Extended abstract

On the realizations of sequence graphs

Sammy Khalife[1]*, Yann Ponty[1], Laurent Bulteau[2]

[1] *LIX, CNRS, Ecole Polytechnique, Institut Polytechnique de Paris, 91128 Palaiseau, France*
[2] *LIGM, CNRS, Université Gustave Eiffel, 77454 Marne-la-Vallée, France*
***Corresponding author**: khalife@lix.polytechnique.fr

Abstract
Several language models rely on an assumption modeling each local context as a (potentially oriented) bag of words, and have proven to be very efficient baselines. Sequence graphs are the natural structures encoding their information. However, a sequence graph may have several realizations as a sequence, leading to a degree of ambiguity. Several combinatorial problems are presented, depending on three levels of generalisation (window size, graph orientation, and weights). We present some complexity results and a dynamic programming algorithm to measure this level of ambiguity.

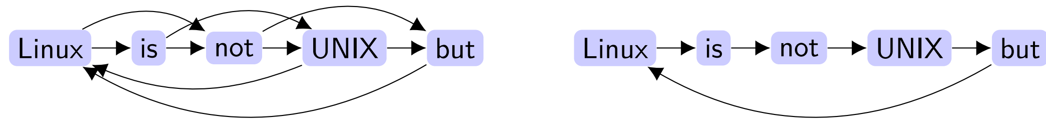
Keywords
Sequence Algorithms — Graphs — Natural Language Models — Inverse problem

1. Introduction

The automated treatment of familiar objects, either natural or artifacts, always relies on a translation into entities manageable by computer programs. However, the correspondence between the object to be treated and "its" representation is not necessarily one-to-one. The representations used for learning algorithms are no exception to this rule. In particular, natural language words and textual documents representations are essential for several tasks, including document classification [1], role labelling [2], and named entity recognition [3]. The traditional models based on pointwise mutual information, or graph-of-words (GOW), [4, 5, 6], supplement the content of bag-of-words (TF, TFIDF) with statistics of co-occurrences within a **window** of fixed size w, introduced to mitigate the degree of ambiguity. Several models [7, 8, 9, 10] also use the same type of information and constitute strong baselines for natural language processing. While these representations are more precise than the traditional bag-of-words (e.g Parikh vectors), they still induce some level of ambiguity, *i.e.* a given graph can represent several sequences. Our study is thus motivated by a quantification of the level of ambiguity, seen as an algorithmic problem, coupled with an empirical assessment of the consequences of ambiguity for the representations.

2. Definitions and problem statement

Let $x = x_1, x_2, ..., x_p$ be a finite sequence of discrete elements among a finite vocabulary X. Without loss of generality, we can suppose that $X = \{1, ..., n\}$. In the following, let $I_p = \{1, ..., p\}$. This motivates the following definition:



(a) No ambiguity ($w = 3$) **(b)** Ambiguity ($w = 2$)

Figure 1. Sequence graphs (or *graphs-of-words*) built for the sentence "Linux is not UNIX but Linux" using window sizes 3 (a) and 2 respectively (b). In the second case, the sequence graph is ambiguous, since any circular permutation of the words admits the same representation.

Definition 1 *$G = (V, E)$ is the graph of the sequence x with window size $w \in \mathbb{N}^*$ if and only if $V = \{x_i \mid i \in I_p\}$, and*

$$(i, j) \in E \iff \exists (k, k') \in I_p^2,\ |k - k'| \leq w - 1\ x_k = i \text{ and } x_{k'} = j \tag{1}$$

For digraphs, Eq. (1) is replaced with

$$(i, j) \in E \iff \exists (k, k') \in I_p^2,\ k \leq k' \leq k + w - 1, x_k = i \text{ and } x_{k'} = j. \tag{2}$$

Finally, a weighted sequence graph G is endowed with a matrix $\Pi(G) = (\pi_{ij})$ such that

$$\pi_{ij} = \text{Card}\ \{(k, k') \in I_p^2 \mid k \leq k' \leq k + w - 1,\ x_k = i \text{ and } x_{k'} = j\} \tag{3}$$

We say that x is a w-admissible sequence for G (or a realization of G), if G is the graph of sequence x with window size w.

The natural integers π_{ij} represent the number of co-occurrences of i and j in a window of size w. Hence, the graph of sequence is unique. An linear time algorithm to construct a weighted sequence digraph is obtained by sliding a window of size w over the sequence and incrementing the counter of presence of two elements in the window. This construction defines a correspondence between the sequence set $X^\star$ into the graph set $\mathcal{G}$: $\phi_w \colon X^\star \to \mathcal{G}, x \mapsto G_w(x)$. Based on these definitions, we consider the following problems:

Problem 1 (Weighted-Realizable (W-Realizable))
Input: *Possibly directed graph G, matrix weights Π, window size w*
Output: *True if (G, Π) is the w-sequence graph of some sequence x, False otherwise.*

Problem 2 (Unweighted-Realizable (U-Realizable))
Input: *Possibly directed graph G, window size w*
Output: *True if G is the w-sequence graph of some sequence x, False otherwise.*

We denote *D*-Realizable (resp. *G*-) the restricted version of Realizable where the input graph G is directed (resp. undirected), and *W*-Realizable (resp. *U*-) the restricted version of Realizable where the input graph G is weighted (resp. unweighted), possibly in combination with the D- or G- variants. We write Realizable$_w$ for the case where w is a fixed (given) constant. We also consider the variants of W-Realizable, denoted WG-Realizable and WD-Realizable where the input graph is restricted to be respectively undirected and directed. We define UG-Realizable and UD-Realizable similarly. Finally, we write (WG-, WD-, ...)Realizable$_w$ for the case where w is a fixed strictly positive integer.

Problem 3 (Unweighted-NumRealizations (U-NumRealizations))
Input: *Possibly directed graph G, window size w*
Output: *The number of **realizations** of G,* i.e. *preimages of G through ϕ_w such that $|\{x \in X^\star \mid \phi_w(x) = G\}|$ if finite, or $+\infty$ otherwise.*

Problem 4 (Weighted-NumRealizations (W-NumRealizations))
Input: *Possibly directed graph G, matrix weights Π, window size w*
Output: *The number of **realizations** of G in the weighted sense.*

Similarly, we use the same prefix for the directed or undirected versions of (D-, G-, i.e. DU- for directed and unweighted):

DW Directed weighted	**DU** Directed unweighted
GW Undirected weighted	**GU** Undirected unweighted

We also denote NumRealizations$_w$ for the case where w is a fixed strictly positive integer. Note that NumRealizations strictly generalizes the previous one, as Realizable can be solved by testing the nullity of the number of suitable realization computed by NumRealizations.

3. Main theoretical results

3.1 Complete characterization of 2-sequence graphs

Table 1. Complexity for various instances of our problems ($w = 2$)

	NumRealizations$_2$		Realizable$_2$	
Variation	Complexity	#Sequences	Complexity	Characterization
GU	P	$\{0, +\infty\}$	P	G connected
GW	#P-hard	$\{0, 1\} \cup 2\mathbb{N}^*$	P	$\psi(G)$ (semi) Eulerian
DU	P	$\{0, 1, +\infty\}$	P	Theorem 1
DW	P	$\mathbb{N}$	P	$\psi(G)$ (semi) Eulerian

Definition 2 *Let G be a digraph, and $R^+(G)$ be the weighted DAG obtained from $R(G)$, such that the weight of an edge is attributed the number of distinct arcs from two strongly connected components in G.*

Theorem 1 *Let $G = (V, E)$ be an unweighted digraph. G is a 2-sequence graph if and only if $R^+(G)$ is a directed path and its weights are all equal to 1.*

3.2 General case: main complexity results

Table 2. Complexity for various instances of our problems ($w \geq 3$)

Variation	NumRealizations$_w$ Complexity	Realizable$_w$ Complexity	NumRealizations Complexity	Realizable Complexity
GU	P	P	W[1]-hard	W[1]-hard
GW	#P-hard $\forall w \geq 3$	NP-hard $\forall w \geq 3$	#P-hard	NP-hard
DU	Open	Open	W[1]-hard	W[1]-hard
DW	#P-hard	NP-hard	#P-hard	NP-hard

4. Dynamic programming formulation for NumRealizations$_w$

The recursion proceeds by extending a partial sequence, initially set to be empty, keeping track of for represented edges along the way. Namely, consider $N_w[\Pi, p, \mathbf{u}]$ to be the number of w-admissible sequences of length p for the graph $G = (V, E)$, respecting a weight matrix $\Pi = (\pi_{ij})_{i,j \in V^2}$, preceded by a sequence of nodes $\mathbf{u} := (u_1, \ldots, u_{|\mathbf{u}|}) \in V^\star$. It can be shown that, for all $\forall p \geq 1$, $\Pi \in \mathbb{N}^{|V^2|}$ and $\mathbf{u} \in V^{\leq w}$, $N_w[\Pi, p, \mathbf{u}]$ obeys the following formula:

$$N_w\left[\Pi, p, \mathbf{u}\right] = \sum_{v \in V} \begin{cases} N_w\left[\Pi'_{(\mathbf{u},v)}, p-1, (u_1, ..., u_{|u|}, v)\right] & \text{if } |\mathbf{u}| < w-1 \\ N_w\left[\Pi'_{(\mathbf{u},v)}, p-1, (u_2, ..., u_{w-1}, v)\right] & \text{if } |\mathbf{u}| = w-1 \end{cases} \tag{4}$$

with $\Pi'_{(\mathbf{u},v)} := (\pi_{ij} - |\{k \in [1, |\mathbf{u}|] \mid (u_k, v) = (i, j)\}|)_{(i,j) \in V^2}$. The base case of this recurrence corresponds to $p = 0$, and is defined as

$$\forall\, \Pi,\; N_w[\Pi, 0, \mathbf{u}] = \begin{cases} 1 & \text{if } \Pi = (0)_{(i,j) \in V^2} \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The total number of admissible sequences is then found in $N_w[\Pi, p, \varepsilon]$, *i.e.* setting $\mathbf{u}$ to the empty prefix ε, allowing the sequence to start from any node.

The recurrence can be computed in $\mathcal{O}(|V|^w \times \prod_{i,j \in V^2}(\pi_{i,j} + 1))$ time using memoization, for p the sequence length. The complexity can be refined by noting that:

$$\sum_{i,j \in V^2} \pi_{i,j} \leq w \times p$$

It follows that, in the worst-case scenario, $\prod_{i,j \in V^2}(\pi_{i,j} + 1) \in \mathcal{O}(2^{w\,p})$. Thus, it is still possible to compute $N_w[\Pi, p, u_{1:w}]$ for "reasonable" values of p and w such as $p \leq 500$ and $w \leq 10$.

Acknowledgments

We thank Guillaume Fertin for his suggestions and questions which helped to orientate this work in the right direction.

References

[1] Konstantinos Skianis, Fragkiskos Malliaros, and Michalis Vazirgiannis. Fusing document, collection and label graph-based representations with word embeddings for text classification. In *Proceedings of the Twelfth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-12)*, pages 49–58, 2018.

[2] Michael Roth and Kristian Woodsend. Composition of word representations improves semantic role labelling. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 407–413, 2014.

[3] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[4] Jaume Gibert, Ernest Valveny, and Horst Bunke. Dimensionality reduction for graph of words embedding. In *International Workshop on Graph-Based Representations in Pattern Recognition*, pages 22–31. Springer, 2011.

[5] François Rousseau, Emmanouil Kiagias, and Michalis Vazirgiannis. Text categorization as a graph classification problem. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1702–1712, 2015.

[6] Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference*, pages 1063–1072, 2018.

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[8] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[9] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399, 2016.

[10] Arora Sanjeev, Liang Yingyu, and Ma Tengyu. A simple but tough-to-beat baseline for sentence embeddings. *Proceedings of ICLR*, 2017.