Graphs of n-grams for Text Generation and Characteristic Phrase Extraction in the Domain of Smell

Nadezhda Komarova
Jožef Stefan Institute
Ljubljana, Slovenia
Technical University of Munich
Munich, Germany

Inna Novalija
inna.koval@ijs.si
Jožef Stefan Institute
Ljubljana, Slovenia

Abstract

We present a transparent, graph-based approach to text generation and characteristic phrase extraction in the domain of smell. Weighted directed graphs of n-grams are constructed from the smell-filtered portion of the Google n-grams dataset (20,000–60,000 n-grams), and three pathfinding strategies – a greedy algorithm, the Bellman–Ford algorithm, and topological sorting – generate continuations of olfactory seed phrases. As reference texts are unavailable, generation is evaluated with two graph-tailored metrics complemented by sentence-transformer coherence scores, and we study the effect of the seed n-gram degree, the dataset size, and whether the graph is cyclic. Building on the same structures, vertex-level clustering coefficients and closeness centrality extract phrases characteristic of olfactory subdomains such as floral scents, under three settings: default, verb-filtered, and generation-augmented. The two families of metrics are found to disagree – the greedy algorithm scores highest on the graph-tailored metrics while the global methods score highest on coherence – and the three extraction settings surface qualitatively different olfactory vocabulary, offering an interpretable complement to purely neural approaches.

Keywords

Smell, Olfactory Language, Graphs, n-grams, Clustering Coefficients, Centrality Measures, Text Generation, Keyphrase Extraction

1 Introduction

Olfactory language – the vocabulary used to describe scents, fragrances and odours – is notoriously difficult to process automatically. Unlike vision or sound, smell lacks a small, well-agreed vocabulary of primitives, and descriptions of scent are improvised and metaphorical: a perfume is *powdery* or *green*, an animalic note *sensual* in marketing copy and *dirty* in a review. Identifying characteristic olfactory phrases nevertheless has practical value: fragrance companies catalogue and search products by textual description, and recommender systems relate free text to scent families such as floral, musky or citrus.

Large language models generate fluent olfactory text but are opaque: it is hard to inspect *why* a phrase was produced, or to keep a description anchored to one scent family. This motivates interpretable, lightweight alternatives that explicitly model co-occurrence relationships between phrases and allow generation and extraction to be traced in a structured manner. This is knowledge discovery from a corpus: n-gram statistics are mined, represented as a weighted graph, and mined again with network measures.

This paper investigates n-gram graph structures as such an alternative. The first stage predicts a plausible continuation of an initial olfactory n-gram: graphs are built from co-occurrence statistics in the Google n-grams dataset [10], and prediction is formulated as graph traversal using a greedy algorithm, the Bellman–Ford algorithm and topological sorting. As no reference texts are available, we introduce graph-tailored metrics and complement them with sentence-transformer coherence scores. The second stage reuses the same construction to extract *characteristic phrases* of smell subdomains – floral scents, for example – from vertex-level clustering coefficients and closeness centrality, under three extraction settings.

The contributions are: (i) a transparent n-gram graph model for phrase prediction in the smell domain, evaluated across pathfinding strategies, seed degree, dataset size and graph structure; (ii) graph-tailored metrics for reference-free generation, complemented by embedding-based coherence; and (iii) a method for extracting characteristic phrases of olfactory subdomains using clustering coefficients and centrality.

2 Related Work

Text generation and n-gram models. Neural text generation has moved from recurrent networks to contextualised representations and transformers [7, 8]. Naive n-gram models remain attractive for their low cost and ease of domain adaptation, though their output quality is well below state-of-the-art transformers.

Recent work narrows this gap: Shareghi et al. [15] establish stronger n-gram baselines, Li et al. [6] enhance an n-gram basis with a neural model, and Ikegami et al. [4] combine n-grams with a recurrent network. Closest to our setting, Nagalavi and Hanumanthappa [12] predict newspaper article-block sequences from n-gram probabilities, and Dasgupta et al. [1] study higher-order n-grams with Laplace smoothing. We reuse such probabilities not for prediction but as *edge weights*, letting graph traversal extract further structure.

Graph-based approaches. Graphs are widely used as input structure for generation: Koncel-Kedziorski et al. [5] generate text from knowledge graphs with graph transformers, and Gao et al. [2] use graph-augmented encoders for RDF-to-text. Graph algorithms also serve linguistic tasks directly: Prabhumoye et al. [13] cast sentence ordering as topological sorting, and Tromp and Pechenizkiy [16] combine n-gram frequencies with a graph for language identification.

Clustering coefficients and phrase extraction. Clustering coefficients quantify connected triangles in a network, applied for instance to small-worldness in brain networks [9]. We use them alongside closeness centrality, both indicating vertex importance.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SiKDD 2026, October 2026, Ljubljana, Slovenia
© 2026 Copyright held by the owner/author(s).
https://doi.org/10.70314/is.2026.sikdd.91

Our task extends keyphrase extraction – detecting phrases that summarise a document – from one document to an entire domain, here *smell*.

3 Methodology

An overview of the two stages is given in Figure 1. Both stages operate on the same weighted directed graph; the dashed arrow marks the generation-augmented setting, where Stage 1 output feeds Stage 2.

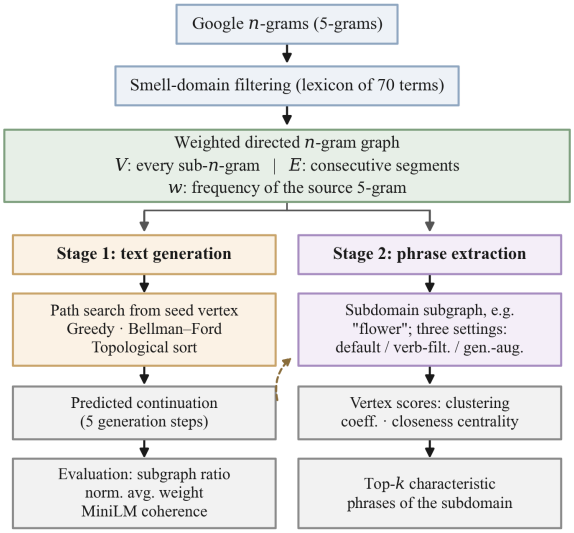


Figure 1: Overview of the methodology.

3.1 Stage 1: Text Generation

3.1.1 Constructing the graph. The graph is built from the smell-related 5-grams of the Google n-grams dataset. Each 5-gram is partitioned in every possible way, i.e. into every concatenation of 1-, 2-, 3-, 4- and/or 5-grams, each a contiguous substring of the initial 5-gram. Every such substring becomes a vertex of a directed graph, and an edge connects a substring to the one that follows it in the source 5-gram, weighted by the frequency of that 5-gram. Because substrings are shared across partitions, vertices are reused and the result is a genuine graph rather than a set of disjoint chains.

3.1.2 First approach: greedy algorithm. The generated text corresponds to a path through the graph. The algorithm starts at the vertex matching the *initial n-gram* to be completed, and at each step selects the locally optimal continuation, i.e. the outgoing edge of maximum weight. The number of steps equals the number of individual phrases to be predicted. Its weakness is that always selecting the locally optimal phrase may degrade quality for longer generated texts, which motivates the two global alternatives below.

3.1.3 Second approach: Bellman–Ford. This method finds the path between two vertices with maximal edge-weight sum. The Bellman–Ford algorithm, normally used for single-source shortest paths, is repurposed by multiplying edge weights by -1; for this to be correct, negative cycles must be eliminated. The graph is therefore replaced by its maximum spanning forest, the collection of maximum spanning trees. To generate text, the subgraph in the proximity of the source vertex is selected via depth-first search, containing all vertices whose shortest path from the source uses at most as many edges as the degree of the n-gram to be generated. Finding the spanning forest costs $O(m \log n)$ for n vertices and m edges. It is preferred over decomposition into strongly connected components ($O(n + m)$), which can leave semantically related concepts mutually unreachable. Spanning-forest construction discards information, but connectedness is itself of interest for the later centrality and clustering measures.

3.1.4 Third approach: topological sorting. This approach shares the intuition of the previous one. The maximum spanning forest is computed to eliminate negative cycles and the topological order of the vertices is obtained. Every vertex is assigned an initial distance of $-\infty$, except the source, which is assigned 0. Edges are then sorted ascending, first by the topological order of the origin vertex and then by that of the target vertex, and each edge is considered in turn for whether it improves the current distance to its target – the established technique for longest-path finding in a directed acyclic graph.

3.2 Stage 2: Characteristic Phrase Extraction

The second task is extracting phrases characteristic of a given subdomain of smell. To assess phrase significance and the semantic connections between expressions within a subdomain, clustering coefficients and centrality measures are computed for every vertex. Since the graph is weighted by n-gram frequencies, the clustering coefficient c_u of vertex u is defined as in [11],

$$c_u = \frac{1}{\deg(u)\big(\deg(u)-1\big)} \sum_{\substack{v,w \in N(u) \\ v \neq w}} \big(\widehat{w}_{uv}\widehat{w}_{uw}\widehat{w}_{vw}\big)^{\frac{1}{3}}, \qquad (1)$$

where $N(u)$ is the set of neighbours of u, $\deg(u)$ its degree, and $\widehat{w}_{uv} = w_{uv}/\max(w)$ the weight of the edge between u and v normalised by the maximum edge weight in the graph. The sum runs over ordered pairs of neighbours, so each triangle contributes twice.

3.2.1 Extraction procedure. Phrases whose vertices have higher clustering coefficients or centrality are assumed to be more characteristic of the subdomain. The procedure is: (1) select a list of words defining subdomains of smell, e.g. *musk*, *vanilla*, *flower*, *rose*, *salt*, *burnt*, *lemon*, *lavender*, *sulfur*, *perfume*, *vinegar*, chosen to span the perceptual families of perfumery and to vary in corpus frequency; (2) filter the data to n-grams containing that word and build a graph as in Section 3.1.1; and (3) compute clustering coefficients and closeness centrality per vertex, taking the top k as characteristic.

3.2.2 The three settings. Three variants of this procedure are compared. The **default** setting applies the procedure unchanged. The **verb-filtered** setting additionally requires each n-gram to contain a smell-related verb. The **generation-augmented** setting adds a preprocessing step in which each original n-gram is extended by the topological-sorting generator of Section 3.1.4, the extended n-grams replacing the originals before the graph is built.

Table 1: Text generation in the smell domain: subgraph ratio (SR) and normalised average weight (NW), aggregated over each factor in turn.

	SR	NW
Pathfinding approach		
Greedy	19.91	0.0303
Bellman–Ford	16.90	0.0312
Topological sorting	16.90	0.0312
Seed n-gram degree		
2	10.85	0.0186
3	15.82	0.0279
4	22.98	0.0356
5	26.75	0.0459
Graph type		
Acyclic	16.75	0.0301
Cyclic	23.38	0.0327
Dataset size (n-grams)		
20,000	16.45	0.0253
40,000	17.48	0.0286
60,000	20.80	0.0370

4 Experimental Settings

4.1 Stage 1: Text Generation

Prediction quality is compared across four models: greedy on the spanning forest (acyclic), greedy on the full graph (cyclic), Bellman–Ford, and topological sorting.

Metrics. The *subgraph ratio* is the mean edge weight along the predicted path divided by the mean edge weight over the whole subgraph defined by the number of prediction steps. The *normalised average weight* is the mean edge weight of the predicted path divided by the maximum edge weight in the graph. Coherence is measured as the cosine similarity between sentence embeddings of the generated text produced by the `all-MiniLM-L6-v2` sentence transformer [14].

Setups. All setups use 5 prediction steps. The effect of seed length is studied on sets of 100 randomly selected *n*-grams from the Google *n*-grams dataset consisting solely of 2-, 3-, 4- or 5-grams. Where a whole seed phrase was not present as a vertex, its longest contained suffix was used. The dataset was filtered to the smell domain using the expressions *stink, scent, scents, smell, smells, odour, odor, odours, odors, stench, reek, aroma, aromas, aromatic, whiff etc.* Words containing non-alphabetical characters were removed. The expressions are thesaurus synonyms of olfactory nouns, verbs and adjectives, with inflectional variants included so that frequency mass is not split across forms.

4.2 Stage 2: Characteristic Phrase Extraction

Subdomains are specified by restricting the dataset to *n*-grams containing the defining phrase, and characteristic phrases are those whose vertices have the highest clustering coefficients or closeness centrality; the top $k = 10$ are reported. Clustering coefficients and centrality are computed with NetworkX [3]. The verbs used for verb-filtering are *smell(s/ed/ing)*, *reek(s/ed/ing)*, *waft(s/ed/ing)*, *stink(s/ing)*, *stank*, *stunk* and *perfume(d/ing)*, on the intuition that the domain contains many *smelling like* constructions relating qualitatively different concepts.

5 Results and Discussion

5.1 Stage 1: Text Generation

Results are given in Tables 1 and 2. The graph-tailored metrics are identical for Bellman–Ford and topological sorting, as expected:

Table 2: Average coherence of generated text (cosine similarity between `all-MiniLM-L6-v2` embeddings), at seed degree 4 and 60,000 *n*-grams.

Graph type	Approach	Avg. coherence
Acyclic	Greedy	0.2175
Cyclic	Greedy	0.2230
Acyclic	Topological sorting	0.2365
Acyclic	Bellman–Ford	**0.2366**

both find the maximum-weight path from the same acyclic subgraph. Reporting both remains meaningful because they diverge under other cycle-elimination strategies. Cyclic graphs yield higher subgraph ratios than acyclic ones, though this may reflect that metric's sensitivity to edge count; the normalised average weight is comparable. The two global approaches consistently yield *lower* subgraph ratios than greedy search – counterintuitive, since greedy is the method prone to local optima, and possibly a property of this smell-filtered subset. Greedy is also the lightest computationally.

Seed length. Performance could not be measured for seeds absent from the graph; resolving them by semantic similarity to the nearest contained phrase would raise cost. High-weight paths are likelier from short seeds, since more vertices correspond to low-degree *n*-grams.

Dataset size. Both metrics increase with dataset size and are comparable at 40,000 and 60,000 *n*-grams. When the dataset was too small, few seeds had a large enough connected neighbourhood: generation steps could not be completed, or were confined to a cycle, suggesting many small strongly connected components.

Metric disagreement. The two families of metrics do not agree (Table 2). Greedy search scores highest on the graph-tailored metrics, yet Bellman–Ford and topological sorting achieve higher coherence. Globally optimal paths therefore yield more coherent output. The graph-tailored scores are functions of edge weight alone, i.e. of corpus transition frequency, and so indifferent to topical drift, near-duplicate wording and grammaticality – the properties coherence responds to.

Qualitative inspection indicated that the generated texts are not comparable in fluency to LLM output, motivating the shift to Stage 2, where the graph is used for extraction rather than generation. Even so, deterministic behaviour has value where a reproducible choice among near-synonymous descriptors is preferable to opaque generation.

Temporal composition. Because the frequencies are aggregate counts, the graphs describe olfactory language across the corpus as a whole rather than in any single period. For *flower* the phrases that surface are predominantly contemporary and commercial (Table 3), indicating where the frequency mass of the subdomain lies.

5.2 Stage 2: Characteristic Phrase Extraction

Table 3 presents results for *flower*. The separation into subdomains is not absolute; *subdomain* here denotes smells tied to a set of circumstances. Clustering coefficients and centrality select overlapping *n*-grams and appear comparably meaningful. The three settings surface different aspects of the subdomain.

Table 3: Top 10 phrases for the *flower* subdomain of smell, ranked by clustering coefficient (CC); three extraction settings.

Default		**Verb-filtered**		**Generation-augmented**	
n-gram	CC	*n*-gram	CC	*n*-gram	CC
Essential Oils Aromatic	0.0083	brazil flower fragrant perfume	0.0273	of this vivacious	0.0026
Fragrance Flower Fresh Flower	0.0075	smell the flowers along	0.0251	and decoration set	0.0019
Fresh Flower Ftd	0.0075	smell the flowers in	0.0233	I have ever	0.0014
Drawings Flower Essence Flower	0.0075	smell of wild flowers	0.0213	that grow only	0.0013
Essences Flower Fairy	0.0075	sweet smelling yellow flowers	0.0203	bloom from July	0.0013
Essence Flower Essences Flower	0.0075	smells of the flowers	0.0199	in late winter	0.0012
candles romantic	0.0065	smell the flowers on	0.0128	of the Gardenia	0.0012
fragrance is a	0.0060	sweet smell of flowers	0.0119	covering the entire	0.0012
and seasonal blooms	0.0058	the flowers smell and	0.0116	attract insects	0.0005
New Action	0.0058	smell the flowers or	0.0100	include freesia	0.0004

The **default** setting favours compound noun phrases tied to fragrance products. The **verb-filtered** setting targets the *"subject smells [like] object"* construction, recovering the *object* as a characteristic *n*-gram; it yields longer phrases but repeats near-identical variants ("smell the flowers along", "…in", "…on"). The **generation-augmented** setting, extending each *n*-gram by 5 generation steps before construction, produces more specific descriptors ("of the Gardenia", "include freesia"); one generation step did not change the top 10 appreciably. No setting is uniformly best: the default and verb-filtered settings might suit fragrance recommendation, the generation-augmented setting fine-grained vocabulary discovery – provided its output is validated. Numerical comparison with Section 2 is obstructed by the absence of a shared task: those methods are scored against held-out text or per-document gold keyphrases, neither available here.

6 Conclusion and Future Work

This paper explored graph algorithms for text generation and characteristic phrase extraction in the domain of smell using simple *n*-gram graph models. We examined the influence of dataset size, *n*-gram length, graph structure, and pathfinding algorithm on generation quality, and extracted phrases characteristic of olfactory subdomains (*flower*) under three settings. Human evaluation is the clearest next step: annotators rating whether each extracted phrase is characteristic of its subdomain, and a blind comparison of 20–30 continuations against LLM output. A frequency baseline should accompany it, with dispersion reported alongside the means, and a time-stamped corpus would allow the same construction per period. Reference-free metrics and a scalability analysis would clarify the interpretability–runtime trade-off [4]. Subdomains could be defined more flexibly through multi-term filtering or unsupervised graph clustering [17]. Applying the extracted phrases in fragrance recommendation or retrieval-augmented generation could further establish their practical value.

Acknowledgements

This work was supported by the Slovenian Research Agency under the project J7-50233, by the European Union through enrichMyData EU HORIZON-IA project under grant agreement No 101070284 and DataPACT EU HORIZON project under grant agreement No 101189771.

References

[1] Shibaranjani Dasgupta, Chandan Maity, Somdip Mukherjee, Rohan Singh, Diptendu Dutta, and Debasish Jana. 2024. HITgram: a platform for experimenting with *n*-gram language models. (2024). arXiv: 2412.10717 [cs.CL].

[2] Hanning Gao, Lingfei Wu, Po Hu, and Fangli Xu. 2020. RDF-to-text generation with graph-augmented structural neural encoders. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 3030–3036. doi:10.24963/ijcai.2020/419.

[3] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy)*, 11–15.

[4] Yukino Ikegami, Setsuo Tsuruta, Andrea Kutics, Ernesto Damiani, and Rainer Knauf. 2024. Fast ML-based next-word prediction for hybrid languages. *Internet of Things*, 25, 101064. doi:10.1016/j.iot.2024.101064.

[5] Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2284–2293. doi:10.18653/v1/N19-1238.

[6] Huayang Li, Deng Cai, Jin Xu, and Taro Watanabe. 2022. N-gram is back: residual learning of neural text generation with *n*-gram language model. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 1523–1533. doi:10.18653/v1/2022.findings-emnlp.109.

[7] Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Pretrained language models for text generation: a survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 4492–4499. doi:10.24963/ijcai.2021/612.

[8] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. 2022. A survey of transformers. *AI Open*, 3, 111–132. doi:10.1016/j.aiopen.2022.10.001.

[9] Naoki Masuda, Michiko Sakaki, Takahiro Ezaki, and Takamitsu Watanabe. 2018. Clustering coefficients for correlation networks. *Frontiers in Neuroinformatics*, 12, 7. doi:10.3389/fninf.2018.00007.

[10] Jean-Baptiste Michel et al. 2011. Quantitative analysis of culture using millions of digitized books. *Science*, 331, 6014, 176–182. doi:10.1126/science.1199644.

[11] Kent Miyajima and Takashi Sakuragawa. 2014. Continuous and robust clustering coefficients for weighted and directed networks. (2014). arXiv: 1412.0059 [cs.SI].

[12] Deepa Nagalavi and M. Hanumanthappa. 2016. N-gram word prediction language models to identify the sequence of article blocks in English e-newspapers. In *2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*. IEEE, 307–311. doi:10.1109/CSITSS.2016.7779376.

[13] Shrimai Prabhumoye, Ruslan Salakhutdinov, and Alan W Black. 2020. Topological sort for sentence ordering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2783–2792. doi:10.18653/v1/2020.acl-main.248.

[14] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP-IJCNLP)*, 3982–3992. doi:10.18653/v1/D19-1410.

[15] Ehsan Shareghi, Daniela Gerz, Ivan Vulić, and Anna Korhonen. 2019. Show some love to your *n*-grams: a bit of progress and stronger *n*-gram language modeling baselines. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 4113–4118. doi:10.18653/v1/N19-1417.

[16] Erik Tromp and Mykola Pechenizkiy. 2011. Graph-based n-gram language identification on short texts. In *Proceedings of the 20th Machine Learning Conference of Belgium and The Netherlands (Benelearn)*, 27–34.

[17] Timothé Watteau, Aubin Bonnefoy, Simon Illouz-Laurent, Joaquim Jusseau, and Serge Iovleff. 2024. Advanced graph clustering methods: a comprehensive and in-depth analysis. (2024). arXiv: 2407.09055 [cs.LG].