Clustering Partial Lexicographic Preference Trees (Student Abstract)

Joseph Allen, Xudong Liu, Karthikeyan Umapathy, Sandeep Reddivari
School of Computing, University of North Florida
1 UNF Dr., Jacksonville, FL 32224
{n01045721, xudong.liu, k.umapathy, sandeep.reddivari}@unf.edu

Abstract

In this work, we consider distance-based clustering of partial lexicographic preference trees (PLP-trees), intuitive and compact graphical representations of user preferences over multi-valued attributes. To compute distances between PLP-trees, we propose a polynomial time algorithm that computes Kendall's τ distance directly from the trees and show its efficacy compared to the brute-force algorithm. To this end, we implement several clustering methods (i.e., spectral clustering, affinity propagation, and agglomerative nesting) augmented by our distance algorithm, experiment with clustering of up to 10,000 PLP-trees, and show the effectiveness of the clustering methods and visualizations of their results.

Introduction

Understanding the decision patterns of users and modeling their preferences is an important problem in artificial intelligence and has received much attention in recent years. A myriad of models for the task of preference representation have been proposed, at the frontier of which are partial lexicographic preference trees (PLP-trees, for short) (Liu and Truszczynski 2015), and conditional preference networks (CP-nets). PLP-trees are particularly interesting, for they succinctly encode a total preorder (i.e., a binary relation that is total, reflexive, and transitive) of exponentially many alternatives into the structure of an often compact tree. CP-nets also provide an intuitive encoding with the *ceteris paribus* semantics, but they generally are computationally hard to learn and reason about. For this reason we focus on PLP-trees, which show great potential as effective and explainable models.

PLP-trees are useful for modeling individual preferences. Meanwhile, being able to cluster and reason about collections of preferences is critical to various research areas, such as recommender systems, marketing, and human- centric machine learning. Clustering a set of entities typically relies on computing distances between them. Examples of distance-based metrics are Euclidean distance, Spearman's ρ, and Kendall's τ. Because PLP-trees induce *total preorders* over alternatives, we focus on Kendall's τ.

Clearly, one may use the straightforward brute-force algorithm to compute τ; however, due to the exponentiality of the space of alternatives, this direct method practically is infeasible. Recently, Li and Kazimipour (Li and Kazimipour 2018) proposed a computationally efficient algorithm, called *LpDis*, that utilizes the tree structures to compute τ in time polynomial in the size of the two given trees.

However, their results are limited to *complete* models, trees that require *every* attribute to be present in every branch. On the contrary, PLP-trees allow for missing attributes, thus encoding total preorders, as opposed to total orders, and allow for more concise preference representations. To this end, we study the clustering problem of PLP-trees. At the core of this problem, we investigate the problem of computing the τ distance and propose a new polynomial time algorithm *PlpDis*, an extension to the *LpDis* algorithm. In the remainder of this paper, we provide necessary preliminaries, define a variant of Kendall's τ for PLP- trees and describe our *PlpDis* algorithm that accommodates partial trees, and demonstrate PLP-tree clustering using *PlpDis*.

Preliminaries

Let $\mathcal{V} = \{X_1, ..., X_n\}$ be a set of n categorical *attributes* with each $X_i \in \mathcal{V}$ having a finite domain of values $D_i = \{x_1^i, \ldots, x_{m_i}^i\}$. The *combinatorial domain* $CD(\mathcal{V})$ over $\mathcal{V}$ is the Cartesian product $D_1 \times \ldots \times D_n$. We call elements of $CD(\mathcal{V})$ *alternatives*. A *PLP-tree* over $\mathcal{V}$ is an ordered labeled tree where every non-leaf node 1) is labeled with an attribute X_i from $\mathcal{V}$, 2) has a local preference $>_i$ (a total order over D_i), and 3) has $|D_i|$ outgoing edges ordered from left to right according to $>_i$. Additionally, each attribute from $\mathcal{V}$ appears *at most once* in any branch of the tree. The leaf nodes in the tree indicate buckets of alternatives.

We reason about the distance between PLP-trees by considering the *disagreements* between the orders they represent. Let T_1 and T_2 be two PLP-trees (two total preorders), and α and β two distinct alternatives from $CD(\mathcal{V})$. The ordering of α and β on T_1 and T_2 either strictly agree, strictly disagree, or partially disagree. Strict agreement occurs when in both T_1 and T_2 either $\alpha \succ \beta$, $\beta \succ \alpha$, or $\alpha \approx \beta$, where $\succ$ means strict preference and $\approx$ means equivalence. Strict disagreement occurs when in T_1 we have $\alpha \succ \beta$ and in T_2 $\beta \succ \alpha$, or vice versa. Partial disagreement occurs when in T_1 $\alpha \approx \beta$ and in T_2 $\alpha \not\approx \beta$ ($\alpha \succ \beta$ or $\beta \succ \alpha$), or vice versa.

Distance Between PLP-Trees

We see that the partiality of PLP-trees introduces partial disagreements, which are not accounted for by Kendall's τ or *LpDis*. Thus, we first define a variant of Kendall's τ called *partial* Kendall's τ, denoted τ'. Let SD_{T_1,T_2}, and PD_{T_1,T_2} be the set of alternative pairs on which T_1 and T_2 strictly disagree and partially disagree, respectively. τ' is then:

$$\tau'(T_1, T_2) = |SD_{T_1,T_2}| + c * |PD_{T_1,T_2}|, \quad (1)$$

where $c \in [0, 1]$ is an adjustable constant coefficient that adjusts the weight of partial disagreements. For our algorithm we set $c = 0.5$, however finding a true weight for partial disagreements w.r.t strict disagreements is an interesting problem for future work. As the computation of $|SD_{T_1,T_2}|$ is originated by Li and Kazimipour (Li and Kazimipour 2018), we refer to their paper for it.

We now present the equation for computing $|PD_{T_1,T_2}|$, the novel contribution in our *PlpDis* algorithm. To compute $|PD_{T_1,T_2}|$, we extend the *LpDis* traversal algorithm down to the leaves in both trees and compute the number of alternative pairs encoded both at a leaf node in one tree and at a non-leaf node in the other, denoted $|P_n \cap P_{n'}|$. Thus, we have $|PD_{T_1,T_2}| = \sum_{n \in T_1, n' \in T_2} |P_n \cap P_{n'}|$, where exactly one of n and n' is a leaf node. We consider the distinct pairs of non-leaf and leaf nodes since they encode strict and partial disagreements, respectively. For all distinct pairs of n and n' from both trees (w.l.o.g., n is a leaf), if the attribute labeling n', $V_{n'}$, is not an ancestor of n and the branching attributes in both trees are *consistent*, we have:

$$|P_n \cap P_{n'}| = \binom{|D_{V_{n'}}|}{2} \times \prod_{X \in \breve{A} \setminus \breve{B}} |D_X| \times \prod_{Y \in \mathcal{V} \setminus (\breve{A} \cup \{V_{n'}\})} |D_Y|^2 \quad (2)$$

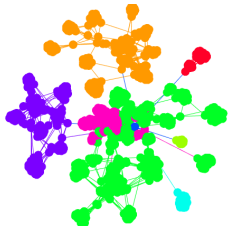
where $\breve{A}$ is the union of ancestor attributes for the nodes and $\breve{B}$ is the set of attributes in $\breve{A}$ that branch out explicitly. The first term computes the number of distinct pairs of values in the domain of $V_{n'}$ and the other two terms adjust for possible values of ancestor and descendant (including missing) attributes, respectively. Clearly, our algorithm *PlpDis* to compute $\tau'(T_1, T_2)$ runs in time polynomial in the size of T_1 and T_2. This follows from the fact that in the worst case, we compare every pair of nodes between the two trees.

Clustering PLP-Trees

We begin clustering by first computing a pairwise distance matrix for a collection of PLP-trees. Then we can use any off-the-shelf clustering algorithm that takes the distance matrix as input. For our experiments, we selected spectral clustering, affinity propagation, and agglomerative nesting (AGNES), the latter of which has three variants: average, single, and complete linkage. To validate the clustering results, we used the Dunn Index (DI) and Davies- Bouldin Index (DBI), which both measure the extent to which a set of clusters minimize overall intra-cluster distances while also maximizing inter-cluster distances. Specifically, since higher is better for DI and lower is better for DBI, we use a ratio of DI/DBI. In addition to the quantitative measures, we qualitatively assess cluster quality by constructing KNN-graphs (with $K = 10$) from the distance matrix and coloring vertices according to each algorithm's cluster assignments.

Forest Size	1000	10,000
# Clusters	8	24
Spectral	0.387	DNF
Aff. Prop.	0.342	0.238
AGNES-A	2.172	1.323
AGNES-S	2.172	**2.034**
AGNES-C	**2.245**	0.886

(a) DI/DBI results



(b) 1k trees clustered by AGNES-C

Figure 1: Selected clustering results and visualization

To test *PlpDis*-powered clustering, we applied the selected algorithms to the task of clustering PLP-forests learned from a Car Evaluation dataset[1]using Liu and Truszczynski's greedy algorithm (Liu and Truszczynski 2019). In the experiment, we learned forests of sizes 100, 500, 1,000, 2,500, 5,000, and 10,000 with each tree learning from 100 examples. Note that since affinity propagation finds the number of clusters k itself, we run it first and provide the k it finds to the other algorithms.

The results of these tests for the forests of size 1,000 and 10,000 are included in Figure 1a, where the best score is bolded in each column. Note that "DNF" in the table for spectral clustering indicates that the algorithm did not finish within a set timeout threshold of 20 minutes. In Figure 1b we show the KNN-graph from the test with 1,000 trees for *complete-linkage* AGNES, which performed the best in this test. Overall, all three variants of AGNES outperform spectral clustering and affinity propagation. AGNES consistently detects the highly-connected (similar) vertices in the KNN-graphs, but the other methods do not. This trend continues for all tested forest sizes, with *single-linkage* taking the lead for the larger forests with 2,500 or more trees.

Conclusion and Future Work

In this work, we proposed *PlpDis* algorithm to measure distance between PLP-trees, and integrated it with several existing clustering methods. Our experimental and visual results show effectiveness of our algorithm. For future directions, we plan to improve the scalability of our implementation for clustering much larger numbers of models. We also intend to explore other preferential datasets as well as interactive cluster visualization techniques.

References

Li, M.; and Kazimipour, B. 2018. An Efficient Algorithm To Compute Distance Between Lexicographic Preference Trees. In *IJCAI*, 1898–1904.

Liu, X.; and Truszczynski, M. 2015. Learning Partial Lexicographic Preference Trees over Combinatorial Domains. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*, 1539–1545. AAAI Press.

Liu, X.; and Truszczynski, M. 2019. Voting-based Ensemble Learning for Partial Lexicographic Preference Forests over Combinatorial Domains. *Annals of Mathematics and Artificial Intelligence* 87: 137–155.

[1] www.unf.edu/~N01237497/preflearnlib.html