MindTrack: Extracting Thematic Structures via Self-Attention

Charoonroj Amornpativet
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
charoonroj.amo@gmail.com

Virach Sornlertlamvanich
AAII, Faculty of Data Science
Musashino University
Tokyo, Japan
virach@musashino-u.ac.jp, virach@gmail.com

Pizzanu Kanongchaiyos
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
pizzanu.k@chula.ac.th

***Abstract*—This paper introduces MindTrack, a novel method for extracting thematic structures from text by leveraging self-attention mechanisms. The approach involves extracting keyphrases from each sentence, identifying a central source keyphrase, and inferring semantic relationships between keyphrases to build a thematic tree structure. The framework combines unsupervised keyphrase extraction with semantic relation modeling to construct interpretable topic maps without relying on external annotations or supervised learning.**

I. Introduction

In conversations and meetings, identifying the current topic of discussion at any given moment, understanding its connection to prior dialogue, and tracking thematic progression are crucial. Traditional note-taking or summarization approaches are typically static and linear, lacking the capacity to capture the dynamic and evolving nature of human conversations. In contrast, dynamic thematic tracking focuses on capturing the ongoing topic of each sentence, identifying where it originated in the conversation, and mapping how topics emerge, shift, or converge.

To represent this thematic structure clearly and intuitively, we propose the generation of a Topic Map from the input text. Similar to concept maps and discourse graphs [1], [2] , which have been used to visualize knowledge and textual coherence, a Topic Map visualizes the core ideas and their interrelations as a mind map. This approach provides users with a high-level overview of the content, along with the ability to drill down into subtopics and explore how different parts of the text contribute to the overall meaning. It serves as both a summary and a cognitive model of the structure of the conversation or the article.

At the core of our proposed method, MindTrack, lies the self-attention mechanism introduced in Transformer-based language models [3]. Self-attention allows a model to compute contextual relationships between all token pairs in a sequence, assigning weights (attention scores) that reflect the importance of one token to another. In our system, we use these attention scores, along with multiple scoring algorithms, to extract keyphrases, determine their source segments within the text, and identify semantic links between concepts. This process forms the basis for generating the Topic Map in an unsupervised manner.

The main advantage of this approach is that it can leverage powerful pre-trained language models, which already encode rich semantic information from large text corpora. This reduces the need for manual feature engineering or labeled data. In addition, attention-based methods are flexible and scalable and are capable of handling long and complex documents. However, a significant drawback is interpretability, as attention weights do not always reflect true model reasoning [4].

In this paper, we introduce MindTrack as a framework for extracting thematic structures using self-attention mechanisms. We present our methodology for identifying topics and building topic maps, discuss the advantages and limitations of attention-based extraction, and demonstrate the effectiveness of our approach through qualitative and quantitative analysis on conversational and textual data.

II. Literature review and based concept

The design of MindTrack is founded upon two fundamental concepts essential for generating a summary Topic Map: (A) keyphrase extraction using self-attention mechanisms, and (B) semantic relation extraction for constructing structured topic graphs. These concepts enable the system to identify meaningful content and understand how it is interconnected, which are crucial for effective topic tracking and summarization.

A. Keyphrase Extraction Using Self-Attention

Transformer-based models such as BERT [5] and GPT [6] produce self-attention maps that capture interactions between tokens within a sequence.SAMRank (Self-Attention Map Rank) [7] is an unsupervised method that utilizes these attention maps to identify and rank keyphrases based on their interaction within the context of the entire text.

SAMRank calculates two types of attention-based scores for each candidate keyphrase. The first is the Global Attention Score, which measures how much attention a keyphrase receives from other tokens. This reflects the prominence or centrality of the phrase in the attention distribution. The second is the Proportional Score, which measures how much attention the keyphrase gives to other tokens, capturing its contextual influence within the text. By summing these two components, SAMRank computes a final importance score for each keyphrase and ranks them accordingly.

Because it relies only on the attention scores from a single layer and head of a pretrained model, SAMRank does not require annotated data or task-specific training. This makes it suitable for general-purpose, unsupervised keyphrase extraction. MindTrack adopts a similar approach to rank keyphrases in both conversational and document-based text. These keyphrases are then used as nodes in the Topic Map, enabling structured and interpretable thematic tracking.

B. Semantic Relation Extraction

In addition to identifying keyphrases, constructing a Topic Map requires understanding the relationships between them. Semantic relation extraction links key concepts, forming edges in the map that express topical flow and conceptual connections.

Prior work by Sornlertlamvanich and Kruengkrai explored semantic relation extraction within a structured cultural database, combining named entity recognition with rule-based templates to generate tuples like (subject, relation, object). These tuples were used to build knowledge maps representing structured information such as infoboxes or cultural metadata. Their approach achieved high accuracy in a limited, well-defined domain by applying domain-specific templates and constraining relation extraction with entity types. [8]

Although our goal of extracting meaningful semantic relationships is similar, MindTrack uses a different approach tailored for open-domain, unstructured content like conversations and articles. Instead of predefined templates or entity types, we identify relationships between keyphrases by analyzing attention maps. We identify connections by seeing which relation phrases get the most attention from each keyphrase. These attention-based links form the foundation of our network of relationships, enabling a flexible, unsupervised approach without relying on domain-specific rules or manual labeling.

III. MindTrack Algorithm

The overall process begins by applying a refined version of SAMRank to extract the top-k keyphrases from each sentence. These keyphrases are then used to identify potential source keyphrases and analyze their semantic relationships with other keyphrases. For each sentence, semantic links are inferred based on attention analysis. Finally, all relationship data across sentences are aggregated to construct a global semantic structure.

A. Candidate Generation

Our method uses two types of candidates: *keyphrases* and *relation phrases*.

Both types are extracted using the same tool, a `regexParser`, but with different grammar rules. For keyphrases, the parser uses patterns designed to identify noun phrases and other informative content units. For relation phrases, the grammar is tailored to capture typical relational expressions, including verb phrases and prepositional phrases.

By using the same parser framework with tailored grammars, we efficiently extract both types of candidates needed for downstream semantic relation analysis.

B. Self-Attention Map Extraction

We employ a GPT-based model for attention map extraction because of its causal attention mechanism. This architecture permits processing the entire context simultaneously without recalculating prior attention scores as new sentences or tokens are introduced. The causal nature ensures that each token attends only to itself and the tokens before it, preserving consistency in attention values across the sequence.

For each sentence of interest, we extract the attention map from the beginning of the context up to the end of that sentence. This design reflects the fact that, during a conversation, we do not have access to future utterances, only the dialogue history. Since the past is essential for understanding meaning and tracking semantic flow, including the full preceding context helps capture more accurate and relevant attention patterns.

However, GPT models often exhibit a positional bias, assigning disproportionately high attention to the first token in the input sequence. As a result, keyphrases containing the first token often receive inflated scores and consistently rank at the top. SAMRank mitigates this by setting one of the first token's scores to zero. While this suppresses the bias, it also penalizes keyphrases that legitimately include the first token, causing their scores to drop unfairly.

To address this bias, we prepend a padding token (".") to the beginning of the context. This simple adjustment ensures that the first actual keyphrase appears after the artificial padding, allowing it to compete more fairly with others without being affected by the model's positional bias.

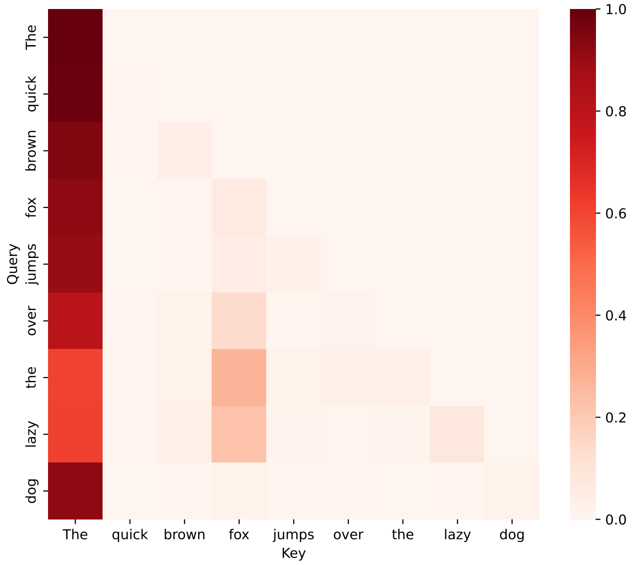


Fig. 1: GPT-2 Attention Heatmap

In Figure 1, we can see that almost every token pays very high attention to the first token in the input, demonstrating the positional bias of GPT-2 [9].

In contrast, Figure 2 shows the effect of prepending a padding token (".") at the start: the first actual token now

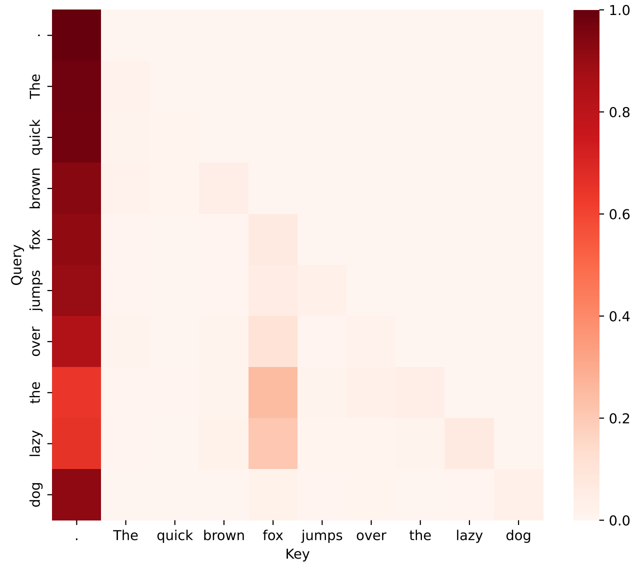


Fig. 2: GPT-2 Attention Heatmap (with padding)

appears after the padding, and we ignore the padding when calculating the final score. This adjustment reduces the undue influence of the first token, allowing attention scores to be distributed more evenly among tokens and keyphrases.

C. Extracting Top-k keyphrase

For each attention map corresponding to a sentence of interest, we compute token-level scores using a refined version of the SAMRank scoring strategy. This involves calculating two components: the global attention score and the proportional score based on backward redistribution. In this step, the attention received by a token is redistributed proportionally to the tokens it attends to, effectively tracing influence backward through the attention flow.

1) Global Attention Score:

$$G_{t_i} = \sum_{j=1}^{n} A_{ji}$$

where A_{ji} is the attention weight from token j to token i, and n is the number of tokens in the document. This represents the total attention received by each token.

2) Proportional Score:

$$P_{t_i} = \sum_{j=1}^{n} B'_{ij}, \quad \text{where} \quad B = A \cdot \mathrm{diag}(G), \quad B'_{ji} = \frac{B_{ji}}{\sum_{j=1}^{n} B_{ji}}$$

where B represents redistributed attention weighted by the global importance scores, and B' is its normalized version ensuring each column sums to one.

3) Phrase-level Score:

$$S_{t_i} = G_{t_i} + P_{t_i}$$

The score at the token-level is determined by the sum of the global attention score G_{t_i} and the proportional attention score P_{t_i} of each token.

$$S_{P_k} = \sum_{i \in P_k} S_{t_i}$$

The score at the phrase-level is calculated as the sum of the final importance scores of the tokens that make up the phrase P_k.

$$S_P = \sum_{k \in P} S_{P_k}$$

The final score at the phrase-level is calculated as the sum of the phrase scores at each location where the phrase appears in the document.

We then focus only on the tokens within the sentence of interest, compute scores for all candidate keyphrases based on their constituent token scores, and select the top-k keyphrases with the highest scores for further processing.

D. Source and Relationship Extraction

Inspired by the global attention score in SAMRank, we compute attention scores for relationship extraction using only the top-k keyphrases identified in the previous step. Specifically, for each keyphrase, we extract its outgoing attention to other tokens within the same sentence's attention map. This allows us to focus on how each keyphrase distributes attention, rather than computing scores from all tokens.

Given a set $K \subseteq \{1, \ldots, N\} \times \{1, \ldots, N\}$ containing tuples (j, n) that represent the start and end token indices of keyphrases (which may appear multiple times in the sentence), we define the global attention score S_i for token i as:

$$S_{t_i} = \sum_{(j,n) \in K} \sum_{k=j}^{n} A_{ki}$$

where A_{ji} is the attention weight from token j to token i.

We then compute phrase-level scores in two separate ways, depending on the type of phrase being evaluated:

- *Source phrase scoring*: For each candidate source phrase P_s^{src}, we calculate the score by summing the token-level scores of all tokens in the phrase:

$$S_s^{\text{src}} = \sum_{i \in P_s^{\text{src}}} S_{t_i}$$

- *Relation phrase scoring*: Similarly, for each candidate relation phrase P_r^{rel}, the phrase score is:

$$S_r^{\text{rel}} = \sum_{i \in P_r^{\text{rel}}} S_{t_i}$$

These two sets of scores are used independently: the source phrase is selected by choosing the phrase with the highest score from the source phrase scoring step, while the relation phrase is selected by choosing the relation phrase with the highest score from the relation phrase scoring step.

Before choosing the highest-scoring phrase, we first sum the scores of all phrases that have the same text, following the same aggregation process as in SAMRank. For example, if the phrase ”the dog” appears more than once in the sentence, we add the scores of every occurrence. The final scores are calculated as:

$$S_p^{\text{src}} = \sum_{s \in P} S_s^{\text{src}}$$

$$S_p^{\text{rel}} = \sum_{r \in P} S_r^{\text{rel}}$$

where P is the set of all phrases with identical text in each category (source or relation).

E. Mind Map Assembly

We start from the latest sentence and go backward to the first sentence. For each sentence, we find the source and relation of each keyphrase. If the source keyphrase does not appear in the top-k keyphrases of the sentence it belongs to, we add it so that we can continue tracking its source and relation.

By doing this recursively, we can get the full source and relationship chain of each keyphrase. The final result forms a tree structure.

IV. Experiment and Evaluation

A. Comparison with refined SAMRank

We compare the performance of the original SAMRank scoring with our refined version, which includes context padding. In the original setup, the first token in the input sequence tends to receive disproportionately high attention scores due to the nature of the GPT attention mechanism. SAMRank addresses this by manually setting one of the scores of the first token to zero. However, this approach can unfairly lower the score of keyphrases that actually include the first token.

Our modification circumvents this issue by adding a padding token (”.”) at the start of the input sequence. This shifts the actual content away from the first position, allowing keyphrases that appear early in the sentence to compete fairly without artificial penalty.

To evaluate our method, we use the same datasets as in the original SAMRank work: Inspec [10], SemEval-2010 [11], and SemEval-2017 [12]. Table I summarizes the statistics of these datasets.

TABLE I: Statistics of datasets

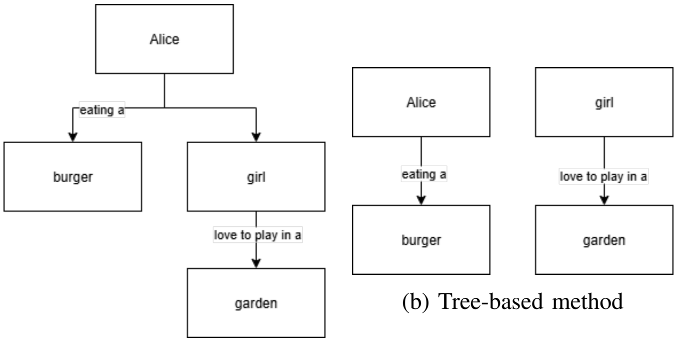
Statistic	**Inspec**	**SemEval2010**	**SemEval2017**
Docs	500	100	493
Avg. Words	135	1589	194
Avg. Sents	6	68	7
Avg. Keys	9	12	17
Unigram (%)	13.47	19.52	24.59
Bigram (%)	52.66	54.57	33.61
Trigram (%)	24.86	19.02	17.40

We evaluate performance by extracting the top keyphrases from the input context, measuring the effectiveness of our method, comparing the unpadded (original) and padded (modified) versions to highlight the impact of this change. Table II presents the performance comparison between the Baseline and With Padding approaches across different datasets.

TABLE II: Performance comparison between Baseline and With Padding on different datasets

Dataset	**F1@5**	**F1@10**	**F1@15**
Baseline			
Inspec	34.42	39.83	39.88
Semeval-2010	16.42	19.58	18.93
Semeval-2017	24.86	34.80	38.80
With Padding			
Inspec	34.58	39.66	39.92
Semeval-2010	16.49	19.40	19.16
Semeval-2017	24.84	34.77	38.76

B. Comparison with Other Approaches (Visualization)



(a) Our method

(b) Tree-based method

Fig. 3: Comparison on: ”Alice is eating a burger. This girl loves to play in the garden.”

Additional topic maps extracted from other example sentences using our method are provided in Appendix A.

V. Discussion

Traditional tree-based dependency parsers such as Stanza [13] and spaCy [14] are effective at identifying intra-sentence syntactic relations but fall short when it comes to linking concepts across sentence boundaries. In contrast, our method captures inter-sentence keyphrase relationships by leveraging attention scores derived from the model’s internal representations. This enables meaningful links between semantically related phrases across sentences, such as connecting ”Alice” and ”girl” in the input ”Alice is eating a burger. This girl loves to play in the garden.”

In contrast to systems utilizing external knowledge graphs like ConceptNet [15], our method infers semantic relationships directly from attention patterns without requiring additional resources. This characteristic renders the method lightweight and adaptable to domains lacking curated knowledge bases or where such resources are incomplete.

However, this attention-only approach introduces interpretability challenges. The model does not determine whether a keyphrase is contextually grounded or arbitrarily introduced. For example, in the input ”The cat eats fish. The dog eats meat,” it may link ”dog” with ”cat” despite a weak semantic connection. Since the model lacks discourse-level reasoning or syntactic structure, it cannot reliably determine whether two keyphrases are truly related in meaning or simply appear in nearby sentences. This can result in incorrect or unsupported links, especially when keyphrases are introduced without clear contextual connection.

In addition, while the performance differences between the baseline and the padding-based approach are relatively small, we argue that the latter may yield better results in practice. This is because keyphrases in the evaluation datasets tend not to appear at the beginning of sentences, where attention biases can unfairly affect their scores. Padding helps mitigate this issue, enabling early keyphrases to compete more fairly during extraction.

VI. Conclusion and Future Work

Our approach leverages raw attention scores from GPT-based models to extract keyphrases and infer semantic relationships, providing a lightweight and unsupervised alternative to traditional syntactic parsers and knowledge-graph-based systems. Unlike tree-based dependency parsers such as Stanza or spaCy, which are limited to intra-sentence analysis, our method captures inter-sentence relationships, enabling broader semantic linking. It also operates without relying on external resources like ConceptNet, making it adaptable to open-domain or low-resource settings.

Compared to prior work like SAMRank, which uses attention maps for ranking keyphrases, our method extends this idea by identifying connections between keyphrases based solely on internal model signals. This general-purpose strategy does not require labeled data or domain-specific rules, making it easy to replicate and apply across various types of texts.

We also introduced padding-based input formatting to reduce position-based attention bias during keyphrase extraction. Although the performance differences are small, this adjustment may improve fairness in identifying keyphrases that appear as the starting phrase of a sentence.

The approach outlined in this work may be useful in various applications such as document summarization and semantic search, where identifying connections between related concepts across sentences can improve coherence and relevance. It may also support educational tools by highlighting relationships between ideas in reading materials or assisting in the generation of concept-based question-answer pairs.

Future work will involve more extensive quantitative evaluations across diverse domains and datasets. In addition, further exploration of specific attention heads and layers in GPT-based models may reveal components that specialize in functions such as topic clustering or source attribution. Understanding these mechanisms could lead to more interpretable and task-specific systems for semantic analysis.

References

[1] J. Novak and A. Cañas, "The theory underlying concept maps and how to construct them," 01 2006.

[2] J. Xu, Z. Gan, Y. Cheng, and J. Liu, "Discourse-aware neural extractive text summarization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 5021–5031. [Online]. Available: https://aclanthology.org/2020.acl-main.451/

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[4] S. Serrano and N. A. Smith, "Is attention interpretable?" *CoRR*, vol. abs/1906.03731, 2019. [Online]. Available: http://arxiv.org/abs/1906.03731

[5] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[6] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[7] B. Kang and Y. Shin, "SAMRank: Unsupervised keyphrase extraction using self-attention map in BERT and GPT-2," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 10 188–10 201. [Online]. Available: https://aclanthology.org/2023.emnlp-main.630

[8] V. Sornlertlamvanich and C. Kruengkrai, "Effectiveness of keyword and semantic relation extraction for knowledge map generation," 03 2016, pp. 188–199.

[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf, 2019, accessed: 2023-08-22.

[10] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," 06 2003.

[11] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, K. Erk and C. Strapparava, Eds. Uppsala, Sweden: Association for Computational Linguistics, Jul. 2010, pp. 21–26. [Online]. Available: https://aclanthology.org/S10-1004/

[12] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, "Semeval 2017 task 10: Scienceie - extracting keyphrases and relations from scientific publications," *CoRR*, vol. abs/1704.02853, 2017. [Online]. Available: http://arxiv.org/abs/1704.02853

[13] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, "Stanza: A python natural language processing toolkit for many human languages," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, A. Celikyilmaz and T.-H. Wen, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 101–108. [Online]. Available: https://aclanthology.org/2020.acl-demos.14/

[14] C. X. Chu, S. Razniewski, and G. Weikum, "ENTYFI: A system for fine-grained entity typing in fictional texts," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Q. Liu and D. Schlangen, Eds. Online: Association for Computational Linguistics, Oct. 2020, pp. 100–106. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.14/

[15] H. Liu and P. Singh, "Conceptnet—a practical commonsense reasoning tool-kit," *BT technology journal*, vol. 22, 06 2004.

Appendix A
Mindmap Attention Visualization

This appendix illustrates the attention-based mindmaps generated by our method using GPT-2 attention heads.

Our method extracts *Top-k keyphrases* and *relation phrases* from layer 11, head 1, while the *source phrases* come from different heads at the same layer to study their effect on relation extraction.

We use two example sequences to demonstrate this:

1) *”Tom cut a plank. He drilled a hole. He added a screw. He painted the wood. He moved the shelf.”*
2) *”A boy walked through the park. Leaves covered a narrow path. A wallet lay near a bench. The boy picked up the wallet. A card showed a name and number. The boy ran to a guard booth.”*

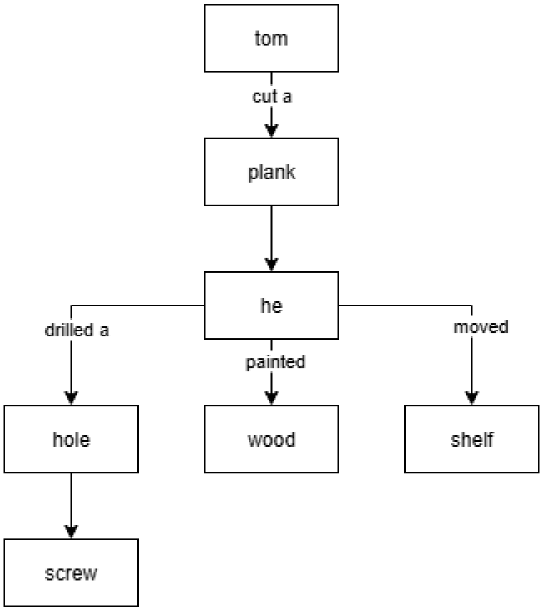


Fig. 4: Mindmap for sequence 1 (Tom story), source phrase extracted from attention at layer 11, head 10.

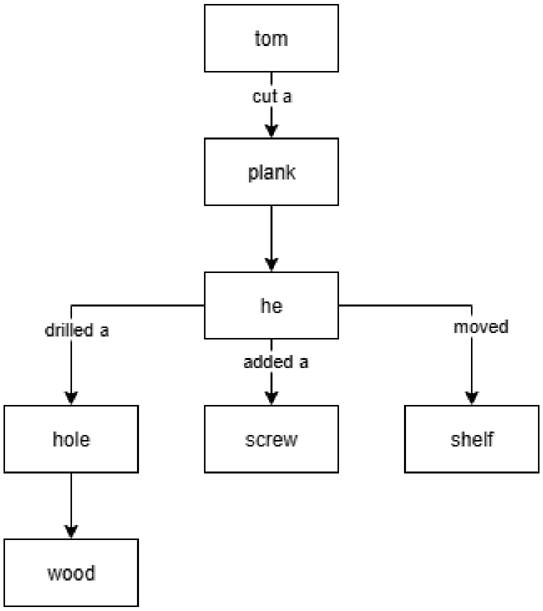


Fig. 5: Mindmap for sequence 1 (Tom story), source phrase extracted from attention at layer 11, head 12.

Figures 4 and 5 show the mindmaps for the first sequence using different heads for source phrase extraction.

Using head 10 (Figure 4), the token ”he” incorrectly attends to ”plank” instead of ”Tom”, and ”screw” points to ”hole” rather than ”he”. When switching to head 12 (Figure 5), the relation between ”screw” and ”he” improves, but ”wood” then incorrectly attends to ”hole”, indicating trade-offs in source phrase selection depending on the head.

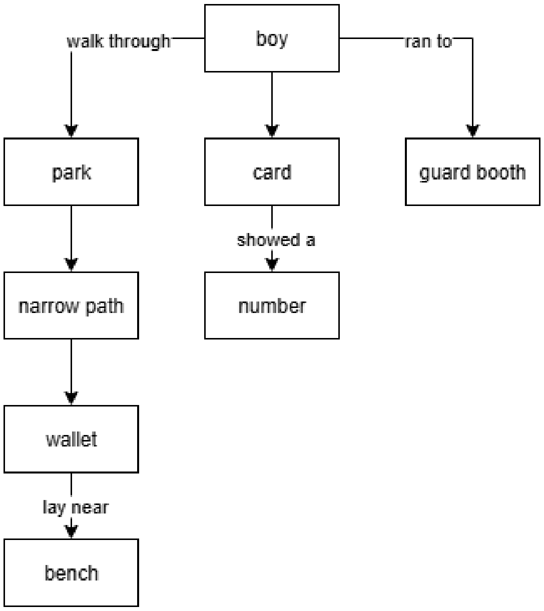


Fig. 6: Mindmap for sequence 2 (Boy story) using source from layer 11, head 10.

Figure 6 shows the mindmap for the second sequence with source phrase extraction from head 10. This example demonstrates more consistent attention alignment, likely due to clearer referents and sentence structure.