Holmes-Graph: A Framework for explainability of the linguistic competence using graph-based text embeddings

Anonymous ACL submission

Abstract

We present Holmes-Graph, a framework to evaluate the linguistic competences of graph-based text embeddings generated by training graph neural networks (GNNs) on graph representations of text. The framework extends the Holmes benchmark, which previously covered only sparse, static, and transformer-based representations. The framework is flexible to be used with any graph representation and GNN architecture. Once weights are learned on a downstream task, frozen embeddings are probed across five linguistic competences. When applied to AI-generated text detection, Integrated Syntactic Graphs (ISG) combined with Graph Neural Networks (GNN) match or exceed DeBERTa-base on syntax, morphology, and semantics, but not for reasoning or discourse. The results confirm that graph-based text embeddings encode non-trivial linguistic knowledge and motivate their inclusion in future probing benchmarks.

1 Introduction

Understanding what linguistic knowledge is encoded in learned text representations has been a central question in Natural Language Processing (NLP) (López-Otal et al., 2025). Early work on language models (LMs) has shown that embeddings implicitly capture a wide range of linguistic competences (Tenney et al., 2019).

The Holmes benchmark formalizes evaluation across five linguistic phenomena and enables a systematic comparison across LM-based representations. Waldis et al. (2024) claim to cover *any* model producing text representations, from sparse models such as bag-of-words (Harris, 1954), to static embeddings such as GloVe (Pennington et al., 2014), to contextualized transformers (Devlin et al., 2019). Despite their wide coverage, Holmes ignores a large class of text representations: graph-based text representations.

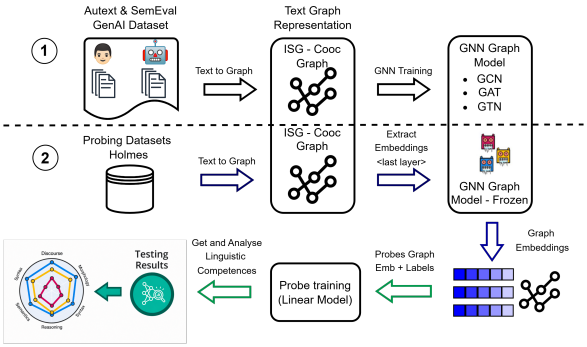

Figure 1: Holmes-Graph framework, an extension of the Holmes benchmark. Text documents are converted into graph representations, processed using different GNN architectures, and transformed into graph embeddings (1). These embeddings are then evaluated with Holmes probes to analyze the linguistic competences captured by the graph-based representations (2).

Graph-based text representations offer a structurally different way of encoding written language (Wang et al., 2024a). Rather than producing token sequence embeddings, Graph Neural Networks (GNN) work on relational structures and propagate information through explicit semantic or grammatical edges. This makes GNNs a promising alternative for tasks where relational structure matters. At the same time, it raises the question: *what linguistic competences do graph-based text embeddings capture, and how does this compare to LM-based embeddings?*

In this work, we extend Holmes to graph-based text embeddings. Figure 1 shows the framework overview. Given a text graph representation and a GNN architecture, we generate document-level embeddings and evaluate them on Holmes, producing a linguistic competence profile. These profiles enable comparison across algorithmic design choices such as graph representation, GNN architecture, node initialization, or training data. We also compare these profiles to transformer-based representations. We demonstrate the framework on

AI-generated text detection, a task where syntactic cues are expected to be meaningful (B. Varga and Baksa, 2025).

We address the following research questions:

- **RQ1:** To what extent do graph-based text embeddings capture the five linguistic competences measured by Holmes?
- **RQ2:** Which algorithmic choices most influence the resulting linguistic capabilities?
- **RQ3:** How do graph-based text embeddings compare to transformer-based text embeddings across linguistic competences?

2 Related work

This section reviews prior work related to the evaluation of linguistic competence in language representations, the application of the Holmes benchmark, graph-based text embeddings, and explainability methods for Graph Neural Networks. Together, these research areas provide the foundation for our proposed *Holmes-Graph* framework.

2.1 Measuring Linguistic Competence

Warstadt et al. (2020) introduce BLiMP, a benchmark that uses minimal-pair diagnostics to test whether LMs assign higher probability to grammatical over ungrammatical sentences across a wide range of core linguistic phenomena. Gauthier et al. (2020) introduce SyntaxGym, which uses psycholinguistically inspired templates to test syntactic generalization in LMs.

Holmes includes more than 200 datasets covering five linguistic phenomena: morphology, syntax, semantics, reasoning, and discourse (Waldis et al., 2024). Frozen pre-trained language model (PLM) embeddings are used as input for classifier-based probing for linguistically related tasks, thus measuring the linguistic competence of LMs.

Complementary to Holmes, Merlo et al. (2023) propose Blackbird Language Matrices (BLMs), a structured multiple-choice framework for probing linguistic competence at multiple levels of representation. BLMs allow finer-grained analysis of whether models detect linguistic objects, use structural regularities, and make linguistic or reasoning errors.

None of these benchmarks includes evaluations on graph-based text representations.

2.2 Applications of Holmes

Holmes has been applied to gain insights into task-specific representations. Alshomary et al. (2025) use Holmes to analyze layer-wise representations of pre-trained transformers for authorship attribution, finding that different transformer layers specialize in distinct linguistic competences and that leveraging all layers together improves out-of-domain generalization.

Nastase and Merlo (2025) shows that cosine similarity between sentence embeddings is not predictive of their relative performance on linguistic tasks. Oota et al. (2026) shows that compression degrades probing performance on morphology, syntax, and discourse while leaving brain activity predictability remains largely unaffected.

These applications demonstrate the versatility of Holmes across representation types and tasks, yet none extend it to graph-based text representations.

2.3 Graph-based Text Embeddings

Graph Neural Networks (GNN) have been widely applied to various text classification tasks (Wang et al., 2024a; Haider Rizvi et al., 2025). Bugueño and De Melo (2023) provides a comparison of different graph construction strategies and GNN architectures across diverse datasets, including sentiment analysis, topic classification, and hyperpartisan news detection. Their results suggest that simple graph representations are especially useful for longer documents, where they can capture local and global textual relations. The current best-performing approach, CoGraphNet (Li et al., 2025), models both words and sentences as nodes. Thus, each document is represented through heterogeneous word–sentence graphs. The model reports improved classification performance over several baselines.

Despite these advancements, understanding what linguistic information is encoded in these learned representations remains an open research question.

2.4 Explainability of Graph-based text Embeddings

Recent work on GNN explainability has focused on identifying the graph components that drive a model's predictions. A foundational example is GNNExplainer (Ying et al., 2019), which generates post-hoc explanations by learning compact subgraphs and node features that are most informa-

tive for a GNN's prediction. Building on this line of work, Agarwal et al. (2023) introduces GraphXAI and ShapeGGen for evaluating GNNs using synthetic graphs with known ground-truth explanations and metrics such as faithfulness, stability, fairness, and explanation accuracy. CoGraphNet (Li et al., 2025) incorporates positional bias weights to inspect how words and sentences are connected and which textual units contribute to the classification.

These approaches are concerned with recovering the nodes, edges, or features responsible for a decision. In contrast, our work addresses a complementary question: whether graph-based text representations encode meaningful linguistic knowledge.

3 Methodology

Holmes-Graph extends Holmes to graph-based text embeddings (see Figure 1). Given a collection of text documents, graph representations are first constructed using either Integrated Syntactic Graphs or Co-Occurrence Graphs. Node features are initialized with contextual embeddings extracted from a PLM. These graph representations are then processed by different Graph Neural Network architectures, such as GCN, GAT, and GTN, trained on a downstream task while keeping the PLM weights frozen. After training, the GNN parameters are also frozen, and document-level embeddings are extracted from the final graph representation layer. Finally, these embeddings are evaluated using the HOLMES probing framework, where linear probes are trained to measure the extent to which the graph-based embeddings encode linguistic competencies across syntax, semantics, reasoning, morphology, and discourse.

Node feature extraction. Each node is initialized with a contextual embedding from a PLM. The full document text is passed through the PLM, and each token receives its contextual representation from the final hidden layer. When a word is split into multiple subword tokens, the mean of its subword embeddings is used as the node representation. PLM weights are frozen throughout GNN training, which isolates the contribution of graph structure and GNN architecture from any PLM adaptation.

Document-level embedding extraction. To obtain a single embedding for each document, the node representations produced by the final GNN layer are aggregated into a graph-level representation. Specifically, given the set of node embeddings in a document graph, we apply a readout function to summarize the graph into a fixed vector size. We experiment with two aggregation strategies: mean pooling, which computes the average of all node embeddings, and attention-based pooling, which learns the relative importance of nodes and assigns higher weights to more informative regions of the graph. The resulting graph-level embedding is then used as the document representation for downstream evaluation and linguistic probing.

Probing with Holmes. Document-level embeddings are evaluated following the Holmes probing setup (Waldis et al., 2024) without modification, which allows direct comparison between graph-based and transformer-based representations.

Framework and code available [1] at `http://holmes-graph-framework.com`.

4 Experimental Settings

We test the framework on the task of AI-generated text identification. Still, generating graph-based text embeddings involves several algorithmic design choices: graph representation, node initialization, GNN architecture, and training dataset. In the next subsections, we describe each of them and the range of choices to be compared.

4.1 Graph Representations

We explore two graph representations: the Co-Occurrence Graph (CoOc), focusing on semantic relationships based on word proximity, and the Integrated Syntactic Graph (ISG), incorporating lexical, morphological, and syntactic information derived from dependency parsing.

A **Co-Occurrence Graph** (CoOc) represents words as nodes and connects words if they co-occur within a context window. Each edge is weighted using the Pointwise Mutual Information (PMI), which quantifies how strongly two words are semantically associated[2]. This representation preserves document-level relationships by constructing one graph per document. Figure 2 (left) illustrates the structure of a co-occurrence graph for the following example sentence: *AI transforms industries through automation and data*. Words such as *AI*,

[1] https://anonymous.4open.science/r/Holmes-GNN-F2A6

[2] PMI compares the joint probability of two words co-occurring with the probability of them occurring independently.

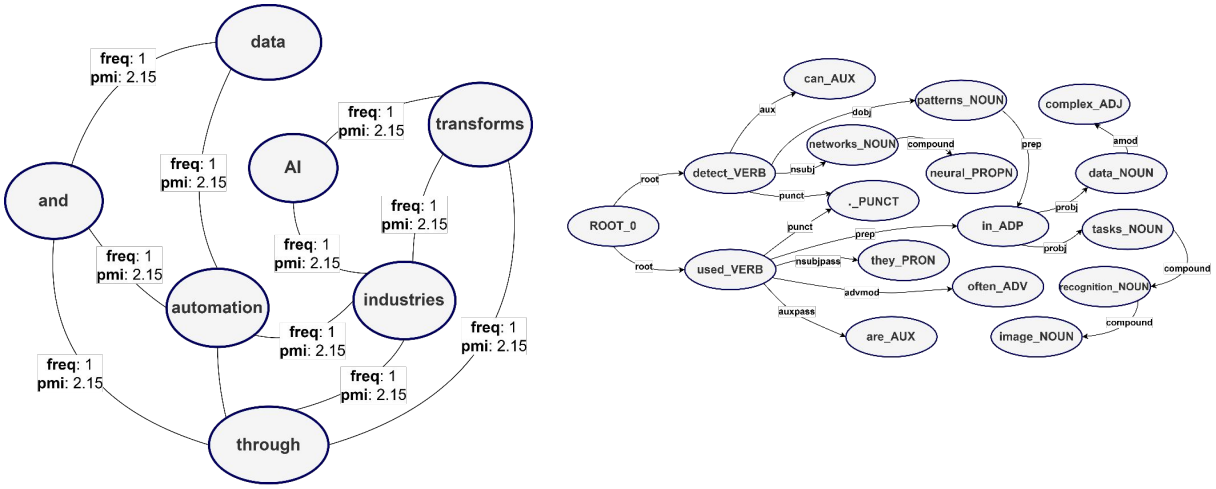

Figure 2: Example of graph representations. Left: Co-Occurrence Graph. Right: Integrated Syntactic Graph.

data, and *automation* frequently co-occur, resulting in stronger edge connections.

The **Integrated Syntactic Graph** (ISG) combines lexical, morphological, and syntactic information into a unified structure, thus capturing multi-level linguistic information (Gómez-Adorno et al., 2016). For each sentence, a dependency tree is extracted using a parser such as spaCy (Honnibal et al., 2020). Each token becomes a node in token_POS format, and edges represent dependency relations (e.g., nsubj, dobj). Sentence-level graphs are merged into a single document graph with a virtual root node (ROOT-0) from which all graph connections are derived. Figure 2 (right) shows an ISG graph for the text document: *Neural networks can detect patterns in complex data. They are often used in image recognition tasks.*

4.2 Node Initialization

Each node in the graph is initialized with a contextual embedding obtained from a PLM. We experiment with three PLMs for node initialization: **BERT-base** (Devlin et al., 2019), **RoBERTa-base** (Liu et al., 2019), and **DeBERTa-base** (He et al., 2020). We also include a more recent Large Language Model (LLM), **Qwen25_05b_Instruct** (Team, 2024), in our comparison. Either words (CoOc) or token-POS pairs (ISG) embeddings are extracted. When a word is split into multiple subword tokens, the mean of its subword embeddings is used.

4.3 Graph Neural Network Architecture

Several GNN architectures have been proposed for learning representations over graph structures, including: *Graph Convolutional Networks* (GCN), *Graph Attention Networks* (GAT), and *Graph Transformer Networks* (GTN). Each architecture introduces different strategies for neighborhood aggregation and relational learning within graph structures.

Graph Convolutional Networks (GCN) (Kipf and Welling, 2017) extend the concept of convolution from CNNs to graph-structured data. GCNs learn node representations by aggregating information from neighboring nodes through normalized weighted averaging. This approach enables efficient local feature propagation and has been widely applied to node and graph classification tasks. However, a limitation of GCNs is that all neighboring nodes contribute with similar importance during aggregation, which may reduce their ability to distinguish more relevant connections.

Graph Attention Networks (GAT) (Veličković et al., 2018) incorporate attention mechanisms into the aggregation process, allowing the model to learn the relative importance of neighboring nodes. Instead of uniformly averaging neighbor representations, GATs assign adaptive attention weights to each connection, enabling more expressive relational modeling. In addition, multi-head attention improves training stability and captures diverse interaction patterns. This architecture is particularly effective in text graphs, where relationships between words or documents may have different semantic relevance.

Graph Transformer Networks (GTN) (Yun et al., 2019) are inspired by the Transformer architecture and extend self-attention mechanisms to graph domains. Unlike GCNs and GATs, which

Dataset	Train		Validation		Test		# Docs
	human	*machine*	*human*	*machine*	*human*	*machine*	
Autext2023	11,963	11,728	5,083	5,071	10,642	11,190	55,677
SemEval2024	58,416	56,350	2,500	2,500	17,130	17,136	159,029

Table 1: Summary stats for the datasets used in the experiments. All text documents are in English.

mainly focus on local neighborhoods, GTNs can capture long-range and global dependencies between nodes. These models integrate node and edge features directly into the attention computation, allowing the representation of semantic, syntactic, or statistical relationships within the graph. GTNs also incorporate residual connections and structural encodings, providing greater expressive power for graph-based learning.

4.4 Training Datasets

To train the GNNs and obtain the graph-based text embeddings, we employed the Autextification2023 and SemEval2024 datasets, which are corpora designed for evaluating AI-generated text detection methods. All of these datasets contain documents from multiple domains (e.g., Wikipedia, Reddit, Reviews, etc) and multiple LLMs. Table 1 provides a detailed distribution of instances across training, validation, and test sets, along with the respective average document lengths.

Autextification2023. This dataset contains human and LLM-generated texts from five domains: tweets, reviews, how-to articles, news, and legal documents. Human texts were sourced from datasets like Amazon Reviews and WikiLingua, while machine-generated texts were created using BLOOM and GPT-3 for their multilingual capabilities (Sarvazyan et al., 2023).

SemEval2024. It is an extension of the M4 dataset (Wang et al., 2024c); this benchmark contains machine-generated text across multiple languages, including English, Spanish, and Chinese. It features texts generated by various LLMs, such as GPT-4, Cohere, and Dolly-v2 (Wang et al., 2024b).

5 Evaluation

This section describes the evaluation setup used to analyze the linguistic competence of graph-based text embeddings. We first summarize the five linguistic competencies defined in Holmes, followed by the experimental configurations designed to evaluate the impact of graph representation, GNN architecture, and node initialization choices.

5.1 Linguistic competences

Holmes organizes its probing tasks into five linguistic competences (Waldis et al., 2024). To evaluate the linguistic competence of graph-based embeddings, we use the original Holmes benchmark, which comprises more than 200 probing datasets designed to assess morphology, syntax, semantics, reasoning, and discourse phenomena.

Each competence is evaluated using a set of probing tasks. The reported task score depends on the type of probing task: we use macro-F1 for classification tasks and Pearson correlation for regression tasks. Competence-level scores are then computed as the arithmetic mean of the task scores assigned to each linguistic competence. Therefore, these values should be interpreted as aggregate probing scores rather than as a single homogeneous metric such as accuracy. Higher scores indicate that the embedding encodes more information relevant to the corresponding linguistic competence. Table 2 shows example probes for each competence.

Morphology probes target word-level grammatical features such as part-of-speech tags, number, gender, tense, and verb form (Haspelmath and Sims, 2013). These features are typically local and recoverable without sentence-level context. *Syntax* probes evaluate grammatical structure at the sentence level, including dependency relations, subject-verb agreement, and argument structure (Chomsky, 2002). *Semantics* probes cover meaning-related phenomena such as semantic role labeling, word sense, and coreference resolution (Palmer et al., 2013). *Reasoning* probes test the capacity to draw linguistic inferences, including negation detection, quantifier scope, and natural language inference. *Discourse* probes evaluate coherence and cross-sentence relations, including discourse relation classification and sentence ordering (Mann et al., 1992).

5.2 Experiments

To answer the research questions introduced in Section 1, we conduct a series of experiments analyzing the linguistic competence of graph-

Table 2: Example probes for each linguistic competency

Competency	Example Probe	Example Sentence	What it Evaluates
Syntax	zorro-agreement_subject_verb	"The keys to the cabinet are missing."	Subject–verb agreement across intervening phrases.
Semantics	protoroles-volition	"Maria deliberately opened the window."	Whether the subject acts intentionally or voluntarily.
Reasoning	SemAntoNeg	"The task is not easy."	Understanding semantic negation and implicit meaning.
Morphology	zorro-irregular-verb	"She went to school yesterday."	Recognition of irregular verb forms.
Discourse	coref	"Laura called Ana because she needed help."	Coreference resolution based on context.

based text embeddings under different settings. Specifically, we explore two graph representations (ISG and CoOc), three GNN architectures (GCN, GAT, and GTN), and four initialization models (BERT-base, RoBERTa-base, DeBERTa-base, and Qwen25_05b_Instruct).

The experiments are organized around four main analyses. First, we evaluate to what extent graph-based text embeddings capture meaningful linguistic knowledge across the five Holmes competencies. Second, we analyze the impact of graph construction by comparing ISG and CoOc representations. Third, we study the effect of different GNN architectures on the resulting linguistic competence profiles. Finally, we compare graph-based embeddings against transformer-based embeddings to identify strengths and weaknesses across linguistic phenomena. Implementation details can be found in Appendix A.1. The results of these experiments are presented and discussed in Section 6.

6 Results

This section presents the experimental results obtained with the proposed *Holmes-Graph* framework. We first analyze the linguistic competence profile of graph-based embeddings, followed by the impact of graph representation and GNN architecture choices. Finally, we compare graph-based embeddings against transformer-based representations to identify the linguistic phenomena where each representation is most effective.[3]

6.1 Linguistic Competence of Graph-Based Embeddings

Figure 3 presents the linguistic competence profile of ISG-GTN embeddings according to the HOLMES probing score. Overall, the model obtains an aggregate score of 0.662, covering a good amount of linguistic information. The strongest competencies are Syntax and Morphology. This confirms our claim that ISG-GNN captures structural and form-related linguistic patterns particularly well. Performance is more moderate for Semantics (0.602) and Reasoning (0.587), while Discourse shows the lowest score (0.399), which is expected because our text graph representations do not model discourse-level features. The error bars (standard deviation) indicate variability across probes within each competency, with wider intervals for semantics, reasoning, and discourse, reflecting greater heterogeneity in the model's performance across these linguistic dimensions.

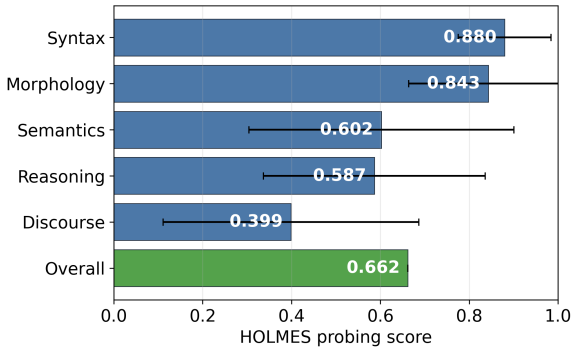

Figure 3: Linguistic competence profile of ISG-GTN embeddings.

6.2 Impact of Graph Representation: ISG vs. CoOc

Figure 4 compares scores obtained with the ISG and CoOc graph representations across the five linguistic competencies. ISG outperforms CoOc across all categories, achieving a higher overall score of 0.648 compared with 0.591 for CoOc. The largest gap appears in Semantics, where ISG reaches 0.565 while CoOc obtains 0.409, which means that ISG captures semantic information more effectively than CoOc. In general, the ISG representation provides a more informative graph structure for encoding linguistic knowledge, particularly for semantic and syntactic competencies.

6.3 Impact of the GNN Architecture

Figure 5 compares scores obtained across the three evaluated GNN architectures: GCN, GAT, and

[3]Unless otherwise specified, *ISG-GTN* refers to the configuration using the Integrated Syntactic Graph (ISG), a Graph Transformer Network (GTN) architecture, and DeBERTa-base embeddings for node feature initialization.

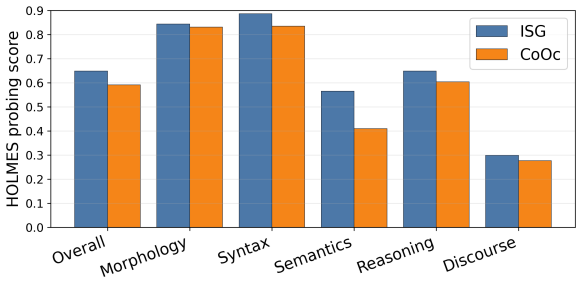

Figure 4: Graph representation: ISG vs. CoOc

GTN (using ISG representation with DeBERTa-base embeddings as node feature initialization). While the overall performance differences are relatively moderate, consistent patterns can be observed across specific linguistic competencies. In general, GTN achieves the strongest overall performance, particularly in syntax, semantics, and discourse, suggesting that transformer-based graph architectures better capture local and long-range dependencies within text graphs.

GAT outperforms GCN in most competencies, indicating that attention mechanisms help identify more informative neighborhood relations during message passing. The largest gains across architectures appear in syntax and semantics, highlighting the importance of graph structure and relational aggregation for modeling grammatical and semantic information. Reasoning and discourse remain challenging competencies for all architectures.

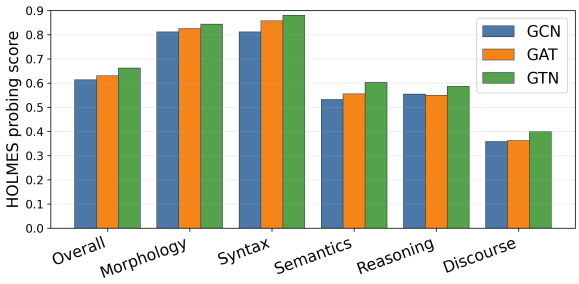

Figure 5: GNN architecture comparison on ISG graphs: GCN vs. GAT vs. GTN

6.4 Comparison with Transformer-Based Embeddings

Table 3 shows that ISG-GTN embeddings achieve overall performance comparable to transformer-based embeddings, with small but steady gains across most encoder families and a larger improvement for Qwen. Models marked with *-ft* correspond to PLMs fine-tuned on the AI-generated text detection task using the Autext2023 and SemEval2024 datasets. For ISG-GTN, node features are initialized using embeddings from the same PLM used in the corresponding comparison. However, aggregate scores alone conceal important differences at the linguistic competence level. Figure 6 highlights these variations by visualizing the score gap between ISG-GTN and transformer-based embeddings across the five Holmes competencies.

The largest gains appear in syntax and semantics. ISG-GTN improves syntactic competence for most encoder families, with particularly large gains for Qwen, RoBERTa-ft, BERT, and BERT-ft. The gains for Qwen are especially large because, unlike encoder-only PLMs, instruction-tuned generative models are not optimized for syntactic structure. This pattern follows from the design of the ISG representation, which explicitly encodes dependency relations and enables the GNN to propagate information through structured linguistic connections. Semantic competence also improves in most settings, suggesting that graph-based aggregation enriches meaning-related information beyond the original transformer representation.

In contrast, gains in reasoning and discourse are less uniform across encoder families. Transformer-based embeddings often retain an advantage in reasoning, and similarly outperform on discourse. Overall, these findings suggest that graph-based embeddings should not be interpreted as a direct replacement for transformer embeddings. Instead, they encode complementary linguistic signals, with their strongest contribution concentrated in syntax- and structure-oriented competencies.

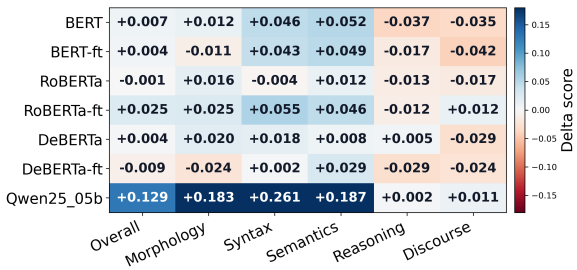

Figure 6: Competence-Level Performance Gaps Between ISG-GTN and LLM Embeddings

7 Discussion

Our results show that graph-based text embeddings are a valuable addition to the language model topology defined in Holmes. Across the five linguistic competences, ISG-GTN achieves performance comparable to strong transformer-based embed-

Table 3: Comparison between transformer-based embeddings (LLM) and graph-based embeddings (ISG-GTN) across linguistic competencies on the HOLMES benchmark. Models with the suffix "-ft" were fine-tuned on the Autext2023 and SemEval2024 datasets before evaluation. For ISG-GTN, node features are initialized using embeddings from the corresponding PLM shown in each comparison.

Model	Overall		Morphology		Syntax		Semantics		Reasoning		Discourse	
	LLM	ISG-GTN	LLM	ISG-GTN	LLM	ISG-GTN	LLM	ISG-GTN	LLM	ISG-GTN	LLM	ISG-GTN
BERT	0.600	**0.608**	0.709	**0.721**	0.775	**0.821**	0.542	**0.594**	**0.530**	0.493	**0.445**	0.410
BERT-ft	0.586	**0.590**	**0.706**	0.695	0.731	**0.774**	0.540	**0.589**	**0.521**	0.504	**0.430**	0.388
RoBERTa	**0.608**	0.607	0.715	**0.730**	**0.814**	0.809	0.562	**0.574**	**0.522**	0.509	**0.429**	0.412
RoBERTa-ft	0.554	**0.580**	0.641	**0.666**	0.688	**0.743**	0.531	**0.577**	**0.517**	0.505	0.395	**0.407**
DeBERTa	0.657	**0.661**	0.823	**0.843**	0.862	**0.880**	0.591	**0.599**	0.574	**0.579**	**0.434**	0.405
DeBERTa-ft	**0.636**	0.626	**0.808**	0.784	0.828	**0.830**	0.546	**0.575**	**0.569**	0.540	**0.426**	0.402
Qwen25_05b	0.443	**0.572**	0.480	**0.663**	0.490	**0.751**	0.376	**0.563**	0.499	**0.501**	0.370	**0.382**

dings such as DeBERTa-base, while exhibiting a distinct competence profile. This demonstrates that graph-based representations encode linguistic knowledge and capture information that is not fully reflected in standard sequence-based embeddings. Extending Holmes through the proposed *Holmes-Graph* framework offers a complementary perspective on how graph-based and transformer-based architectures encode linguistic structure.

The largest improvements appear in syntax and morphology, where ISG-GTN outperforms DeBERTa-base by the largest margins. Unlike transformers, where syntactic structure must be inferred implicitly from token sequences, ISG explicitly incorporates dependency relations as labeled graph edges. During message passing, the GNN propagates information directly through these syntactic connections, enabling the resulting embeddings to preserve grammatical and structural patterns more effectively. These findings suggest that for tasks where syntactic competence is critical, constructing a graph over the syntactic structure of a document may be more effective than fine-tuning a transformer on task-specific data.

Semantics also benefits from the graph-based representation, although the gains are generally smaller than those observed for syntax. The improvements in semantic role and proto-role probes indicate that graph aggregation enriches local relational information beyond the original PLM embeddings. The gap between ISG and CoOc is largest in semantics. CoOc encodes word proximity and co-occurrence strength, whereas ISG dependency edges explicitly represent grammatical relations that carry semantic content, which explains why ISG better preserves meaning-related information.

Reasoning presents a challenge since phenomena such as negation, quantifier scope, and natural language inference require understanding propositional content and scope relationships. These are not directly encoded in syntactic dependency structures, which explains why transformer-based embeddings retain an advantage in this competence.

Graph-based embeddings fall short on discourse. Discourse coherence depends on long-range relationships across sentences that are not well captured by local message-passing over a syntactic graph. This points to a structural limitation of the graph representations we evaluate, and suggests that future work should explore graph constructions that explicitly model inter-sentence and document-level relations (Baez Santamaria et al., 2021).

The choice of GNN architecture also shapes the linguistic competence profile. GTN achieves the strongest performance, particularly in syntax and semantics, suggesting that global self-attention over graph structure is more effective than local neighborhood aggregation for capturing relational dependencies. The advantage of GAT over GCN further confirms that adaptive attention weights improve the model's ability to identify informative linguistic connections during message passing.

8 Conclusion

We presented a framework for evaluating the linguistic competence of graph-based text embeddings, addressing a gap not only in the Holmes benchmark but also in the field of GNN interpretability. We show that graph-based text embeddings are competitive and mostly capture syntax-related linguistic competencies. Furthermore, graph-based text embeddings (when trained on an authorship analysis task) match or exceed DeBERTa on syntax, semantics, and morphology, while falling short on reasoning and discourse. The framework enables future work to trace such differences back to specific design choices in graph construction, architecture, or node initialization.

Limitations

The space of design choices for graph-based text embeddings is large. We explore two graph representations, three GNN architectures, and three node initializers, but many other combinations remain untested. This work is best understood as a proof of concept that demonstrates the framework rather than an exhaustive comparison.

The framework is demonstrated on a single downstream task: AI-generated text detection. Whether the resulting linguistic competence profiles generalize to other tasks or domains is an open question.

The discourse competence gap is a structural limitation of the graph representations we evaluate. Both CoOc and ISG graphs model intra-sentence and local word-level relationships, and neither explicitly encodes inter-sentence or document-level relations. Extending the framework to discourse-aware graph representations is left for future work.

Graph construction and GNN inference are more computationally expensive than transformer-based inference, particularly for long documents where graph size grows with document length.

References

Chirag Agarwal, Owen Queen, Himabindu Lakkaraju, and Marinka Zitnik. 2023. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144.

Milad Alshomary, Nikhil Reddy Varimalla, Vishal Anand, Smaranda Muresan, and Kathleen McKeown. 2025. Layered insights: Generalizable analysis of human authorial style by leveraging all transformer layers. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10279–10292, Suzhou, China. Association for Computational Linguistics.

Erika B. Varga and Attila Baksa. 2025. Syntactic comparison of human and ai-written scientific texts. In *Annales Mathematicae et Informaticae*, volume 61, pages 248–260.

Selene Baez Santamaria, Thomas Baier, Taewoon Kim, Lea Krause, Jaap Kruijt, and Piek Vossen. 2021. EMISSOR: A platform for capturing multimodal interactions as episodic memories and interpretations with situated scenario-based ontological references. In *Proceedings of the 1st Workshop on Multimodal Semantic Representations (MMSR)*, pages 56–77, Groningen, Netherlands (Online). Association for Computational Linguistics.

Margarita Bugueño and Gerard De Melo. 2023. Connecting the dots: What graph-based text representations work best for text classification using graph neural networks? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8943–8960.

Noam Chomsky. 2002. *Syntactic structures*. Walter de Gruyter.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jon Gauthier, Jennifer Hu, Ethan Wilcox, Peng Qian, and Roger Levy. 2020. SyntaxGym: An online platform for targeted evaluation of language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 70–76, Online. Association for Computational Linguistics.

Helena Gómez-Adorno, Grigori Sidorov, David Pinto, Darnes Vilariño, and Alexander Gelbukh. 2016. Automatic authorship detection using textual patterns extracted from integrated syntactic graphs. *Sensors*, 16(9):1374.

Syed Mustafa Haider Rizvi, Ramsha Imran, and Arif Mahmood. 2025. Text classification using graph convolutional networks: A comprehensive survey. *ACM computing surveys*, 57(8):1–38.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Martin Haspelmath and Andrea Sims. 2013. *Understanding morphology*. Routledge.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. In *To appear in Proceedings of the 7th International Conference on Natural Language Processing (LREC)*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Pengyi Li, Xueying Fu, Juntao Chen, and Junyi Hu. 2025. Cographnet for enhanced text classification using word-sentence heterogeneous graph representations and improved interpretability. *Scientific Reports*, 15(1):356.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

Miguel López-Otal, Jorge Gracia, Jordi Bernad, Carlos Bobed, Lucía Pitarch-Ballesteros, and Emma Anglés-Herrero. 2025. Linguistic interpretability of transformer-based language models: a systematic review. *arXiv preprint arXiv:2504.08001*.

William C Mann, Christian MIM Matthiessen, and Sandra A Thompson. 1992. Rhetorical structure theory and text analysis. In *Discourse Description*, page 39. John Benjamins.

Paola Merlo, Chunyang Jiang, Giuseppe Samo, and Vivi Nastase. 2023. Blackbird language matrices tasks for generalization. In *Proceedings of the 1st GenBench Workshop on (Benchmarking) Generalisation in NLP*, pages 163–172, Singapore. Association for Computational Linguistics.

Vivi Nastase and Paola Merlo. 2025. Testing the assumptions about the geometry of sentence embedding spaces: the cosine measure need not apply. *arXiv preprint arXiv:2509.01606*.

Subba Reddy Oota, Vijay Rowtula, Satya Sai Srinath Namburi, Khushbu Pahwa, Anant Khandelwal, Manish Gupta, Tanmoy Chakraborty, and Bapi S Raju. 2026. Linguistic properties and model scale in brain encoding: from small to compressed language models. *arXiv preprint arXiv:2602.07547*.

Martha Palmer, Ivan Titov, and Shumin Wu. 2013. Semantic role labeling. In *NAACL HLT 2013 Tutorial Abstracts*, pages 10–12, Atlanta, Georgia. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Areg Mikael Sarvazyan, José Ángel González, Marc Franco Salvador, Francisco Rangel, Berta Chulvi, and Paolo Rosso. 2023. Overview of autextification at iberlef 2023: Detection and attribution of machine-generated text in multiple domains. In *Procesamiento del Lenguaje Natural*, Jaén, Spain.

Qwen Team. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Andreas Waldis, Yotam Perlitz, Leshem Choshen, Yufang Hou, and Iryna Gurevych. 2024. Holmes a benchmark to assess the linguistic competence of language models. *Transactions of the Association for Computational Linguistics*, 12:1616–1647.

Kunze Wang, Yihao Ding, and Soyeon Caren Han. 2024a. Graph neural networks for text classification: A survey. *Artificial Intelligence Review*, 57(8):190.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, jinyan su, Artem Shelmanov, Akim Tsvigun, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, Chenxi Whitehouse, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024b. Semeval-2024 task 8: Multidomain, multimodel and multilingual machine-generated text detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 2041–2063, Mexico City, Mexico. Association for Computational Linguistics.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, Thomas Arnold, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024c. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1369–1407, St. Julian's, Malta. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32.

Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. *Advances in neural information processing systems*, 32.

A Appendix

A.1 Implementation details.

Experiments were executed on two NVIDIA RTX A5000 GPUs with 24GB of memory each. The implementation used Python 3.10, CUDA 12.2, PyTorch, and PyTorch Geometric 2.5. The default

configuration used two GNN layers, two attention heads for attention-based architectures, 256 hidden dimensions, ReLU activations, and layer normalization. Models were optimized with AdamW using a learning rate of 2×10^{-5}, weight decay of 10^{-3}, batch size 16, and up to 100 epochs with early stopping.