplification is a key objective of our paper.

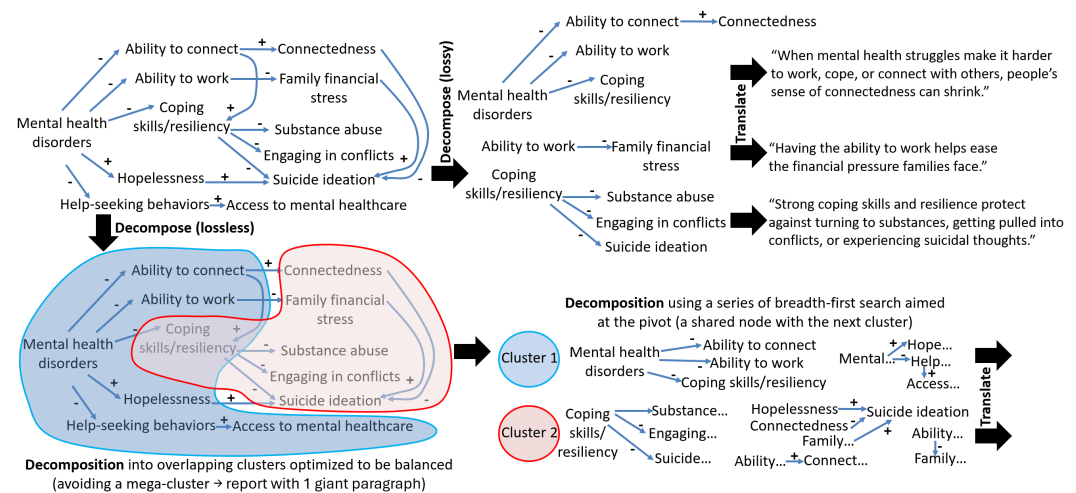


Figure 2. As large conceptual models should not be provided as a single object, our early work decomposed them with a lossy process that created simple sentences (here exemplified with GPT-5) but did not provide control over the simplifications [12]. Our more recent work uses a lossless process that decomposes the causal map into communities (corresponding to paragraphs) and further visits each community to output sentences in a logical order [13]. The nodes and edges exemplified here illustrate parts of a large open conceptual model on suicide, used in our experiments.

Our more recent work (Figure 2, bottom) provided a lossless decomposition pre-processing and organized sentences into thematically coherent paragraphs with a logical flow from one paragraph to the next [13]. This innovation is achieved by optimizing the choice of overlapping graph community detection algorithms to find large and thematically coherent parts of a model (thus, corresponding to a paragraph) that share some elements (thus, facilitating transitions between paragraphs). Within each community, another algorithm performs a modified form of breadth-first search to produce smaller parts, thus, finding the best order in which sentences should be organized within a paragraph. In particular, the search algorithm is oriented to end its exploration of the current community with an element share by the next community (i.e., a ‘pivot’ to smoothly transition between topics). This revised process better aligns the content with expectations for detailed documentation, which consists of paragraphs rather than just ‘bags of text’. However, the resulting reports can be very long when the input is a large causal map, thus, significantly exceeding the attention span associated with an executive summary.

2.3. Text Summarization: A Primer

Summarization algorithms fall into two broad categories (Figure 3). For a comprehensive survey on methods for text summarization, including early statistical approaches and recent developments with LLMs, we refer the readers to [63]. *Abstractive summarizations* resembles human-like summaries by creating new sentences based on the original text. These summaries should be *about* the input material, but they are not bound to the exact sentences of the input. Many abstractive summarization models are created through transfer learning, by pre-training a LLM and fine-tuning it for summarization tasks [64]. This allows tokens outside of the original text to be used to create a cohesive summary. DistilBART is a lightweight, distilled version of the BART model [65] that retains much of its generative capacity while being faster and more efficient. T5 (Text-To-Text Transfer Transformer) is a unified model that casts all NLP tasks, including summarization, as text generation problems, using a pre-trained encoder–decoder architecture [64]. DistilBART and T5 are widely adopted transformer architectures for general summarization tasks [63,66]. LED (Longformer Encoder-Decoder) extends BART with Longformer’s sparse attention to handle

much longer documents in an abstractive manner [67]. GPT-4 can perform summarization through instruction-following or few-shot prompting.

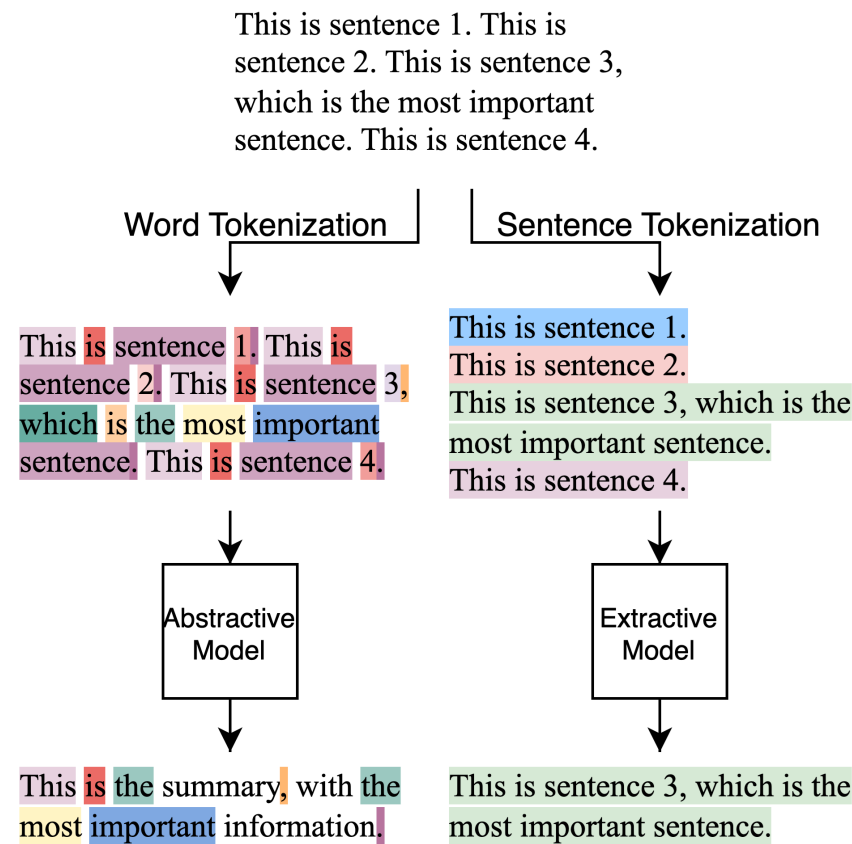


Figure 3. Extractive summarization resembles binary classifiers by deciding whether each sentence should be included or excluded from the summary. Abstractive summarization is similar to generative models as they can include text from their own knowledge base. Colored highlights indicate token or sentence boundaries, where each color represents a distinct token (in word tokenization) or sentence (in sentence tokenization) used to illustrate the difference between abstractive and extractive summarization approaches.

In contrast, the category of *extractive summarization* does not generate new sentences: it sorts existing sentences by significance and assembles them without changes. The resulting summary may feel disconnected between sentences, but outputting only existing sentences leaves no room for possible hallucinations, which may happen with abstractive approaches. TextRank is classical unsupervised baseline, widely used due to its simplicity and interpretability [68]. This unsupervised graph-based algorithm models sentences as nodes and their similarities as edges, applying a variant of the PageRank algorithm to rank and select the most central sentences. BertExt (BERT for extractive summarization) is a more modern neural approach that leverages contextual embeddings from a pre-trained BERT model to compute sentence representations and uses a neural classifier to score sentence relevance [69]. LongformerExt overcomes the input length limitations of models like BERT through its sparse attention mechanism [67], making it suitable to score many candidate sentences (as can be encountered with large causal models) beyond the length limitations of traditional transformers.

The quality of a summary is typically evaluated using automated quantitative metrics that compare generated summaries to reference (human-written) summaries based on textual overlap. In particular, six widely-used metrics capture different aspects of similarity. ROUGE-1 measures the overlap of single words between the generated and reference

summaries. It reflects basic recall: how many important individual words from the reference are preserved [70]. ROUGE-2 extends this to pairs of consecutive words, so we can evaluate the preservation of meaningful phrase structures. ROUGE-L focuses on the Longest Common Subsequence of words, emphasizing the preservation of sentence-level word order without requiring strict adjacency. It values for maintaining the general sequence and structure of information. ROUGE-Sum includes both single words and pairs of words that appear in order but not necessarily adjacent (i.e., skip bigrams), thus, providing an intermediate between ROUGE-1 and ROUGE-2. METEOR evaluates summaries based on semantic similarity (stemming, synonyms, paraphrases) instead of strictly adhering to word matches like ROUGE. BLEU (Bilingual Evaluation Understudy) measures the precision of sequences of n words (i.e., n -grams) in the generated summary, with a brevity penalty to discourage overly short outputs [71]. Overall, these metrics provide a multi-faceted evaluation of summaries: from simple content overlap (ROUGE-1), to fluency (ROUGE-2), structural fidelity (ROUGE-L), and semantic similarity (METEOR), with BLEU offering a stricter, precision-focused perspective. Since extractive methods copy sentences from the source text, they tend to perform well on metrics that value n -gram and sequence overlap (ROUGE-1, ROUGE-2, ROUGE-L) but they may score lower on metrics that reward varied phrasing or semantic richness (METEOR, BLEU). In addition, they may lack coherence at the paragraph level if the extracted sentences do not flow naturally, which may be reflected into other metrics on readability. Expectations are reversed for abstractive methods (score higher on METEOR and BLEU, lower on ROUGE), which may also hallucinate.

3. Methods

Our work compares two methods (Figure 4). In the modeler-led simplification, we use graph algorithms that can be controlled to reflect different priorities. In our case, the algorithms combine structural (e.g., compress chains of links to avoid intermediates) and semantic objectives (e.g., prioritize the preservation of the most central nodes). Then, the simplified model is turned into text using the open-source model-to-paragraphs method previously released at ER'25 [13]. In the LLM-led simplification, we take the entire map and immediately generate the paragraphs, then we perform summarization. Our process evaluates all seven text summarization algorithms listed in Section 2.3, covering abstraction methods (Distilbart, T5, LED, GPT-4) and extractive methods (TextRank, BertEXT, longformerEXT). The outputs from the two methods are then quantitatively evaluated. The two methods are formalized at a high-level in Algorithm 1, then Section 3.1 covers the custom algorithms for the modeler-led approach and Section 3.2 explains how we used text summarization algorithms.

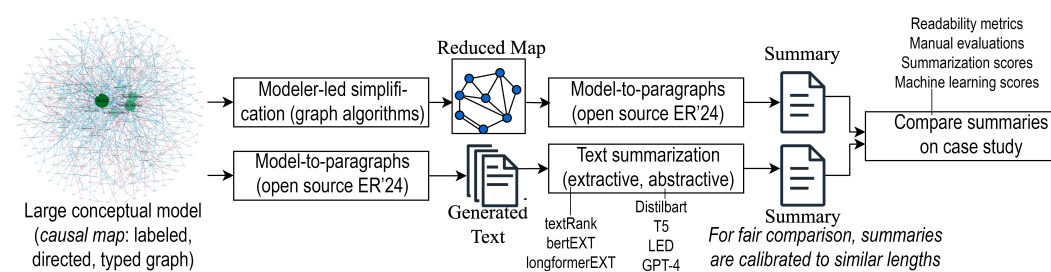


Figure 4. Overview of our methods.

Algorithm 1 Two Pipelines for Graph-to-Text Simplification**Require:** Causal graph $G = (V, E)$ **Ensure:** Simplified textual explanation T

- 1: Choose simplification strategy: Modeler-Led or LLM-Led
- 2: **if** Modeler-Led **then**
- 3: Apply graph simplification algorithm to G producing G'
- 4: Generate text T from G' using graph-to-text LLM
- 5: **else if** LLM-Led **then**
- 6: Generate initial text T_{raw} from full graph G using graph-to-text LLM
- 7: Apply text summarization (abstractive or extractive) to T_{raw} producing T
- 8: **return** T

3.1. Modeler-Led: Algorithms for Conceptual Model Simplification

By definition, a model is a *simplification* of a phenomenon. As the same phenomenon can be modeled at several levels of granularity, different schools of practice exist in the modeling community. For example, the KISS approach aims at the simplest model and only adds content when strictly necessary, whereas the KIDS approach starts with the most comprehensive model possible and then consider simplifying it in light of the available evidence [72]. Intuitively, two schools of thought distinguish the prevention of complexity (keeping a model simple) from post-hoc simplification (allowing a model to get complex before simplifying it) [73,74]. As showed algorithmically, these two approaches could produce different causal maps given the same evidence base [75]. For example, two concepts could be linked by more intermediary nodes, or connected by more alternate pathways. Automating a simplification process based solely on the model can be challenging, as we lack information about the intended *use* of the model (e.g., an apparently unnecessary intermediate between two concepts may have been highly meaningful for the model commissioner) and we cannot measure the *impact* on decision-making activities (e.g., is removal of edges a simplification or error removal?). While the literature on graph reduction offers many options (e.g., sparsification, coarsening, condensation [76]), these options are neither all applicable to causal maps nor able to provide a sufficient level of controls for modelers. In this section, we developed a series of simplification for causal maps that can be adapted by modelers based on their own needs. That is, the algorithms below are not intended to be universally applied to simplify models: rather, we provide them as tools that can be tailored by users, and we recommend that their applications consider the rationale for simplification, the expected effect, and the risks for validity [73].

The overall simplification process is orchestrated by Algorithm 2, which simplifies the conceptual model by progressively eliminating structural elements that do not contribute meaningfully to its overall connectivity or flow. It begins with a typical reduction step that removes edges pointing from a node to itself (i.e., self-loops). Then it relies on Algorithm 3 to prune excess connections from highly connected nodes by removing their least central neighbors, helping to reduce noise in dense regions of the models, which may have been over-detailed. The core of the algorithm is an iterative process that alternates between identifying and removing linear chains of nodes that serve only as pass-through points (via chain compression in Algorithm 4; see Figure 5—bottom), and trimming peripheral nodes that function purely as sources or sinks (i.e., endpoints with only incoming or outgoing edges; see (Figure 5—top). By repeating these steps until no further changes occur, our algorithm gradually distills the model down to its essential structure, simplifying over-detailed areas and intermediate concepts while preserving the key relational backbone.

In dense parts of the conceptual model, pruning (Algorithm 3) preserves the most important nodes while removing their less important neighbors to reduce density. The user

controls the removal process by stating that important nodes can keep up to *max* neighbors, which we experimentally set to 2.

The Chain Compression (Algorithm 4) identifies and collapses linear chains in the conceptual model. Starting from a given node, the algorithm traverses backward to find the chain's head and forward to find its tail, so that intermediate nodes (i.e., not involved in a branch or cycle) can be skipped. To avoid concurrent modifications on a data structure, the skipped nodes and edges are not directly removed by the algorithm; rather, they are identified and returned for removal. As a result, the conceptual model maintains connectivity and cumulative causal weights without intermediate concepts.

Algorithm 2 Function SimplifyGraph: input graph $G \mapsto$ simplified graph

```

1:  $n_{old} \leftarrow |V(G)|$  ▷ Initialize node and edge counts to detect convergence
2:  $e_{old} \leftarrow |E(G)|$ 
3:  $n_{new} \leftarrow -1$ 
4:  $e_{new} \leftarrow -1$ 
5:  $starting \leftarrow True$ 
6: Remove all self-loops from  $G$ 
7: for each node  $u \in V(G)$  do ▷ Step 1: Prune neighbors via betweenness centrality
8:    $T \leftarrow PRUNE(u, 2, betweenness\_centrality)$  ▷ See Alg. 2
9:   for each  $v \in T$  do
10:    Remove edge  $(u, v)$  from  $G$ 
11: Remove all nodes with degree 0 from  $G$  ▷ Step 2: Remove isolated nodes
12: while  $n_{new} \neq n_{old}$  or  $e_{new} \neq e_{old}$  do ▷ Step 3: Iteratively compress/trim model
13:   if not  $starting$  then
14:      $n_{old} \leftarrow n_{new}$ 
15:      $e_{old} \leftarrow e_{new}$ 
16:    $V_{void} \leftarrow \emptyset$  ▷ Nodes identified as non-removable
17:    $V_{remove} \leftarrow \emptyset$  ▷ Nodes marked for deletion
18:   for each node  $u \in V(G)$  do ▷ Step 3.1: Try to compress chains through  $u$ 
19:      $(C, isRemovable) \leftarrow CHAINCOMPRESSION(G, u, V_{void})$  ▷ See Alg. 4
20:     if  $isRemovable = true$  then
21:        $V_{remove} \leftarrow V_{remove} \cup C$ 
22:     else if  $isRemovable = false$  then
23:        $V_{void} \leftarrow V_{void} \cup C$ 
24:     if  $in-degree(u) = 0$  or  $out-degree(u) = 0$  then
25:        $V_{remove} \leftarrow V_{remove} \cup \{u\}$  ▷ Step 3.2: Trim sources and sinks
26: Remove all nodes in  $V_{remove}$  from  $G$ 
27:  $n_{new} \leftarrow |V(G)|$  ▷ Update node and edge counts for convergence check
28:  $e_{new} \leftarrow |E(G)|$ 
29:  $starting \leftarrow False$ 
30: Remove all nodes with degree 0 from  $G$  ▷ Remove new isolated nodes
31: return  $G$ 

```

Algorithm 3 Prune: Remove Least-Central Neighbors

```

1: Input: node, sub (neighbors of node), max (maximum edges to keep), centrality function  $f$ , graph  $G$ 
2: Output: Set of nodes to remove from node's outgoing edges
3: function PRUNE(node, sub, max,  $f$ ,  $G$ )
4:    $centralityMap \leftarrow \{(n, f(n, G)) \mid n \in sub\}$  ▷ Get centrality for each neighbor
   ▷ Sort neighbors by increasing centrality value
5:    $sorted \leftarrow$  nodes in  $centralityMap$ , ordered by centrality (lowest first)
6:   if  $max = 0$  then ▷ If no edges should be retained, remove all neighbors
7:     return  $sorted$ 
8:   else ▷ Keep the top 'max' most central neighbors; remove the rest
9:     return first  $|sorted| - max$  elements of  $sorted$ 

```

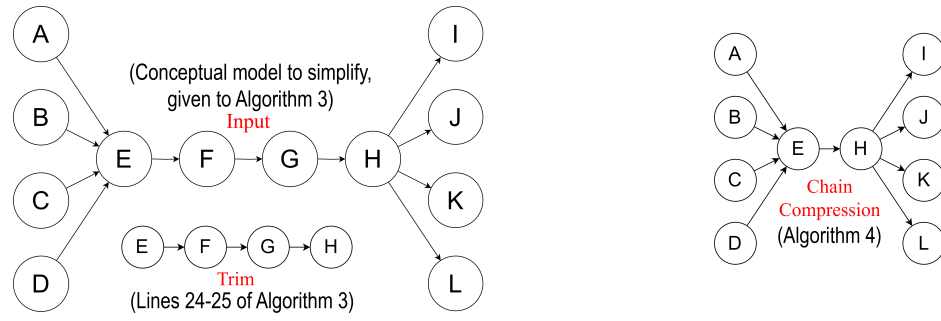


Figure 5. The input map with 12 nodes and 11 edges can be compressed (right) to remove intermediate parts of a chain or trimmed (bottom) to remove peripheral ideas.

Algorithm 4 Chain Compression in a Graph

```

1: Input: Directed graph  $G = (V, E)$ , node  $v \in V$ , set of preserved nodes  $\mathcal{V} \subseteq V$ 
2: Output: Set of nodes to remove  $S \subseteq V$ , and Boolean indicating whether compression occurred
3: function CHAINCOMPRESSION( $G, v, \mathcal{V}$ )
4:   if  $\mathcal{V} = \emptyset$  or  $v \notin \mathcal{V}$  then
5:      $S \leftarrow \{v\}$  ▷ Set of candidate nodes to remove (initially just  $v$ )
6:      $w \leftarrow v$  ▷ Pointer used to traverse the chain
7:      $weight \leftarrow 1$  ▷ Aggregated weight of the compressed chain
8:     while  $w$  has exactly one incoming edge  $(u, w)$  and  $u$  has exactly one outgoing edge do
9:       if  $u \in S$  then
10:        return  $S, \text{False}$  ▷ Cycle detected; abort compression
11:        $S \leftarrow S \cup \{u\}$  ▷ Add  $u$  to the set of removable nodes
12:        $weight \leftarrow weight \cdot weight(u, w)$  ▷ Multiply cumulative weight
13:        $w \leftarrow u$  ▷ Move pointer one step backward
14:        $head \leftarrow w$  ▷  $head$  is now the start of the chain
15:        $w \leftarrow v$  ▷ Reset pointer for forward traversal
16:       while  $w$  has exactly one outgoing edge  $(w, u)$  and  $u$  has exactly one incoming edge do
17:         if  $u = head$  then
18:          return  $S, \text{False}$  ▷ Cycle detected; abort compression
19:          $S \leftarrow S \cup \{u\}$  ▷ Add  $u$  to the set of removable nodes
20:          $weight \leftarrow weight \cdot weight(w, u)$  ▷ Multiply cumulative weight
21:          $w \leftarrow u$  ▷ Move pointer one step forward
22:          $tail \leftarrow w$  ▷  $tail$  is now the end of the chain
23:         if  $head \neq tail$  then
24:           Add edge  $(head, tail)$  to  $G$  with weight  $weight$  ▷ Insert compressed edge
25:           return  $S, \text{True}$  ▷ Return removable nodes and success flag
26:       return  $\emptyset, \text{False}$  ▷ No compression performed

```

3.2. LLM-Led: Applying Abstractive and Extractive Summarization

For a fair comparison of modeler- and LLM-led summarization, the summaries should have a similar length. Otherwise, results may not reflect a difference in methods but rather a difference in length. For example, metrics that rely on overlap between generated words and a ground-truth summary would be affected by length: longer summaries are more likely to contain the target words, thus, boosting recall-oriented scores. Conversely, precision-oriented scores may be lower with longer summaries, as there are more chances for hallucinations. To guarantee a fair comparison, we set the length of LLM-led summaries to match the length of the modeler-led summaries. This is accomplished by setting a ratio

parameter (from original text to summarized text) of 0.114 or by chunking, as explained at the end of this section.

To create summaries with extractive methods, *sentences* are tokenized, then clustered by K-means, finally the summarization occurs. Clustering reduces redundancy by grouping similar sentences and preserves coverage by ensuring that each semantic group is represented, thus, balancing content diversity and concision. In contrast, abstractive summarization tokenizes *words* to generate new sentences; thus, we cannot pre-process the data into clusters. The larger number of tokens can exceed a model's token limits, often ranging from 512 to 4096 tokens. A straightforward solution would be to use up the token limit, e.g., by processing the first 512 tokens followed by the next batch of 512 tokens. However, such fragmented inputs can yield confused outputs for abstractive summarization models, since they risk breaking the text mid-sentence or mid-thought. Preserving semantic coherence, thus, requires meaningful 'chunks', which may be below the token limits: for example, if coherent units have 200, 300, and 150 tokens then the first batch would have $200 + 300 = 500$ tokens (< 512). The well-established technique of 'chunking' or 'multi-stage summarization' divides the input into coherent units that fit within each model's limit. Given that the recent open-source for conceptual model produces paragraphs that have a coherent theme [13], we perform chunking at the level of paragraphs (Algorithm 5). Then, each batch is summarized and summaries are concatenated (Algorithm 6).

Algorithm 5 Chunking for Abstractive Summarization: (text, tokenizer, max_tokens) $\mapsto$ list of batches of paragraphs

```

1: paragraphs  $\leftarrow$  split text by newline           ▷ Split input text into individual paragraphs
2: batches  $\leftarrow \emptyset$                              ▷ Initialize empty list to hold paragraph batches
3: total_tokens  $\leftarrow$  tokenizer(text)               ▷ Count total tokens in the input text
4: batch_length  $\leftarrow \lceil \text{total\_tokens} / \text{max\_tokens} \rceil$    ▷ Estimate number of paragraphs per batch
5: div  $\leftarrow$  total_tokens                             ▷ Expected number of tokens per batch
6: i  $\leftarrow$  0                                           ▷ Initialize batch index
7: previous_difference  $\leftarrow$  total_tokens             ▷ Track previous token difference
8: difference  $\leftarrow -1$                                ▷ Initialize current token difference
9: for each paragraph in paragraphs do
10:   if paragraph  $\neq \emptyset$  then                       ▷ Ignore empty lines
11:     temp  $\leftarrow$  batch[i] concatenated with paragraph   ▷ Try adding paragraph to
current batch
12:     tokens  $\leftarrow$  tokenizer(temp)                     ▷ Token count of temporary batch
13:     diff  $\leftarrow | \text{tokens} - \text{div} |$                    ▷ Difference from expected token count
14:     if previous_difference < diff or tokens > max_tokens then ▷ Too far from target
or over limit
15:       i  $\leftarrow i + 1$                                    ▷ Start a new batch
16:       batch[i]  $\leftarrow$  paragraph                       ▷ Initialize new batch with this paragraph
17:       previous_difference  $\leftarrow$  tokens               ▷ Reset token difference tracker
18:     else
19:       batch[i]  $\leftarrow$  temp                             ▷ Append paragraph to current batch
20:       previous_difference  $\leftarrow$  diff                 ▷ Update token difference
21: return batches                                         ▷ Return the list of completed batches

```

While chunking (Algorithm 5) preserves paragraph integrity (we never split within a paragraph), it may nonetheless introduce discontinuities across chunks. For example, thematic links between consecutive paragraphs may be weakened if they are processed separately as summarization (Algorithm 6) may alter the order in which ideas connect across boundaries: the last paragraph of chunk i may originally flow into the first paragraph of chunk $i + 1$, but there is no guarantee that summaries preserve such connections. We

mitigated this risk by ensuring chunks followed the paragraph order produced by the graph-to-text process, which was already designed for thematic coherence. However, we acknowledge that residual discontinuities may remain, and addressing this through overlap strategies or discourse-aware summarization is an area for future work.

Algorithm 6 Generate Abstract

Input: Text, ModelName, Tokenizer, MaxTokens, Optional GroundTruthSummary

Output: Summary paragraphs generated from Text

```

1: function GENERATEABSTRACT(Text, ModelName, Tokenizer, MaxTokens,
   GroundTruthSummary)
2:   Batches  $\leftarrow$  CHUNKING(Text, Tokenizer, MaxTokens) ▷ See Alg. 5
3:   Summarizer  $\leftarrow$  PIPELINE(ModelName) ▷ Initialize summarization model
4:   if GroundTruthSummary  $\neq \emptyset$  then ▷ If there is truth summary, use its length
5:      $r \leftarrow$  WORDCOUNT(GroundTruthSummary)  $\div$  WORDCOUNT(Text)
6:     groundTruthParagraphs  $\leftarrow$  PARAGRAPHCOUNT(GroundTruthSummary)
7:   else
8:      $r \leftarrow 0.1$  ▷ Default length ratio
9:     groundTruthParagraphs  $\leftarrow 3$  ▷ Default number of output paragraphs
10:  results  $\leftarrow \emptyset$  ▷ We will track the list of batches and associated summaries
11:  for  $i \leftarrow 1, \dots, |Batches|$  do
12:    if Batches $i$  is not empty then
13:      MaxSummaryLength  $\leftarrow$  WORDCOUNT(Batches $i$ )  $\times (r + 0.05)$  ▷ Buffer
14:      Sentences  $\leftarrow$  SUMMARIZER(Batches $i$ , MaxSummaryLength)
15:      Clean-up sentences ▷ Depends on summarization; see paragraph below ▷
16:      if  $i \bmod \text{groundTruthParagraphs} = 0$  then
17:        Sentences  $\leftarrow$  Sentences + newline + newline ▷ Add paragraph break
18:      else
19:        Sentences  $\leftarrow$  Sentences + space ▷ Continue sentence inline
20:      Append (Batches $i$ , Sentences) to results ▷ input text and its summary
21:  return results
  
```

Various models have different pre-processing and post-processing techniques in encoding/decoding, which can result in poorly formatted text. For example, T5-base does not capitalize the first letter of a sentence. Algorithm 6, thus, includes a model-specific ‘clean-up’ step (line 15) in which we fix capitalization (for BART), spaces around punctuations (for BART, T5, LED), spaces around parentheses (for BART and T5), and line breaks (for BART and T5). These fixes were not needed when using GPT.

Our open-source *implementation is available online* on a permanent repository at <https://doi.org/10.5281/zenodo.15660803>, accessed on 26 September 2025. For extractive algorithms, our Python implementation used gensim 3.8.2 for TextRank (no token limit) and bert-extractivesummarizer 0.10.1 for both Longformer-Ext (token limit 4096) and BERT (token limit 512). For abstractive algorithms, we used transformers 4.35.2 for BART (1024 tokens), T5 (512 tokens), and LED (16,384). We used OpenAI 1.3.3 for GPT-4-Turbo (128,000 tokens). Given the token limitations, we used 11 batches to fit within BART’s limits and 24 batches to work with T5. The high token limit for LED was sufficient given our text size; thus, this specific model did not require chunking. *Theoretically*, GPT has a high token limit that does not require chunking either. However, remember that we need to control the length of the generated summaries for fair comparison with the modeler-led solution. When using LLMs such as GPT, setting a maximum number of output tokens to achieve a desired length did not work (it was always well below the limit); thus, *in practice* we used chunking to better control the summary’s length.

4. Results

4.1. Case Study

To allow for comparison with prior works, we use the same open-access case study [12,13] consisting of a large conceptual model on suicide with 361 concept nodes and 946 edges. This map exemplifies the type of conceptual model produced by a participatory process, with the unintended effect of becoming harder to understand. This conceptual model was developed by synthesizing the views of 15 subject-matter experts on suicide and Adverse Childhood Experiences (ACEs) among children and adolescents in the US; thus, it covers several domains, at multiple levels, and from different areas of expertise (e.g., behavioral science, psychiatry, epidemiology). To score the summaries generated by the two methods of interest with quantitative methods (e.g., ROUGE), experts wrote two summaries of similar length (three long paragraphs): one that summarizes the model from a socio-ecological perspective by grouping constructs at the individual, interpersonal, community, and societal level; and another that describes the content either as it relates to ACEs or suicide ideation.

4.2. Model Simplification

By applying the algorithms from Section 3.1 on the case study, we reduced the conceptual model from 361 nodes and 946 edges originally to 57 nodes (84% decrease) and 103 edges (89% decrease). Transforming this simplified model to text produces a summary with 945 words and 5979 characters, in comparison with almost 10,000 words when translating the original model to text.

4.3. Evaluations

For a thorough assessment, we completed the automatic quantitative scores of overlaps between generated and reference summaries (ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-Sum, Meteor, BLEU) presented in Section 2.2 with three categories of scores. First, *BERTScore* computes token-level embeddings using the pretrained language model BERT and aligns each token in the generated text with the most similar token in the reference summary based on cosine similarity [77]. This aggregated similarities are scrutinized through three scores: precision (are the generated tokens semantically similar to the reference?), recall (do the reference tokens appear in the generated output?), and F1 (the harmonic mean of precision and recall). Second, we used *manual scores* assigned by human raters to capture faithfulness and fluency on a five-point Likert scale, with one being the worst (e.g., many hallucinations) and five being the best (no hallucinations). Third, we should evaluate not just what information is included in a summary, but how it is *expressed*: readability is captured by Flesch Reading Ease, where a score of 50–60 indicates K-10 to K-12, 30–50 is college-level, and 0–30 is a college graduate [78]; diversity is measured by the number of unique words; and syntactic complexity is reported through the average number of words per sentence.

Our results for similarities between generated and reference summaries (Table 1) show that BART achieved the most number of high scores, including ROUGE-1, ROUGE-2, and ROUGE-LSum, and BERTScore recall, while textRANK achieved the highest Rouge-L score. Longformer-EXT achieved the same ROUGE-2 score as BART and outperformed on METEOR and BLEU, while LED achieved the highest BERTScore precision. BERTScore results were close on all three metrics. Using graph algorithms in the modeler-led simplification (bottom row) is competitive with other methods across all scores.

We note that scores for LED stand out, as it has the best precision (by 2 percentage points) but the worst ROUGE, MET, and BLEU scores by a wide margin. These notably lower performances of LED may stem from its optimization for handling very long doc-

uments, which in our case was unnecessary since our texts were relatively shorter after preprocessing. Consequently, LED's sparse attention mechanism may have underutilized contextual information compared to other abstractive models.

Table 1. Evaluation metrics for summarization models in relation to handwritten summaries with ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-LSum, METEOR, BLEU, and BERTscore for precision, recall, and F1 score.

Method	Model	R-1	R-2	R-L	R-LSum	MET	BLEU	Prec	Rec	F1
Extractive	textRANK	0.43	0.10	0.16	0.32	0.30	0.08	0.85	0.83	0.84
	bertEXT	0.45	0.09	0.15	0.22	0.29	0.06	0.85	0.83	0.84
	longformerEXT	0.45	0.11	0.15	0.29	0.31	0.10	0.85	0.83	0.84
Abstractive	distilbart	0.49	0.11	0.14	0.33	0.30	0.07	0.84	0.84	0.84
	T5	0.45	0.10	0.14	0.26	0.29	0.08	0.84	0.83	0.84
	LED	0.14	0.05	0.08	0.10	0.05	0.00	0.87	0.81	0.84
	GPT-4	0.42	0.08	0.14	0.29	0.32	0.05	0.83	0.83	0.83
	Graph	0.41	0.09	0.15	0.28	0.28	0.07	0.85	0.83	0.84

The close scores between modeler-led simplifications and the language models are noteworthy because the simplification had a disadvantage by design. Simplifying *then* summarizing limits the information that can be used for summarization: for example, if there are 27 edges connected to ACEs and we prune all but 2 edges, then the subsequent model for text generation believes there are only 2 edges. Furthermore, simplifying the structure of the graph means that the ensuing decomposition process will create a list of *new* subgraphs that did not necessarily exist with the original model. For instance, simplifications can create a subgraph from suicide ideation to attempt, whereas the original model may have resulted in two subgraphs (ideation and planning, planning and attempting).

For manual evaluation, we found extractive summaries to be the most faithful but the least fluent (Table 2). This was expected, as extractive summaries only utilize sentences from their original text rather than generating new sentences. The lower fluency stems from a lack of connection between sentences, unless sentences that were already next to each other in the original text were selected in the summary. For abstractive summarizations, we observed the opposite behavior. While abstractive summarizations were more fluent, there were cases of hallucinations. This was particularly concerning in our application context, as the generated summaries started to include nonexistent concepts as well as imaginary quotes from fake public health officials. For instance, LED included a quote from a fake doctor confidently stating that the CDC recommends electroshock therapy to address suicide, which is false since this is a psychiatric treatment used in extreme episodes (whereas the CDC's focus is on public health and community-level strategies) and it is simply not part of the CDC's prevention recommendations. There were also minor hallucinations in T5. For example, our summary referenced Simon Tisdall, who is a foreign affairs commentator (not a public health official or an expert on suicide), and is not mentioned in the original text. Distilbart and GPT-4 were the most faithful in abstractive summarizations.

Readability scores varied, but some of this variation is expected because the same system can be explained through different frameworks to produce materials that cater to various audiences. For example, using the socio-ecological framework or decomposing the conceptual model into prevention themes resulted in different Flesch Reading Ease scores. GPT had the lowest reading ease by far, in part because its very formal summaries used longer words such as 'exacerbate', 'strengthening', and 'community-focused'. The modeler-led simplification, along with BERT, Longformer, BART, and T5, all produced summaries that were accessible to pre-college readers.

Table 2. Evaluation metrics on manual aspects (faithfulness, fluency) and automatic readability by Flesch reading ease (FRE), lexical count, i.e., distinct words, and Words Per Sentence (WPS). Values with * are preferred because they are the closest to our textual representation of the reduced graph.

Method	Model	FRE	WPS	Lex. Ct.	Faithfulness	Fluency
Extractive	Original Text	53.61	17.56	8322	5	2
	Reference 1	51.07	20.05	782	5	5
	Reference 2	27.22	18.64	783	5	4
	textRANK	31.82	22.37	917	5	3
	bertEXT	54.42	16.76	922 *	5	3
	longformerEXT	53.41	17.84 *	981	5	3
Abstractive	distilbart	53.61	17.56	981	5	5
	T5	56.86	14.35	890	4	4
	LED	38.55	24.05	457	2	5
	GPT-4	7.96	20.90	836	5	4
	Graph	52.70	18.53	945	5	5

5. Discussion

In this paper, we proposed a new solution that improves the accessibility of large conceptual models reducing their size both structurally and textually. In particular, we have empirically shown that our proposed modeler-led algorithms for reducing the size of causal maps leads to a summary that is comparable to methods based on language models, while providing the high faithfulness and fluency scores achieved by few language models (e.g., BART).

While our study includes multiple categories of scores (e.g., overlap between summaries, readability, faithfulness and fluency), the ability of summaries to be *used* is ultimately the most important criterion. Future works should, thus, consider user studies to examine how summaries support key activities for participatory modeling such as identifying key causal mechanisms or facilitating group learning. Such investigations may also discover how the parameters involved in the generation of summaries (e.g., tone, length, selection of conceptual model components) depends on the list of intended activities, the characteristics of the audience, and the application domain.

While deriving conceptual models from text has been the subject of numerous recent works, the reciprocal relation of generating (explanatory) text from large model has received relatively less attention. There is also extensive work in graph reduction and text summarization, but never in-tandem with one another, as we have shown in our work. Consequently, there is currently a paucity of benchmarks that contain a causal map, a textual representation of the map, and summarized versions of the text and graph. The creation of such datasets is currently very labor intensive, as subject matter experts need to summarize large conceptual models, which is a challenging task that prompted the design of our methods.

The modeler-led approach discards information early in the process, which puts it at a disadvantage with respect to summaries generated by language models. Instead of permanently *deleting* nodes and/or edges during simplification, we could create metadata to track which entities were simplified and why (e.g., the rule that triggered removal). A hybrid approach could leverage the metadata by examining whether the summary misses key topics and, if so, trigger a rollback by using the metadata to reintegrate parts that should not have been pruned. Such reversibility and feedback-guided simplification could make conceptual model simplification more intelligent and adaptive.

Simplifications have been studied in other applications, including graph simplifications (e.g., removing self-loops) or neural network pruning techniques such as compressing chains [79] and node-centric pruning [80], which resembles our removal of sinks and

sources. While our algorithms are novel for conceptual modeling, we emphasize that our primary contribution is to *compare* simplifications by language models vs. algorithms controlled by modelers. This controlled simplification through explicit, documented rules (e.g., chain compression, pruning by centrality) can give modelers an advantage on interpretability by contrast with LLM-led summarization, which invokes opaque transformations that make it harder to trace why certain information is included or omitted. However, this potential for higher interpretability depends on using rules that are appropriate for the application domain (e.g., self-loops are valid self-monitoring mechanisms that should not be discarded in process diagrams) and that modelers can explain to the target audience. Our algorithms offer direct access to adapting the simplification controls, for instance by changing chain compression to preserve short chains, or using a different centrality measure to determine what is ‘important’. In particular, these centrality measures provide modelers with tools to control the consideration given to the different structural roles of the nodes in the model, such as by giving higher values to nodes that are involved in multiple pathways (e.g., betweenness or flow). Our open-source code and commented processes support researchers to adapt our work for their simplification needs. While we emphasize structural simplifications, semantics (e.g., concept labeled as ‘decision’ or ‘risk’) could play a role in making some nodes exempt from simplification, which can be explored in future works through interactive or ontology-informed model simplifications.

Author Contributions: Conceptualization, T.J.G. and P.J.G.; methodology, T.J.G. and P.J.G.; formal analysis, T.J.G. and P.J.G.; investigation, T.J.G. and P.J.G.; data curation, T.J.G.; writing—original draft preparation, T.J.G. and P.J.G.; visualization, T.J.G.; supervision, P.J.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Our source code is provided on an open third-party repository at <https://osf.io/whqkd>, accessed on 26 September 2025.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LLM Large Language Model

References

1. Yilmaz, L.; Liu, B. Model credibility revisited: Concepts and considerations for appropriate trust. *J. Simul.* **2022**, *16*, 312–325. [[CrossRef](#)]
2. Belfrage, M.; Johansson, E.; Lorig, F.; Davidsson, P. [In] Credible Models—Verification, Validation & Accreditation of Agent-Based Models to Support Policy-Making. *JASSS J. Artif. Soc. Soc. Simul.* **2024**, *27*, 4.
3. Bitencourt, J.; Osho, J.; Wooley, A.; Harris, G. Do you trust digital twins? A framework to support the development of trusted digital twins through verification and validation. *Int. J. Prod. Res.* **2025**, 1–21. [[CrossRef](#)]
4. Harper, A.; Mustafee, N.; Yearworth, M. Facets of trust in simulation studies. *Eur. J. Oper. Res.* **2021**, *289*, 197–213. [[CrossRef](#)]
5. Harper, A.; Mustafee, N.; Yearworth, M. The issue of trust and implementation of results in healthcare modeling and simulation studies. In Proceedings of the 2022 Winter Simulation Conference (WSC), Singapore, 11–14 December 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1104–1115.
6. Nguyen, L.K.N.; Kumar, C.; Jiang, B.; Zimmermann, N. Implementation of systems thinking in public policy: A systematic review. *Systems* **2023**, *11*, 64. [[CrossRef](#)]

7. Grimm, V.; Augusiak, J.; Focks, A.; Frank, B.M.; Gabsi, F.; Johnston, A.S.; Liu, C.; Martin, B.T.; Meli, M.; Radchuk, V.; et al. Towards better modelling and decision support: Documenting model development, testing, and analysis using TRACE. *Ecol. Model.* **2014**, *280*, 129–139. [\[CrossRef\]](#)
8. Eddy, D.M.; Hollingworth, W.; Caro, J.J.; Tsevat, J.; McDonald, K.M.; Wong, J.B. Model transparency and validation: A report of the ISPOR-SMDM Modeling Good Research Practices Task Force–7. *Med Decis. Mak.* **2012**, *32*, 733–743. [\[CrossRef\]](#)
9. Wanner, J.; Herm, L.V.; Heinrich, K.; Janiesch, C. The effect of transparency and trust on intelligent system acceptance: Evidence from a user-based study. *Electron. Mark.* **2022**, *32*, 2079–2102. [\[CrossRef\]](#)
10. Giabbanelli, P.J.; Tison, B.; Keith, J. The application of modeling and simulation to public health: Assessing the quality of agent-based models for obesity. *Simul. Model. Pract. Theory* **2021**, *108*, 102268. [\[CrossRef\]](#)
11. Wang, L.; Deng, T.; Zheng, Z.; Shen, Z.J.M. Explainable modeling in digital twin. In Proceedings of the 2021 Winter Simulation Conference (WSC), Phoenix, AZ, USA, 12–15 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–12.
12. Shrestha, A.; Mielke, K.; Nguyen, T.A.; Giabbanelli, P.J. Automatically explaining a model: Using deep neural networks to generate text from causal maps. In Proceedings of the WinterSim, Singapore, 11–14 December 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 2629–2640.
13. Gandee, T.J.; Giabbanelli, P.J. Combining Natural Language Generation and Graph Algorithms to Explain Causal Maps Through Meaningful Paragraphs. In Proceedings of the International Conference on Conceptual Modeling, Pittsburgh, PA, USA, 28–31 October 2024; Springer: Cham, Switzerland, 2024; pp. 359–376.
14. Giabbanelli, P.; Phatak, A.; Mago, V.; Agrawal, A. Narrating Causal Graphs with Large Language Models. In Proceedings of the 57th Hawaii International Conference on System Sciences (HICSS-57), Honolulu, HI, USA, 3–6 January 2024; p. 6.
15. Zhang, N.; Vergara-Marcillo, C.; Diamantopoulos, G.; Shen, J.; Tziritas, N.; Bahsoon, R.; Theodoropoulos, G. Large Language Models for Explainable Decisions in Dynamic Digital Twins. In Proceedings of the 5th International Conference on Dynamic Data Driven Applications Systems (DDDAS) 2024, New Brunswick, NJ, USA, 6–8 November 2024.
16. Giabbanelli, P.J.; Agrawal, A. Towards Personalized Explanations for Health Simulations: A Mixed-Methods Framework for Stakeholder-Centric Summarization. In Proceedings of the AAAI Fall Symposium Series on Safe, Ethical, Certified, Uncertainty-aware, Robust, and Explainable AI for Health (SECURE-AI4H), Arlington, VA, USA, 6–8 November 2025.
17. Giabbanelli, P.J.; Dumas, C.; Flandre, N.Y.; Pitkar, A.; Vazquez-Estrada, J. Promoting Empathy in Decision-Making by Turning Agent-Based Models into Stories Using Large-Language Models. *J. Simul.* **2025**, 1–21. [\[CrossRef\]](#)
18. Uhrmacher, A.M.; Frazier, P.; Hähnle, R.; Klügl, F.; Lorig, F.; Ludäscher, B.; Nenzi, L.; Ruiz-Martin, C.; Rumpe, B.; Szabo, C.; et al. Context, composition, automation, and communication: The C2AC roadmap for modeling and simulation. *ACM Trans. Model. Comput. Simul.* **2024**, *34*, 1–51. [\[CrossRef\]](#)
19. Tolk, A.; Clemen, T.; Gilbert, N.; Macal, C.M. How can we provide better simulation-based policy support? In Proceedings of the 2022 Annual Modeling and Simulation Conference (ANNSIM), San Diego, CA, USA, 18–20 July 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 188–198.
20. Mendoza, G.A.; Prabhu, R. Participatory modeling and analysis for sustainable forest management: Overview of soft system dynamics models and applications. *For. Policy Econ.* **2006**, *9*, 179–196. [\[CrossRef\]](#)
21. Mitchell, T. Trust and Transparency in Artificial Intelligence: T. Mitchell. *Philos. Technol.* **2025**, *38*, 87. [\[CrossRef\]](#)
22. Herrera, F. Making Sense of the Unsensible: Reflection, Survey, and Challenges for XAI in Large Language Models Toward Human-Centered AI. *arXiv* **2025**, arXiv:2505.20305.
23. El-Kassas, W.S.; Salama, C.R.; Rafea, A.A.; Mohamed, H.K. Automatic text summarization: A comprehensive survey. *Expert Syst. Appl.* **2021**, *165*, 113679. [\[CrossRef\]](#)
24. Liu, Y.; Safavi, T.; Dighe, A.; Koutra, D. Graph summarization methods and applications: A survey. *ACM Comput. Surv. (CSUR)* **2018**, *51*, 1–34. [\[CrossRef\]](#)
25. Verma, P.; Om, H. MCRMR: Maximum coverage and relevancy with minimal redundancy based multi-document summarization. *Expert Syst. Appl.* **2019**, *120*, 43–56. [\[CrossRef\]](#)
26. Rutter, H.; Savona, N.; Glonti, K.; Bibby, J.; Cummins, S.; Finegood, D.T.; Greaves, F.; Harper, L.; Hawe, P.; Moore, L.; et al. The need for a complex systems model of evidence for public health. *Lancet* **2017**, *390*, 2602–2604. [\[CrossRef\]](#) [\[PubMed\]](#)
27. Russo, F.; Broadbent, A.; Castellani, B.; Fustolo-Gunnink, S.; Rod, N.H.; Rod, M.H.; Moore, S.; Rutter, H.; Stronks, K.; Uleman, J. A Pluralistic (Mosaic) Approach to Causality in Health Complexity. In *The Routledge Handbook of Causality and Causal Methods*; Routledge: London, UK, 2024; pp. 241–253.
28. Pearce, T.; Maple, M.; Wayl, S.; McKay, K.; Woodward, A.; Brooks, A.; Shakeshaft, A. A mixed-methods systematic review of suicide prevention interventions involving multisectoral collaborations. *Health Res. Policy Syst.* **2022**, *20*, 40. [\[CrossRef\]](#)
29. Reed, M.S.; Barbrook-Johnson, P. Complex systems methods for impact evaluation: Lessons from the evaluation of an environmental boundary organisation. *Mires Peat* **2022**, *28*, 34.

30. Kodish, T.; Kim, J.J.; Le, K.; Yu, S.H.; Bear, L.; Lau, A.S. Multiple stakeholder perspectives on school-based responses to student suicide risk in a diverse public school district. *Sch. Ment. Health* **2020**, *12*, 336–352. [\[CrossRef\]](#)
31. Wilkinson, S. Focus groups. *Doing Soc. Psychol. Res.* **2004**, *23*, 344–376.
32. Voinov, A.; Jenni, K.; Gray, S.; Kolagani, N.; Glynn, P.D.; Bommel, P.; Prell, C.; Zellner, M.; Paolisso, M.; Jordan, R.; et al. Tools and methods in participatory modeling: Selecting the right tool for the job. *Environ. Model. Softw.* **2018**, *109*, 232–255. [\[CrossRef\]](#)
33. Gray, S.; Sterling, E.J.; Aminpour, P.; Goralnik, L.; Singer, A.; Wei, C.; Akabas, S.; Jordan, R.C.; Giabbanelli, P.J.; Hodbod, J.; et al. Assessing (social-ecological) systems thinking by evaluating cognitive maps. *Sustainability* **2019**, *11*, 5753. [\[CrossRef\]](#)
34. Hovmand, P.S. Group model building and community-based system dynamics process. In *Community Based System Dynamics*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 17–30.
35. Voinov, A.; Bousquet, F. Modelling with stakeholders. *Environ. Model. Softw.* **2010**, *25*, 1268–1281. [\[CrossRef\]](#)
36. White, C.T.; Mitsova, H.; BenDor, T.K.; Foy, K.; Pala, O.; Vukomanovic, J.; Meentemeyer, R.K. Spatially explicit fuzzy cognitive mapping for participatory modeling of stormwater management. *Land* **2021**, *10*, 1114. [\[CrossRef\]](#)
37. Tomoaia-Cotisel, A.; Allen, S.D.; Kim, H.; Andersen, D.F.; Qureshi, N.; Chalabi, Z. Are we there yet? Saturation analysis as a foundation for confidence in system dynamics modeling, applied to a conceptualization process using qualitative data. *Syst. Dyn. Rev.* **2024**, *40*, e1781. [\[CrossRef\]](#)
38. Singh, P.K.; Chudasama, H. Assessing impacts and community preparedness to cyclones: A fuzzy cognitive mapping approach. *Clim. Change* **2017**, *143*, 337–354. [\[CrossRef\]](#)
39. Schuerkamp, R.; Giabbanelli, P.J.; Grandi, U.; Dautre, S. How to combine models? Principles and mechanisms to aggregate fuzzy cognitive maps. In Proceedings of the 2023 Winter Simulation Conference (WSC), San Antonio, TX, USA, 10–13 December 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 2518–2529.
40. Gray, S.; Hilsberg, J.; McFall, A.; Arlinghaus, R. The structure and function of angler mental models about fish population ecology: The influence of specialization and target species. *J. Outdoor Recreat. Tour.* **2015**, *12*, 1–13. [\[CrossRef\]](#)
41. Giabbanelli, P.J.; Tawfik, A.A. Reducing the gap between the conceptual models of students and experts using graph-based adaptive instructional systems. In Proceedings of the International Conference on Human-Computer Interaction, Copenhagen, Denmark, 19–24 July 2020; Springer: Cham, Switzerland, 2020; pp. 538–556.
42. Freund, A.J.; Giabbanelli, P.J. Automatically combining conceptual models using semantic and structural information. In Proceedings of the 2021 Annual Modeling and Simulation Conference (ANNSIM), Fairfax, VA, USA, 19–22 July 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–12.
43. Drasic, L.; Giabbanelli, P.J. Exploring the interactions between physical well-being, and obesity. *Can. J. Diabetes* **2015**, *39*, S12–S13. [\[CrossRef\]](#)
44. McPherson, K.; Marsh, T.; Brown, M. Foresight report on obesity. *Lancet* **2007**, *370*, 1755. [\[CrossRef\]](#)
45. Allender, S.; Owen, B.; Kuhlberg, J.; Lowe, J.; Nagorcka-Smith, P.; Whelan, J.; Bell, C. A community based systems diagram of obesity causes. *PLoS ONE* **2015**, *10*, e0129683. [\[CrossRef\]](#)
46. Papoutsis, C.; Shaw, J.; Paparini, S.; Shaw, S. We need to talk about complexity in health research: Findings from a focused ethnography. *Qual. Health Res.* **2021**, *31*, 338–348. [\[CrossRef\]](#)
47. Giabbanelli, P.J.; Vesuvala, C.X. Human factors in leveraging systems science to shape public policy for obesity: A usability study. *Information* **2023**, *14*, 196. [\[CrossRef\]](#)
48. Finegood, D.T.; Merth, T.D.; Rutter, H. Implications of the foresight obesity system map for solutions to childhood obesity. *Obesity* **2010**, *18*, S13–S16. [\[CrossRef\]](#) [\[PubMed\]](#)
49. Jin, B.; Liu, G.; Han, C.; Jiang, M.; Ji, H.; Han, J. Large language models on graphs: A comprehensive survey. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 8622–8642. [\[CrossRef\]](#)
50. He, J.; Yang, Y.; Long, W.; Xiong, D.; Gutierrez-Basulto, V.; Pan, J.Z. Evaluating and Improving Graph to Text Generation with Large Language Models. *arXiv* **2025**, arXiv:2501.14497. [\[CrossRef\]](#)
51. Yuan, S.; Faerber, M. Evaluating Generative Models for Graph-to-Text Generation. In Proceedings of the Recent Advances in Natural Language Processing, Varna, Bulgaria, 4–6 September 2023; pp. 1256–1264.
52. Li, Y.; Li, Z.; Wang, P.; Li, J.; Sun, X.; Cheng, H.; Yu, J.X. A Survey of Graph Meets Large Language Model: Progress and Future Directions. In Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24, Jeju, Republic of Korea, 3–9 August 2024; Larson, K., Ed.; International Joint Conferences on Artificial Intelligence Organization: Montreal, QC, Canada, 2024; pp. 8123–8131.
53. Wang, Z.; Liu, S.; Zhang, Z.; Ma, T.; Zhang, C.; Ye, Y. Can LLMs Convert Graphs to Text-Attributed Graphs? In Proceedings of the Annual Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL), Albuquerque, NM, USA, 29 April–4 May 2025.

54. Flandre, N.Y.; Giabbanelli, P.J. Can large language models learn conceptual modeling by looking at slide decks and pass graduate examinations? An empirical study. In Proceedings of the International Conference on Conceptual Modeling, Pittsburgh, PA, USA, 28–31 October 2024; Springer: Berlin/Heidelberg, Germany, 2024; pp. 198–208.
55. Köpke, J.; Safan, A. Efficient LLM-based conversational process modeling. In Proceedings of the International Conference on Business Process Management, Krakow, Poland, 1–6 September 2024; Springer: Cham, Switzerland, 2024; pp. 259–270.
56. Tel, T.; Minor, M. Utilizing the Structure of Process Models for Guided Generation of Explanatory Texts. In Proceedings of the International Conference on Case-Based Reasoning, Biarritz, France, 30 June–3 July 2025; Springer: Cham, Switzerland, 2025; pp. 157–171.
57. Kourani, H.; Berti, A.; Hennrich, J.; Kratsch, W.; Weidlich, R.; Li, C.Y.; Arslan, A.; Schuster, D.; van der Aalst, W.M. Leveraging large language models for enhanced process model comprehension. *arXiv* **2024**, arXiv:2408.08892.
58. Minor, M.; Kaucher, E. Retrieval augmented generation with LLMs for explaining business process models. In Proceedings of the International Conference on Case-Based Reasoning, Merida, Mexico, 1–4 July 2024; Springer: Cham, Switzerland, 2024; pp. 175–190.
59. Gürtl, S.; Schimetta, G.; Kerschbaumer, D.; Liut, M.; Steinmaurer, A. Automated Feedback on Student-Generated UML and ER Diagrams Using Large Language Models. *arXiv* **2025**, arXiv:2507.23470. [[CrossRef](#)]
60. Bashiri, H.; Khalilipour, A.; Bakhtiari, P.; Challenger, M. Large Language Models as an Assistant to Interpret UML Models in Model-Based Engineering: An Exploratory Study. *Artif. Intell.* **2024**, *1*, 45–50. [[CrossRef](#)]
61. Zhang, S.; Zheng, D.; Zhang, J.; Zhu, Q.; Adeshina, S.; Faloutsos, C.; Karypis, G.; Sun, Y. Hierarchical compression of text-rich graphs via large language models. *arXiv* **2024**, arXiv:2406.11884.
62. Li, L.; Geng, R.; Li, B.; Ma, C.; Yue, Y.; Li, B.; Li, Y. Graph-to-text generation with dynamic structure pruning. *arXiv* **2022**, arXiv:2209.07258.
63. Zhang, H.; Yu, P.S.; Zhang, J. A systematic survey of text summarization: From statistical methods to large language models. *ACM Computing Surveys* **2024**, *57*, 277. [[CrossRef](#)]
64. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67.
65. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv* **2019**, arXiv:1910.13461.
66. Daraghmi, E.; Atwe, L.; Jaber, A. A Comparative Study of PEGASUS, BART, and T5 for Text Summarization Across Diverse Datasets. *Future Internet* **2025**, *17*, 389. [[CrossRef](#)]
67. Beltagy, I.; Peters, M.E.; Cohan, A. Longformer: The long-document transformer. *arXiv* **2020**, arXiv:2004.05150. [[CrossRef](#)]
68. Mihalcea, R.; Tarau, P. TextRank: Bringing order into text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–26 July 2004; pp. 404–411.
69. Liu, Y.; Lapata, M. Text Summarization with Pretrained Encoders. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; Inui, K., Jiang, J., Ng, V., Wan, X., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 3730–3740.
70. Lin, C.Y. Rouge: A package for automatic evaluation of summaries. In Proceedings of the Text Summarization Branches out, Barcelona, Spain, 25–26 July 2004; pp. 74–81.
71. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002; pp. 311–318.
72. Edmonds, B.; Moss, S. From KISS to KIDS—an ‘anti-simplistic’ modelling approach. In Proceedings of the International Workshop on Multi-Agent Systems and Agent-Based Simulation, New York, NY, USA, 19 July 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 130–144.
73. Robinson, S.; Brooks, R. Assumptions and simplifications in discrete-event simulation modelling. *J. Simul.* **2024**, 1–18. [[CrossRef](#)]
74. van der Zee, D.J. Approaches for simulation model simplification. In Proceedings of the 2017 Winter Simulation Conference (WSC), Las Vegas, NV, USA, 3–6 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 4197–4208.
75. Freund, A.J.; Giabbanelli, P.J. The necessity and difficulty of navigating uncertainty to develop an individual-level computational model. In Proceedings of the International Conference on Computational Science, Krakow, Poland, 16–18 June 2021; Springer: Cham, Switzerland, 2021; pp. 407–421.
76. Hashemi, M.; Gong, S.; Ni, J.; Fan, W.; Prakash, B.A.; Jin, W. A comprehensive survey on graph reduction: Sparsification, coarsening, and condensation. In Proceedings of the IJCAI ’24: Thirty-Third International Joint Conference on Artificial Intelligence, Jeju, Republic of Korea, 3–9 August 2024.
77. Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K.Q.; Artzi, Y. Bertscore: Evaluating text generation with bert. *arXiv* **2019**, arXiv:1904.09675.
78. Flesch, R. A new readability yardstick. *J. Appl. Psychol.* **1948**, *32*, 221. [[CrossRef](#)]

79. Schoonhoven, R.; Hendriksen, A.A.; Pelt, D.M.; Batenburg, K.J. LEAN: Graph-based pruning for convolutional neural networks by extracting longest chains. *arXiv* **2020**, arXiv:2011.06923.
80. Shokouhinejad, H.; Razavi-Far, R.; Higgins, G.; Ghorbani, A.A. Node-Centric Pruning: A Novel Graph Reduction Approach. *Mach. Learn. Knowl. Extr.* **2024**, *6*, 2722–2737. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.