Mining Patterns in Data

Implementing Sequence Mining

Gilles Peiffer (23421600)
Université catholique de Louvain
gilles.peiffer@student.uclouvain.be

Liliya Semerikova (64811600)
Université catholique de Louvain
liliya.semerikova@student.uclouvain.be

ABSTRACT

The following paper contains a detailed analysis of the performance and results of using various sequential pattern mining algorithms to fulfill several important tasks in the field of data mining. In a first part of the paper, algorithms existing in the literature are described, which are then compared on different tasks. The second part of the paper looks at the results of these mining tasks and gathers insights based on them.

1. INTRODUCTION

Frequent sequential pattern mining is an active area of research in data mining with broad applications. Finding efficient algorithms for this task is thus majorly important, and with the rise of machine learning techniques such as supervised learning, it is interesting to consider which algorithms are able to combine both tasks well.

Additionally, some large datasets contain a lot of redundant information: consider the database made of the single sequence $\langle (a_1)(a_2)\dots(a_{100})\rangle$. With a minimum support of 1, it will generate $2^{100}-1$ frequent subsequences, all of which are redundant except for the last one because they have the same support. For this reason, we also consider algorithms which perform well when mining closed sequential patterns.

2. TASKS

2.1 Frequent Sequence Mining

The goal of this task is, given two datasets of respectively positive and negative examples, to find the top-k most frequent sequential patterns across both of them. If multiple patterns obtain the same total support, all of them should only count for 1 in the value of k.

2.2 Supervised Sequence Mining

In supervised sequence mining, the aim is still to find top-k most frequent patterns, but with a new scoring function instead of the total support. Let $p(\alpha)$ and $n(\alpha)$ be the support of sequential pattern α in both datasets, and P and N be the number of transactions in each dataset; in that case, the *weighted relative accuracy* is given by

$$\mathrm{WRAcc}(\alpha) \triangleq \frac{PN}{(P+N)^2}\left(\frac{p(\alpha)}{P}-\frac{n(\alpha)}{N}\right). \tag{1}$$

In order to search for frequent patterns efficiently, an upper bounding procedure is necessary. By computing this bound, one can prune the search tree as soon as the bound does not exceed or equal the lowest score found amongst the current top-k sequential patterns. It is easy to see that for a sequential pattern $\beta \sqsupseteq \alpha$, the highest possible WRAcc score that can be attained is bounded by

$$\mathrm{WRAcc}(\beta) \leqslant \frac{Np(\alpha)}{(P+N)^2}, \quad \forall \beta \sqsupseteq \alpha, \tag{2}$$

where the assumption is made that all transactions containing α in the positive dataset also contain β, yet none of the transactions in the negative dataset do.

2.3 Supervised Closed Sequence Mining

For our purposes, we define a closed sequential pattern α as a sequence such that for any sequence $\beta \supsetneq \alpha$, $p(\alpha) > p(\beta)$ or $n(\alpha) > n(\beta)$, that is, no supersequence exists which has the same support in both datasets.

The task of supervised closed sequence mining is applied using three different scoring functions:

- The WRAcc scoring function described earlier.
- The AbsWRAcc scoring function, defined as

  $$\mathrm{AbsWRAcc}(\alpha) \triangleq |\mathrm{WRAcc}(\alpha)|. \tag{3}$$

  Upper bounding can then be done as follows: for a sequential pattern $\beta \sqsupseteq \alpha$, the highest possible AbsWRAcc score that can be attained is bounded by

  $$\mathrm{AbsWRAcc}(\beta) \leqslant \frac{\max\{Np(\alpha), Pn(\alpha)\}}{(P+N)^2}, \ \forall \beta \sqsupseteq \alpha. \tag{4}$$

- The information gain function, suggested by Quinlan [3] (where p and n are used instead of $p(\alpha)$ and $n(\alpha)$, to alleviate notations):

  $$\mathrm{IG}(\alpha) \triangleq \mathrm{imp}\left(\frac{P}{P+N}\right) - \frac{p+n}{P+N}\,\mathrm{imp}\left(\frac{p}{p+n}\right) - \frac{P+N-p-n}{P+N}\,\mathrm{imp}\left(\frac{P-p}{P+N-p-n}\right), \tag{5}$$

  where imp is the entropy, defined as

  $$\mathrm{imp}(x) \triangleq -x\lg x - (1-x)\lg(1-x). \tag{6}$$

  Computing a bound for this scoring function is harder to do analytically. To compute the maximum score

for a sequential pattern $\beta \sqsupseteq \alpha$, one has the following relationship:

$$\mathrm{IG}(\beta) \leqslant \max_{\substack{0 \leqslant \pi \leqslant p(\alpha) \\ 0 \leqslant \nu \leqslant n(\alpha)}} \mathrm{IG}(\pi, \nu), \quad \forall \beta \sqsupseteq \alpha, \tag{7}$$

where we have used another definition of the information gain function directly taking the supports of the sequence as arguments. By precomputing the information gain for all pairs of values, this bound can be computed as a cumulative maximum.

3. ALGORITHMS

Various algorithms and implementations were used to complete the tasks outlined in Section 2.

3.1 PrefixSpan

The PrefixSpan algorithm was proposed by Pei et al. [2, 1] in order to mine sequential patterns using a pattern-growth approach. This algorithm was used for tasks 2.1 and 2.2. For our purposes, two implementations of this algorithm were written, one using a priority queue or heap to store the top-k patterns and one using a sorted list.

The algorithm is ran with a value of k starting from 1 all the way up to the original value, since this strategy allowed for faster pruning on large datasets; for larger values of k, the algorithm would occasionally insert low-scoring patterns in its results list, which would then prevent the algorithm from efficiently pruning its search tree. By using an incremental strategy, we were able to stop this from happening, with minimal overhead on easier datasets.

3.2 SPADE

The SPADE algorithm was first proposed by Zaki [6], to solve the problem of frequent sequence mining. For our survey, it was used to solve task 2.1. It uses a depth-first approach, and is similar to the ECLAT algorithm, also proposed by Zaki [5].

3.3 CloSpan

The CloSpan algorithm is an adaptation of PrefixSpan designed specifically to mine closed sequential patterns proposed by Yan et al. [4], as in task 2.3. In order to do so, the CloSpan algorithm runs in two phases:

- A *search* phase, during which PrefixSpan is run and the following score is computed as $\sum_{(t,p)\in\mathcal{D}|_\alpha} \mathcal{D}[t] - p + 1$, where $\mathcal{D}|_\alpha$ is the α-projected database. For any pattern, we check whether a pattern has already been seen that has the same score while also being a supersequence. If so, we cut the search tree and backtrack.
- A *post-processing* phase, where all patterns which are not closed are removed.

As with PrefixSpan, in order to avoid exploring the search tree too much because of inefficient bounding, the algorithm is run incrementally.

4. PERFORMANCE AND ANALYSIS OF RESULTING PATTERNS

4.1 Frequent Sequence Mining

Figures 2 and 13 show a comparison of execution times for the frequent sequence mining tasks, respectively on the "Reuters" and "Protein" datasets. One can observe that the PrefixSpan implementations are more memory-efficient. This can be explained by the fact that SPADE is more straightforward, whereas PrefixSpan uses a more intelligent pattern-growth principle to generate new candidates. Figures 3 and 14 compare their maximal memory usage. On this figure and on the next ones, the spike for very low values of k is presumably a consequence of other tasks running in parallel, which are very visible due to the short execution time for these tasks.

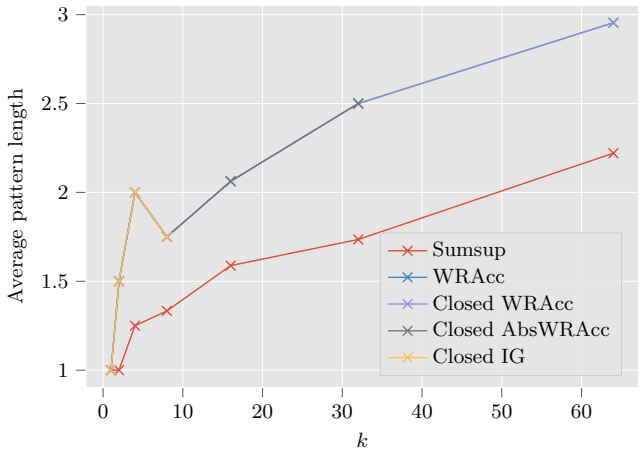

Figure 1: Average pattern length on the "Reuters" dataset, using different scoring functions.

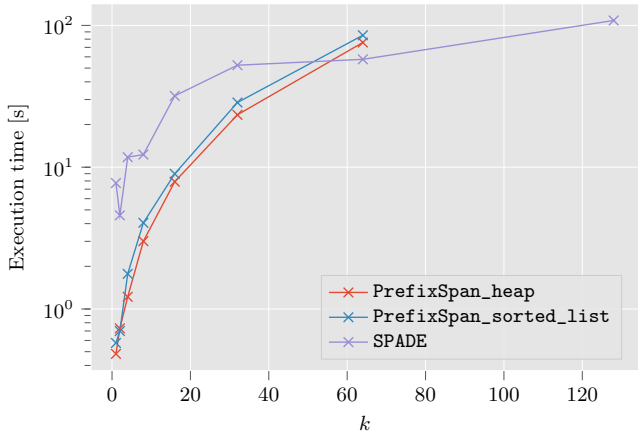

Figure 2: Execution time comparison for the "Reuters" dataset, for the frequent sequence mining task.

By observing the output patterns in this task on Figures 1 and 12, one can see that shorter patterns always obtain higher support scores. While this can be useful, Section 4.2 shows that this is very different from the results obtained when trying to obtain information from the patterns that are being selected.

4.2 Supervised Sequence Mining

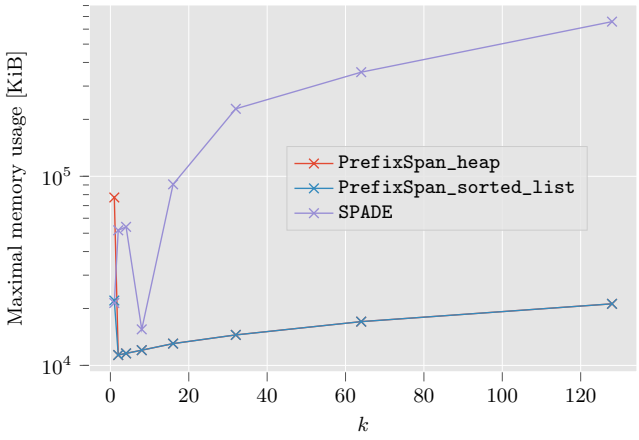

Figure 3: Maximal memory usage comparison for the "Reuters" dataset, for the frequent sequence mining task.

Figures 4 and 15 show a comparison of execution times for the frequent sequence mining tasks, respectively on the "Reuters" and "Protein" datasets. Figures 5 and 16 compare their maximal memory usage. Both implementations of PrefixSpan are nearly identical in both aspects, except for some outliers at lower values of k.

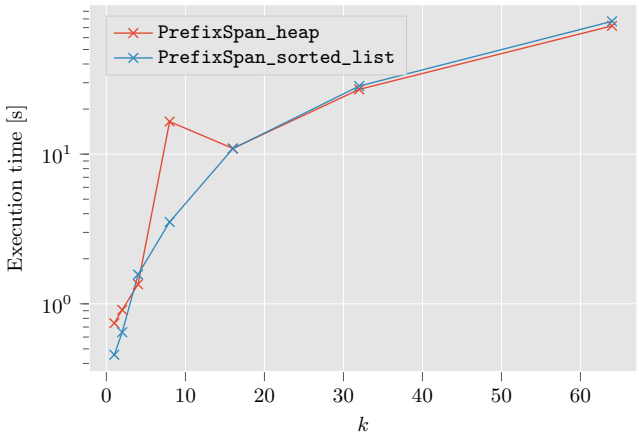

Figure 4: Execution time comparison for the "Reuters" dataset, for the supervised sequence mining task.

Weighted relative accuracy allows one to only keep those sequences which are strongly represented in the positive dataset, but not in the negative dataset. One possible issue with this is that sequences which appear a lot in the negative dataset but not the positive dataset are ignored, despite giving a lot of information. AbsWRAcc and information gain, as explored in Section 4.3, can help solve this problem. When comparing the results with those of Section 4.3, one can also observe a lot of superfluous information is present in the output of the non closed algorithms. By only considering closed patterns, this effect can be reduced, as one gets a lossless representation of the dataset. As expected, the average pattern length is also longer than for the sum of supports scoring function of Section 4.1, which is apparent on Figures 1 and 12.

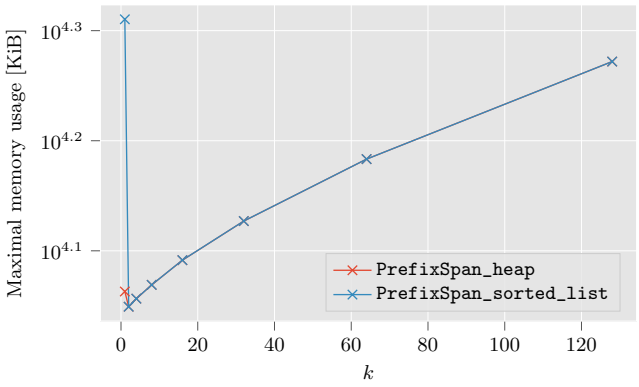

Figure 5: Maximal memory usage comparison for the "Reuters" dataset, for the supervised sequence mining task.

4.3 Supervised Closed Sequence Mining

As mentioned in Section 4.2, closed patterns are a lossless representation of the dataset. Figures 6, 8, and 10 give the execution for the three scoring functions for the supervised closed sequence mining task on the "Reuters" dataset, while Figures 7, 9, and 11 give the maximal memory consumption. Figures 17 through 22 do the same for the "Protein" dataset. As one can see on these figures for the "Reuters" dataset, there seems to be an apparent memory overhead for low values of k when using the implementation with a sorted list, though this difference disappears for higher values of k. The inverse effect appears on the "Protein" dataset, with the heap implementation using more memory. This leads us to believe that certain properties of the dataset might affect the various algorithms differently. Another observation is that for the information gain scoring function on the "Reuters" dataset, which requires a lot of precomputation and is thus slower to compute even for small values of k, the spike which could be seen on other memory usage figures is gone, which corroborates our belief that this spike is due to noise on the measurements as a consequence of low execution time, rather than an intrinsic property of the algorithms. On the figures for closed sequence mining with the

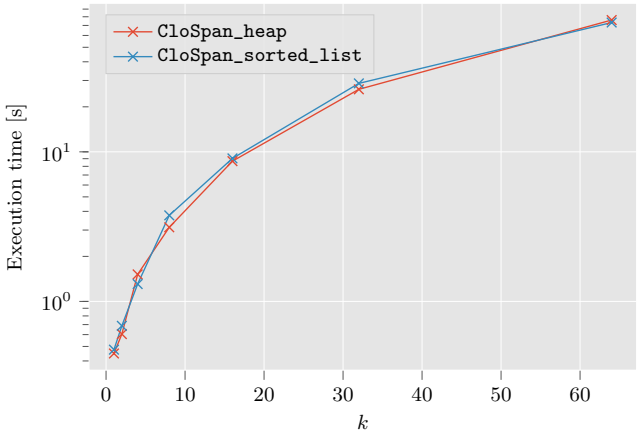

Figure 6: Execution time comparison for the "Reuters" dataset, for the supervised closed sequence mining task with the WRAcc scoring function.

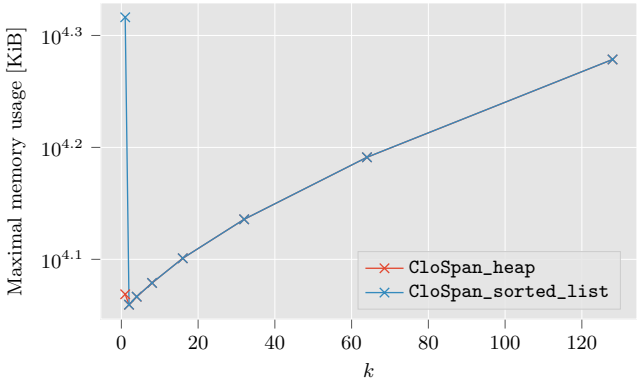

Figure 7: Maximal memory usage comparison for the "Reuters" dataset, for the supervised closed sequence mining task with the WRAcc scoring function.

WRAcc scoring function, one can observe that using closed sequences is slightly faster than not doing so, which might be due to the additional pruning of the search tree as described in Section 3.3. The patterns that are obtained are a subset of the ones obtained previously, when mining non closed sequences as well.

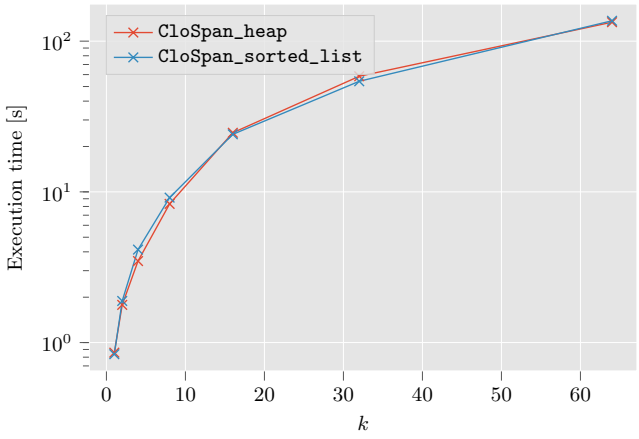

Figure 8: Execution time comparison for the "Reuters" dataset, for the supervised closed sequence mining task with the AbsWRAcc scoring function.

Performance-wise, the AbsWRAcc scoring function does not make a big difference compared to the original WRAcc function, but it is a much better guide in determining whether a given pattern brings a lot of information with regards to either class in the dataset.

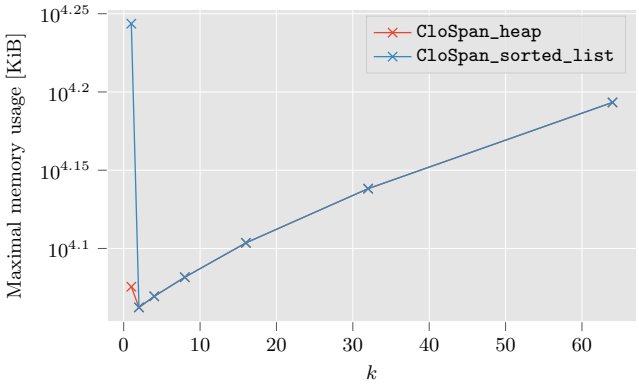

Figure 9: Maximal memory usage comparison for the "Reuters" dataset, for the supervised closed sequence mining task with the AbsWRAcc scoring function.

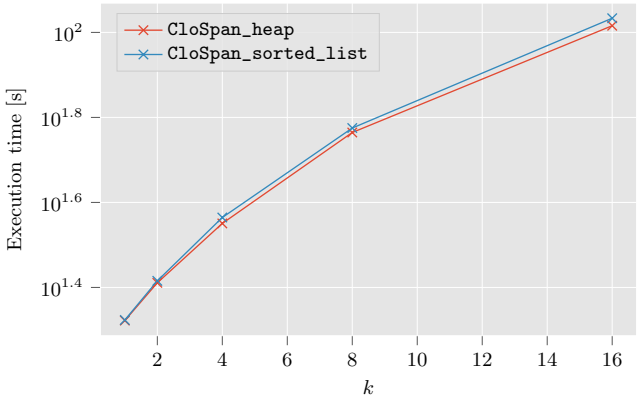

Figure 10: Execution time comparison for the "Reuters" dataset, for the supervised closed sequence mining task with the information gain scoring function.

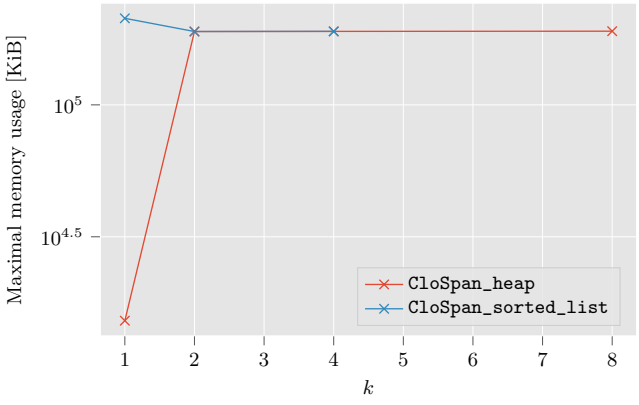

Figure 11: Maximal memory usage comparison for the "Reuters" dataset, for the supervised closed sequence mining task with the information gain scoring function.

As with the AbsWRAcc scoring function, the information gain is a way of measuring the quality of a pattern in a supervised environment. However, it is often slower, due to its weaker upper-bound (depsite the latter being optimal).

Often, both functions give similar results, though this is not always the case. Choosing which scoring function to use thus depends on problem-specific properties.

5. CONCLUSION

Sequence mining, whether supervised or not, is a hugely important task, with broad applications. By combining it with supervised learning, it becomes even more useful, as sequence mining can be used to obtain information about multiple datasets, and to find defining features of each, according to various scoring functions. Closed sequence mining is useful in that it provides the user with a lossless representation of the dataset.

References

[1] Jian Pei, Jiawei Han, B. Mortazavi-Asl, Jianyong Wang, H. Pinto, Qiming Chen, U. Dayal, and Mei-Chun Hsu. Mining sequential patterns by pattern-growth: the PrefixSpan approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1424–1440, 2004.

[2] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. PrefixSpan: Mining Sequential Patterns by Prefix-Projected Growth. In *Proceedings of the 17th International Conference on Data Engineering*, pages 215–224, USA. IEEE Computer Society, 2001. ISBN: 0769510019.

[3] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986. ISSN: 1573-0565. DOI: 10.1007/BF00116251. URL: https://doi.org/10.1007/BF00116251.

[4] X. Yan, J. Han, and R. Afshar. CloSpan: Mining: Closed Sequential Patterns in Large Datasets. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, pages 166–177. DOI: 10.1137/1.9781611972733.15. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611972733.15. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611972733.15.

[5] M. J. Zaki. Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, 12(3):372–390, 2000.

[6] M. J. Zaki. SPADE: An Efficient Algorithm for Mining Frequent Sequences. *Machine Learning*, 42(1):31–60, 2001. ISSN: 1573-0565. DOI: 10.1023/A:1007652502315. URL: https://doi.org/10.1023/A:1007652502315.

APPENDIX

A. OMITTED FIGURES

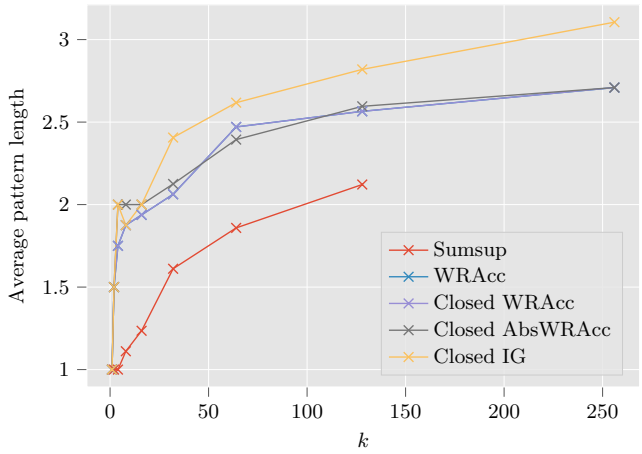

Figure 12: Average pattern length on the "Protein" dataset, using different scoring functions.

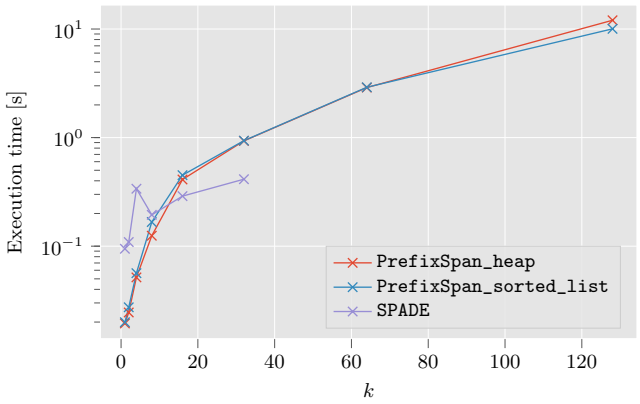

Figure 13: Execution time comparison for the "Protein" dataset, for the frequent sequence mining task.

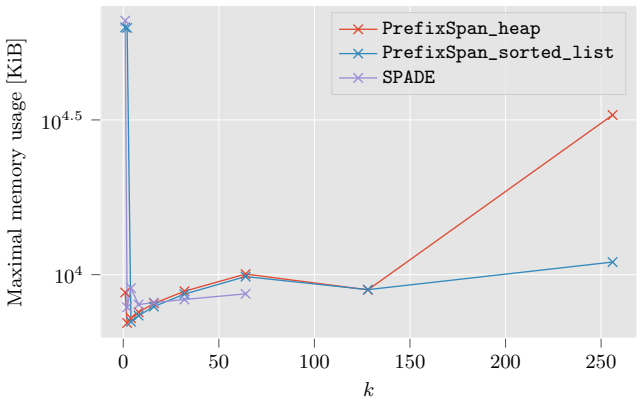

Figure 14: Memory usage comparison for the "Protein" dataset, for the frequent sequence mining task.

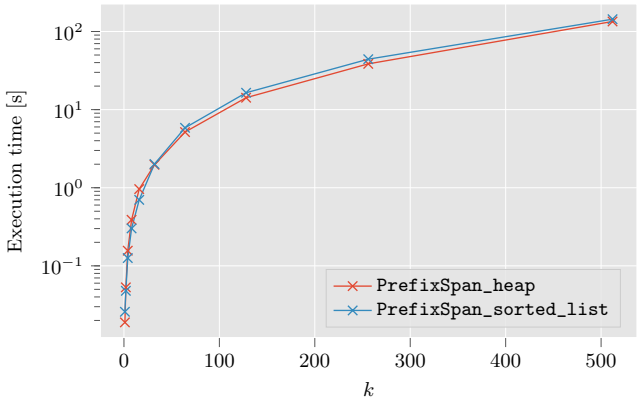

Figure 15: Execution time comparison for the "Protein" dataset, for the supervised sequence mining task.

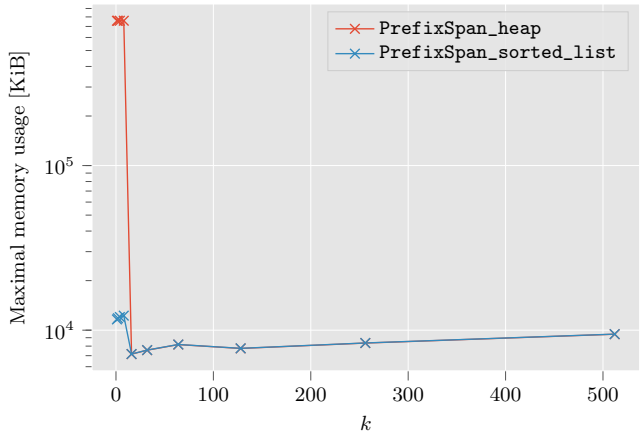

Figure 16: Memory usage comparison for the "Protein" dataset, for the supervised sequence mining task.

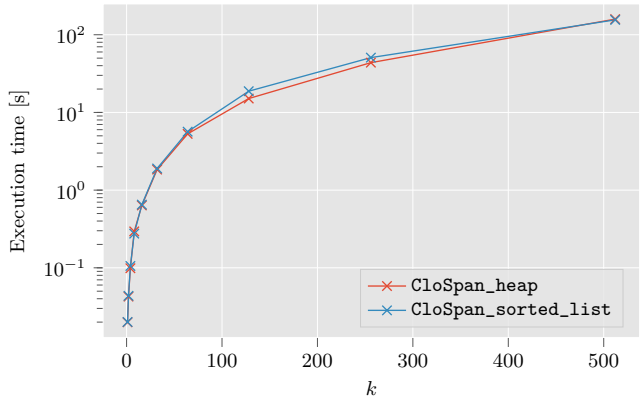

Figure 17: Execution time comparison for the "Protein" dataset, for the supervised closed sequence mining task with the WRAcc scoring function.

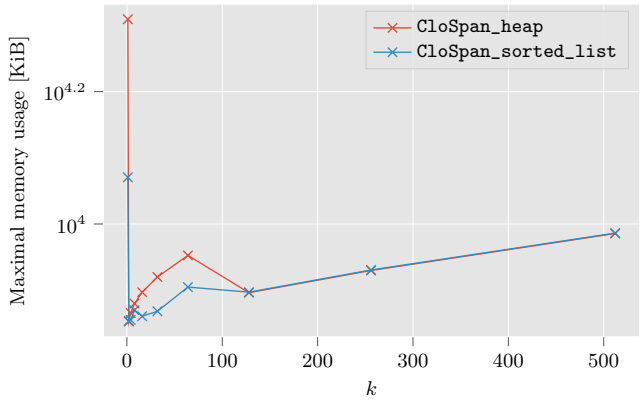

Figure 18: Maximal memory usage comparison for the "Protein" dataset, for the supervised closed sequence mining task with the WRAcc scoring function.

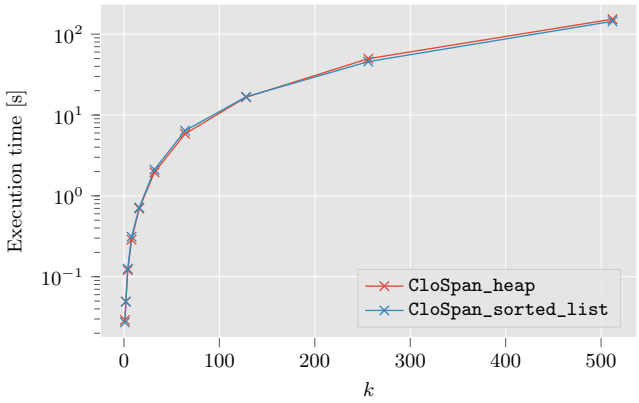

Figure 19: Execution time comparison for the "Protein" dataset, for the supervised closed sequence mining task with the AbsWRAcc scoring function.

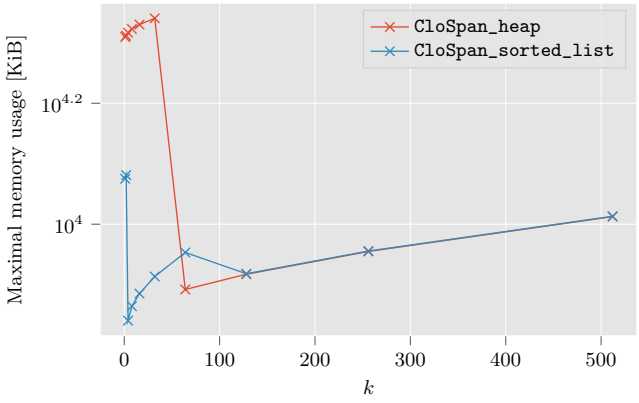

Figure 20: Maximal memory usage comparison for the "Protein" dataset, for the supervised closed sequence mining task with the AbsWRAcc scoring function.

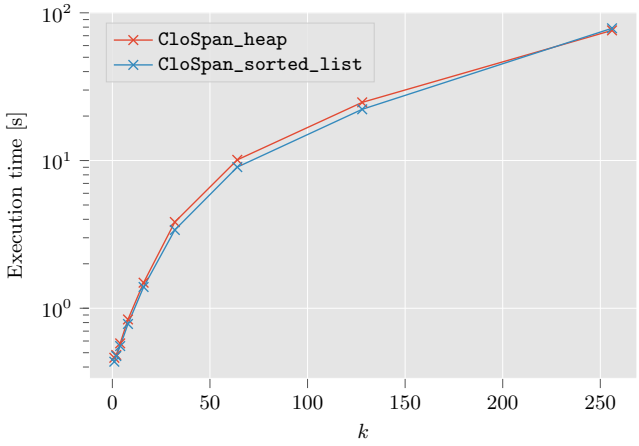

Figure 21: Execution time comparison for the "Protein" dataset, for the supervised closed sequence mining task with the information gain scoring function.

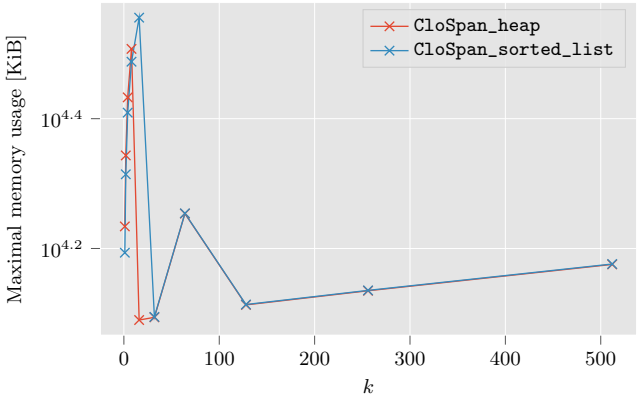

Figure 22: Maximal memory usage comparison for the "Protein" dataset, for the supervised closed sequence mining task with the information gain scoring function.