SCENT: SELF-CONSISTENT EPISTEMIC NODE TEXTUALIZER

TECHNICAL DOCUMENTATION

ABSTRACT

Entity disambiguation serves as a foundational capability in Natural Language Processing, enabling the precise resolution of ambiguous mentions to specific, unambiguous entities in a Knowledge Graph. To facilitate robust and structure-aware generation, we propose **Scent**, a constrained generation framework that strictly grounds textual output in structured graph knowledge. Scent leverages an encoder backbone (e.g., RoBERTa) augmented with two specialized, simultaneously trained Low-Rank Adaptation (LoRA) modules: a **Graph-adapter** (f_{graph}), designed to synthesize semantic representations from structured data, and a **Text-adapter** (f_{text}), which manages the integration of linguistic context with these structure-grounded entity slots. By enforcing consistency between textual outputs and graph topology, this framework ensures that generated content remains strictly valid within the defined candidate set.

source code: https://github.com/ra0o0f/scent

1 Illustrative Framework

To demonstrate the Scent mechanism, consider a scenario where the model must disambiguate a specific mention. This generation is **constrained**: the output entity must be selected from a pre-defined candidate set of entities $\mathcal{C}$, ensuring the text is grounded in the Knowledge Graph (KG).

Example Scenario: We aim to resolve the ambiguous mention "Kyoto" to its canonical entity entry within the following context:

> "The breathtaking gardens of **Kyoto** reflect the serene beauty of traditional Japanese culture."

Here, the mention m is "Kyoto" and the target entity n_{target} is **Kyoto**.

1.1 Dual-View Representation

In our framework, the mention and its target entity are processed through two distinct but aligned views.

The Textual View: The textual view presents the ambiguous mention within its linguistic context. We prepare the sequence for prediction by explicitly demarcating the mention span using a `<mention_start>` token, followed immediately by the entity prediction slot. This slot is defined by an entity start marker and a fixed buffer of mask tokens (N). Crucially, to allow the model to dynamically learn entity boundaries within this fixed buffer, the entity label is explicitly terminated by a special `<end_of_title>` token.

$$X_{text} = [\dots \text{of}, \texttt{<mention\_start>}, \text{Kyoto}, \texttt{<entity\_start>}, \underbrace{\texttt{<mask>}, \texttt{<mask>}, \texttt{<mask>}}_{N}, \texttt{<entity\_end>}, \text{reflect}, \dots] \tag{1}$$

The Structural View: Simultaneously, Kyoto exists as a node in the KG. We follow the TokenGT [3] formatting to linearize its local neighborhood (k-hop traversal) into a sequence of node and edge semantic units. For example:

- (Kyoto) $\xrightarrow{\text{located in}}$ (Japan)
- (Kyoto) $\xrightarrow{\text{contains}}$ (Kinkaku-ji)

Crucially, to construct the initial feature representation for each entity, rather than employing a static embedding layer, we generate semantic representations directly from the text. We encode the entity **name** and its corresponding **short abstract** from Wikipedia using the backbone model (e.g., `<s> Kyoto </s> </s> Abstract... </s>`).

Instead of relying solely on the `<s>` token, we adopt the "First-Last-Avg" strategy [7], which computes the element-wise average of the hidden states from the first and last layers of the encoder, followed by mean pooling over the token sequence.

The sequence places the masked target node in its linearized context, as shown in:

$$X_{graph} = [\dots, \mathbf{n}_{Japan}, \mathbf{e}_{located\_in}, \mathbf{n}_{\texttt{<node\_mask>}}, \mathbf{e}_{contains}, \mathbf{n}_{Kinkaku-ji}, \dots] \tag{2}$$

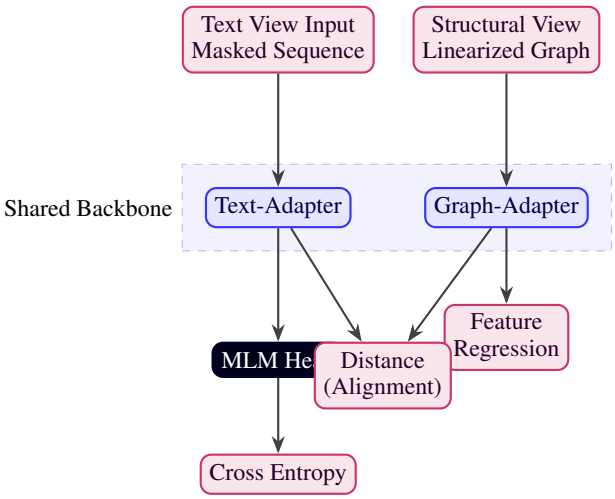

Figure 1: The Scent Architecture. Inputs are processed through shared backbones with specialized LoRA adapters, optimizing for multi-task objectives.

1.2 Training Objectives

Scent employs a multi-task strategy to enforce consistency between the textual output and the underlying graph topology.

1.2.1 Textual Generation

Using the **Text-Adapter**, we perform standard **Masked Language Modeling** on the tokens within the mask buffer. The model must predict the token IDs for "Kyoto" based on the surrounding sentence context.

We minimize the Cross-Entropy loss between the predicted logits and the actual entity label token IDs (Y_{target}) as defined in:

$$\mathcal{L}_{text} = \text{CrossEntropy}\Big(\text{Head}_{MLM}(f_{text}(X_{text})),\ Y_{target}\Big) \tag{3}$$

While the buffer size N is fixed (e.g., 20 masks) to accommodate the longest potential entities, the Cross-Entropy loss is calculated only on the specific subset of mask tokens corresponding to the actual entity length, meaning the remaining padding masks are excluded from the loss update even though they still participate in the self-attention mechanism.

1.2.2 Structure-Aware Feature Regression

Using the **Graph-Adapter**, we apply a **Masked Feature Regression** on the graph sequence. The model must reconstruct the semantic vector of the masked target node solely from its structural context (neighbors and edge relations).

We minimize the distance (e.g., Mean Squared Error or Negative Cosine Similarity) between the predicted vector and the ground truth as follow:

$$\mathcal{L}_{graph} = \text{Distance}\Big(f_{graph}(X_{graph})[\texttt{<node\_mask>}],\ \mathbf{n}_{ref}\Big) \tag{4}$$

1.2.3 Cross-Modal Alignment

To bridge the two views, we enforce a strict consistency constraint on the `<entity_start>` token. The hidden state of `<entity_start>` generated by the Text-Adapter must align with the graph representation of the target node generated by the Graph-Adapter.

To generate the reference representation for the target, we utilize the unmasked version of the graph sequence. Unlike the masked input used for feature regression, we replace the mask token with the specific target entity node while retaining the surrounding linearized context:

$$X^*_{graph} = [\dots, \mathbf{n}_{Japan}, \mathbf{e}_{located\_in}, \mathbf{n}_{Kyoto}, \mathbf{e}_{contains}, \mathbf{n}_{Kinkaku-ji}, \dots] \tag{5}$$

We pass this unmasked sequence through the Graph-Adapter to obtain the target hidden state. We then minimize the distance between the text-derived hidden state of `<entity_start>` and the graph-derived representation at the target node index:

$$\mathcal{L}_{align} = \text{Distance}\Big(f_{text}(X_{text})[\texttt{<entity\_start>}],\ f_{graph}(X^*_{graph})[\text{target\_idx}]\Big) \tag{6}$$

1.2.4 Whitening Transformation

We observe that the Graph-Adapter is susceptible to representation collapse, leading to anisotropic embeddings where the model converges toward trivial solutions. To address this, we implement a **Whitening Transformation** [7], a post-processing technique that transforms the embedding distribution to be isotropic (zero mean and identity covariance).

Given a batch of raw entity embeddings derived via the "First-Last-Avg" strategy, we compute the mean vector μ and the transformation matrix W such that the covariance of the transformed embeddings becomes the identity matrix I. The whitened representation $\tilde{x}$ for an entity x is computed as:

$$\tilde{x} = (x - \mu)W \tag{7}$$

This operation effectively eliminates the representation collapse by enforcing a standard orthogonal basis in the embedding space. This serves as a hard constraint on the geometry of the graph representations, ensuring they remain semantically distinct and robust without requiring auxiliary decorrelation loss functions.

2 Inference and Constrained Generation

Scent introduces a high-efficiency inference mechanism designed to circumvent the latency bottlenecks typical of constrained generation. By leveraging the encoder-only architecture, we score all potential entities from the candidate set $\mathcal{C}$ simultaneously in a single forward pass. This approach guarantees that the output is strictly valid (i.e., exists within the known candidate set) while maintaining the semantic consistency enforced during training.

2.1 Offline Structural Indexing

Prior to inference, we construct a dense vector index of the Knowledge Graph. This is a one-time, offline process utilizing the Graph-Adapter.

For every unique entity $u \in \mathcal{C}$, we construct an input sequence X_u consisting of the entity's **name** and **short abstract**. We pass this sequence through the encoder, apply the "First-Last-Avg" pooling strategy, and project the result using the pre-computed whitening parameters (μ, W) to form the static index $\mathcal{M}$ (Equation 8):

$$\mathcal{M}[u] = \text{Whitening}\Big(\text{Pool}(f_{graph}(X_u))\Big) \tag{8}$$

This map provides the "ground truth" semantic vectors against which text generation will be reranked, ensuring that the predicted entity aligns with its structured representation in the graph.

2.2 Parallel Linguistic Evaluation

During inference, the model operates primarily in LoRA-Text mode. The input query is prepared with a mask buffer of fixed length N at the target entity position.

Unlike autoregressive models that generate tokens sequentially, Scent evaluates the likelihood of the entire candidate vocabulary $\mathcal{V}$ across all N mask positions in parallel. We employ a highly optimized tensor "gather" operation to extract specific log-probabilities corresponding to the pre-tokenized sequences of the candidate set.

For a candidate c, we append the special `<end_of_title>` token to its token sequence, resulting in $[t_1, t_2, \ldots, t_k, t_{end}]$. The linguistic score S_{ling} is calculated as the length-normalized sum of log-probabilities for the full sequence, rewarding the model for correctly predicting both the entity content and its termination point:

$$S_{ling}(c) = \frac{1}{k} \sum_{j=1}^{k} \log P(t_j \mid X_{text}) \tag{9}$$

This vectorization allows Scent to evaluate thousands of candidates instantly, with the constraint being that entity token sequences must fit within the mask buffer N.

2.3 Cross-Modal Alignment and Re-ranking

To ensure the generated entity is not only linguistically fluent but also topologically consistent with the knowledge graph, we refine the initial linguistic scores using the graph alignment objective. In the same forward pass used for text generation, the model projects a predicted structural representation $\hat{\mathbf{v}}$ from the hidden state of the `<entity_start>` token. This vector represents the model's expectation of the entity's graph position given the textual context and the disambiguated mention.

It is important to note that despite utilizing vector similarity, this stage does not constitute a standard dense vector retrieval task. In a typical dense retrieval setting, the query vector would be compared against the entire index $\mathcal{M}$ (which could contain millions of entities) to find the nearest neighbors.

Instead, Scent employs a **cascade re-ranking strategy**. We utilize the linguistic scores S_{ling} as a high-recall filter to prune the search space, selecting only a subset $\mathcal{C}_{top}$ consisting of the top-k most likely textual candidates (e.g., the top 50 or 100 matches). The structural verification is computed exclusively for this reduced subset.

For each candidate $c \in \mathcal{C}_{top}$, we retrieve its pre-computed embedding $\mathcal{M}[u_c]$ from the index and calculate the final score as a weighted combination of the linguistic probability and the cosine similarity between the predicted and actual graph vectors (Equation 10):

$$S_{total}(c) = S_{ling}(c) + \alpha \cdot \cos(\hat{\mathbf{v}}, \mathcal{M}[u_c]) \tag{10}$$

3 Preliminary Experiments

We present a preliminary evaluation of the Scent framework. The primary objective of these experiments is to verify the hypothesis that grounding the generation in structured graph knowledge alters the semantic distribution of the output. These experiments represent a "proof-of-concept" conducted under constrained compute resources (limited to 3 training epochs on the AIDA train set) and serve as a baseline for future optimization.

3.1 Experimental Setup

Architecture & Backbone: We utilize RoBERTa [1] as the encoder backbone. To maintain computational efficiency, the backbone parameters are frozen. We train the model using Low-Rank Adaptation (LoRA) [2].

Datasets: We employ the following resources for graph construction, training, and evaluation:

- **Knowledge Graph (YAGO4):** We construct our graph structure using **YAGO4** [4], leveraging its rich taxonomy and rigorous type constraints.

- **Entity Abstracts (Structured Wikipedia):** To generate the semantic initialization for graph nodes (as described in the "First-Last-Avg" strategy), we extract short abstracts for every entity from the **Structured Wikipedia** dataset [8].
- **Training Data (AIDA-YAGO2):** We train the model exclusively on the training split of the **AIDA-YAGO2** dataset [5]. Mentions in AIDA-YAGO2 were mapped to their corresponding nodes in YAGO4 to create ground truth targets.
- **Robustness Evaluation (ShadowLink):** To evaluate the model's resilience against prior bias, we utilize the **ShadowLink** benchmark [9]. ShadowLink is specifically designed to assess "Entity Overshadowing", a phenomenon where models favor common entities (Top) over less popular ones (Shadow) sharing the same surface form.

Configurations: We compare two variations of our framework to conduct an ablation study on the impact of structural grounding:

1. **Scent (Text-Only):** This configuration utilizes the **Text-Adapter** (f_{text}) and the specific fixed-buffer masking strategy required for constrained generation. It is trained solely with the Masked Language Modeling objective ($\mathcal{L}_{text}$). This baseline relies exclusively on the linguistic patterns available in the pre-trained encoder and the textual context.
2. **Scent (Text + Graph):** The full framework, utilizing both the **Text-Adapter** and **Graph-Adapter** (f_{graph}). It is trained with the multi-task objectives: Text Generation ($\mathcal{L}_{text}$), Feature Regression ($\mathcal{L}_{graph}$), and Cross-Modal Alignment ($\mathcal{L}_{align}$).

3.2 Candidate Space Restrictions

It is crucial to note that our experimental setup differs from standard State-of-the-Art (SoA) evaluations. Standard evaluations typically rank candidates against the entire Wikipedia entity set (millions of nodes). However, to isolate the impact of structural grounding without the noise of massive retrieval, we restricted the candidate set $\mathcal{C}$ for inference:

- **AIDA Evaluation:** The candidate pool consists of the union of all entities present in the AIDA Train and Test sets.
- **ShadowLink Evaluation:** The candidate pool consists of the union of the Top, Shadow, and Tail subsets.

Because the candidate space is significantly smaller, absolute accuracy scores are naturally higher than standard benchmarks. Specifically for ShadowLink, this setup often simplifies the task to a binary or ternary decision between a "Top" entity and a "Shadow" entity sharing the same surface form. Therefore, these results should be interpreted as a relative comparison of model bias rather than absolute retrieval performance.

3.3 Results and Analysis

Table 1 summarizes the Top-1 Accuracy across the AIDA test set and the three ShadowLink subsets (Top, Shadow, and Tail).

Table 1: Top-1 Accuracy on AIDA-YAGO2 and ShadowLink subsets. The inclusion of graph constraints results in a minor redistribution of performance, showing slight gains on Shadow and Tail entities.

Configuration	AIDA (Test)	ShadowLink (Top)	ShadowLink (Shadow)	ShadowLink (Tail)
Scent (Text-Only)	0.8190	**0.5022**	0.4159	0.8783
Scent (Text + Graph)	**0.8201**	0.4900	**0.4215**	**0.8905**
Diff	*+0.11%*	*-1.22%*	*+0.56%*	*+1.22%*

Stability on Standard Benchmarks: On the standard AIDA test set, the performance difference between the two configurations is negligible (+0.11%). This suggests that within the constraints of this preliminary setup, the addition of auxiliary graph objectives ($\mathcal{L}_{graph}$ and $\mathcal{L}_{align}$) does not negatively impact the model's general capability to resolve standard entity mentions found in news corpora.

Analysis of Overshadowing and Priors: The results on the ShadowLink subsets reveal a subtle trade-off in the model's decision boundary. The **Text + Graph** configuration shows a slight decrease in accuracy on **Top** entities (-1.22%), accompanied by marginal improvements on **Shadow** (+0.56%) and **Tail** (+1.22%) entities.

While these shifts are small, they align with the phenomenon of "entity overshadowing" described in [9]. The authors define overshadowing as a scenario where a common entity (e_1) is selected over a less common candidate (e_2) due to a higher prior probability ($P(e_1) > P(e_2)$), even when they share the same surface form. They argue that high performance on standard datasets is often overestimated, as models can achieve high accuracy on "Top" entities by merely learning this prior distribution rather than utilizing local context [9].

The slight drop in "Top" accuracy combined with the lift in "Shadow" accuracy suggests that grounding the model in the graph structure may act as a weak regularizer, slightly dampening the model's tendency to default to the popularity prior.

Finally, regarding the **Tail** subset, it is important to note the distinction made in [9] between "overshadowed" and "long-tail" entities. The authors observe that while long-tail entities are rare, they are often unambiguous, making them easier for systems to resolve compared to the ambiguous "Shadow" entities. Our results reflect this trend, with both configurations achieving their highest ShadowLink scores on the Tail subset. The slight improvement here (+1.22%) suggests that structural grounding does not hinder and may marginally assist the retrieval of unambiguous rare entities, though the primary challenge remains the disambiguation of common but overshadowed mentions.

References

[1] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR, abs/1907.11692*.

[2] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., & Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. *CoRR, abs/2106.09685*.

[3] Kim, J., Nguyen, T. D., Min, S., Cho, S., Lee, M., Lee, H., & Hong, S. (2022). Pure Transformers are Powerful Graph Learners. *arXiv preprint arXiv:2207.02505*.

[4] Pellissier Tanon, T., Weikum, G., & Suchanek, F. (2020). YAGO 4: A Reason-able Knowledge Base. In *The Semantic Web* (LNCS 12123, pp. 583–596). Springer.

[5] Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., & Weikum, G. (2011). Robust Disambiguation of Named Entities in Text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 782–792.

[6] Zbontar, J., Jing, L., Misra, I., LeCun, Y., & Deny, S. (2021). Barlow Twins: Self-Supervised Learning via Redundancy Reduction. *arXiv preprint arXiv:2103.03230*.

[7] Su, J., Cao, J., Liu, W., & Ou, Y. (2021). Whitening Sentence Representations for Better Semantics and Faster Retrieval. *arXiv preprint arXiv:2103.15316*.

[8] Wikimedia Enterprise, Wikimedia Foundation. (2024). Structured Wikipedia. *Hugging Face Datasets*.

[9] Provatorova, V., Vakulenko, S., Bhargav, S., & Kanoulas, E. (2021). Robustness Evaluation of Entity Disambiguation Using Prior Probes: the Case of Entity Overshadowing. *arXiv preprint arXiv:2108.10949*.