Neural Network Approaches for Learning Permutation Matrices in Column Subset Selection Problems

Hanish Dhanwalkar
Department of Computer Science
Indian Institute of Technology Bombay
Mumbai, India
210100060@iitb.ac.in

Harsh Kavediya
Department of Computer Science
Indian Institute of Technology Bombay
Mumbai, India
210100067@iitb.ac.in

Vignesh Nayak
Department of Computer Science
Indian Institute of Technology Bombay
Mumbai, India
210100169@iitb.ac.in

Abstract

This research proposes a novel approach to column subset selection problems by implementing continuous approximations of discrete steps in selection algorithms. We focus on the Linear Time Approximation Algorithm with Local Search, identifying discrete operations and developing differentiable alternatives to facilitate backpropagation in neural networks. Our work bridges the gap between combinatorial optimization techniques and gradient-based learning by developing continuous relaxations for key discrete operations including sampling, set membership updates, and matrix operations. Additionally, we introduce an Enhanced-LSCSS algorithm that improves upon the original algorithm through adaptive sampling, diagonal perturbation, and a two-phase local search approach, reducing the computational complexity from $O(ndk^4 \log k)$ to $O(ndk^3 \log k)$ while achieving a tighter approximation ratio. We evaluate our approach on the MNIST dataset and analyze the trade-offs between approximation quality and computational efficiency. This research contributes to the growing field of differentiable programming for combinatorial optimization problems and offers insights into learning permutation matrices with neural networks.

1 Introduction

Column Subset Selection Problems (CSSP) represent an important class of dimensionality reduction techniques that aim to select the most representative columns from a data matrix. Traditional approaches to CSSP rely on discrete algorithms that involve operations such as discrete sampling, set membership tests, and combinatorial optimization. While these methods provide strong theoretical guarantees, they present challenges for integration with modern deep learning frameworks due to their non-differentiable nature.

Recent advances in differentiable programming have opened new possibilities for solving combinatorial optimization problems within neural network architectures. By replacing discrete operations with continuous approximations, it becomes possible to leverage gradient-based optimization techniques while maintaining the structural properties of the original algorithms.

Preprint. Under review.

In this work, we focus on developing continuous approximations to the discrete steps in the Linear Time Approximation Algorithm for Column Subset Selection with Local Search. Our goal is to enable end-to-end training of neural networks for learning permutation matrices that represent optimal column subsets.

2 Problem Statement

2.1 Column Subset Selection Problem

Given a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ and an integer $k < n$, the Column Subset Selection Problem aims to find a subset of k columns from $\mathbf{A}$ that best represents the entire matrix. This can be formulated as finding a permutation matrix $\mathbf{P} \in 0, 1^{n \times n}$ such that the first k columns of $\mathbf{AP}$ minimize the reconstruction error. Formally, we seek to minimize:

$$|\mathbf{A} - \mathbf{C}(\mathbf{C}^T\mathbf{C})^{-1}\mathbf{C}^T\mathbf{A}|_F^2 \tag{1}$$

where $\mathbf{C}$ represents the selected k columns of $\mathbf{A}$ and $|\cdot|_F$ denotes the Frobenius norm.

2.2 Challenges in Differentiable Approximation

The core challenge in developing a neural network approach to Column Subset Selection Problem (CSSP) lies in the discrete nature of the permutation matrices. Key operations in CSSP algorithms include discrete sampling of column indices, binary set membership operations - checking if a particular element belongs to a specific set, discrete matrix updates based on selected columns, and finding minimum or maximum indices from the matrix. None of these operations are naturally differentiable, making gradient-based optimization challenging. Our work focuses on developing continuous relaxations for each of these operations while preserving their essential properties to enable effective gradient flow through the entire computational graph.

3 Related Work

Column subset selection (CSS) has been a focal point in matrix approximation and dimensionality reduction, with various methodologies proposed to tackle its computational challenges. The following works have significantly contributed to the development of both deterministic and randomized approaches, as well as differentiable methods pertinent to CSSP.

Deterministic Approaches to CSSP Farahat et al. Farahat et al. [2013] introduced a fast greedy algorithm for generalized column subset selection. Their method focuses on selecting a subset of columns from a source matrix that best approximates the span of a target matrix. By employing a recursive formula to calculate the reconstruction error, the algorithm efficiently identifies representative columns, making it suitable for large-scale data sets.

Randomized Methods for CSSP Boutsidis et al. Boutsidis et al. [2009] proposed an improved approximation algorithm for the CSSP that combines randomized and deterministic techniques. The algorithm operates in two stages: initially, it randomly selects a set of columns based on a probability distribution influenced by the top-k right singular subspace of the matrix. Subsequently, a deterministic selection procedure refines this set to exactly k columns. This hybrid approach balances computational efficiency with approximation quality.

Differentiable Approaches to Permutations and Sorting Mena et al. Mena et al. [2018] presented the Gumbel-Sinkhorn networks, a method for learning latent permutations through continuous relaxations. By approximating discrete maximum-weight matching using the continuous Sinkhorn operator, their approach enables end-to-end learning in models where exact marginalization over permutations is intractable. This method has demonstrated effectiveness across various tasks, including sorting numbers and solving jigsaw puzzles.

Differentiable Ranking via Optimal Transport Cuturi et al. Marco Cuturi and Vert [2019] proposed a framework for differentiable ranking and sorting using optimal transport. Recognizing that

traditional sorting operations are non-differentiable, they introduced a differentiable proxy by formulating sorting as an optimal assignment problem. By regularizing the optimal transport problem with an entropic penalty and solving it using Sinkhorn iterations, they developed smooth approximations for ranking and sorting operations, facilitating their integration into gradient-based learning pipelines.

Differentiable Ranking Using Continuous Relaxations Adams and Zemel Adams and Zemel [2011] developed differentiable ranking methods by employing continuous relaxations of sorting operations. Their approach allows for the approximation of discrete ordering operations within a gradient-based framework, enabling the incorporation of ranking objectives into neural network training processes.

Our Contribution Building upon these foundational works, our research specifically addresses the challenges associated with the Linear Time Approximation Algorithm for CSSP. By integrating differentiable selection mechanisms inspired by the aforementioned methods, we aim to enhance the efficiency and scalability of CSSP solutions in large-scale and real-time applications.

4 Proposed Approach

4.1 Algorithm Analysis

We begin by analyzing the Linear Time Approximation Algorithm for Column Subset Selection with Local Search, identifying the discrete steps that require continuous approximations. Our analysis focuses on two key algorithms: LSCSS (Algorithm 1) and LS (Algorithm 2).

In Algorithm 1 (LSCSS), we identify four primary non-differentiable operations that present challenges for gradient-based optimization. The first is the discrete sampling of column indices, where a column index i is selected with probability proportional to its squared norm. This operation involves a discontinuous selection mechanism that impedes gradient flow. The second challenge involves set membership updates, specifically the discrete action of adding element i to set I, which creates a binary state change not amenable to gradual optimization. The third challenge concerns matrix updates based on set I, particularly the creation of a zeros matrix D with specific diagonal entries determined by set membership, introducing discontinuities in the computational graph. Finally, set operations such as emptying set I represent abrupt state changes that lack natural differentiable counterparts.

Algorithm 2 (LS) presents additional discrete operations that require differentiable approximations. The algorithm involves sampling $10k$ column indices in a discrete manner, followed by uniform sampling for the selection of individual indices. Set membership tests, which determine whether indices exist in set I, introduce binary decision boundaries that block gradient propagation. The algorithm also relies on finding minimum indices through discrete selection operations and performs various set operations including removing and adding elements to set I, all of which lack inherent differentiability.

4.2 Continuous Approximations

For each discrete step identified in our algorithm analysis, we develop a differentiable approximation that preserves the essential properties while enabling gradient flow through the computational graph. Our approach systematically addresses each category of discrete operations with carefully designed continuous counterparts.

4.2.1 Sampling Approximations

Discrete sampling operations are central to the column selection process but present significant challenges for gradient-based optimization. We address these challenges through a comprehensive set of continuous relaxations. For the discrete sampling of column indices, we implement soft sampling that replaces the hard selection with a differentiable mechanism using the Gumbel-Softmax trick. This approach introduces controlled noise into the sampling process while maintaining differentiability. We construct probability vectors based on column norms and apply temperature-controlled softmax operations to approximate the discrete sampling distribution. The temperature parameter allows us to control the sharpness of the approximation, gradually transitioning from exploration to exploitation during training.

For cases requiring top-k selection, we implement a differentiable approximation via softmax normalization with temperature scaling. This approach produces a continuous relaxation of the top-k operation by assigning higher weights to the k columns with the largest values while maintaining non-zero gradients for all columns. To address uniform sampling requirements, we replace discrete uniform selection with a continuous relaxation that maintains equal probabilities across all candidates but allows for soft selection through appropriate temperature-controlled normalization.

4.2.2 Set Membership Approximations

Set operations present particular challenges for differentiable approximations due to their inherently discrete nature. To address these challenges, we replace discrete sets with continuous membership representations. Instead of a binary set I, we maintain a continuous membership vector where each element has values between 0 and 1, representing the degree to which each column belongs to the selected subset. We employ sigmoid functions with learnable parameters to approximate membership indicators, allowing for gradual transitions between non-membership and full membership states.

For membership testing operations, we develop continuous membership functions that replace discrete tests with differentiable measures of "degree of membership." These functions provide smooth approximations to the binary membership tests while preserving the essential behavior of the original discrete operations. The continuous nature of these approximations enables gradient flow through the membership operations, allowing the network to learn optimal membership patterns through backpropagation.

4.2.3 Matrix Operation Approximations

Matrix operations in the original algorithms often rely on discrete column selections, presenting challenges for gradient-based optimization. To address these challenges, we develop continuous approximations of the matrix operations that maintain differentiability while preserving the essential computational structure. We replace discrete selection matrices with soft selection matrices using the continuous membership weights derived from our set membership approximations. This approach allows for weighted contributions from all columns, with weights determined by the continuous membership values.

The zeros matrix D with specific diagonal entries based on set membership is approximated with a continuous function of the membership vector. This approximation allows for smooth transitions in the matrix values as the membership weights change during optimization, enabling gradient flow through these matrix operations while preserving the computational intent of the original discrete operations.

4.2.4 Optimization Operation Approximations

Optimization operations in the original algorithms, such as finding minimum indices, are inherently discrete and present challenges for gradient-based learning. We address these challenges through carefully designed continuous approximations. The discrete argmin operation is replaced with a differentiable soft-argmin function that produces a probability distribution over indices, with higher probabilities assigned to indices with lower values. We implement this approximation using negative temperature-scaled softmax operations, where the temperature parameter controls the sharpness of the approximation.

For discrete set updates involving the addition or removal of elements, we develop continuous approximations based on element-wise operations on the membership vectors. These operations allow for gradual updates to the membership values rather than abrupt changes, enabling gradient flow through the update operations. The continuous nature of these approximations allows the network to learn optimal update patterns through backpropagation while maintaining the essential behavior of the original discrete operations.

4.3 Neural Network Architecture

We propose a comprehensive neural network architecture that integrates a continuous approximation of the Linear Time Approximation Algorithm for Column Subset Selection Problem (LSCSS). This architecture is composed of two primary components: the column selection module and the downstream classifier. The design facilitates end-to-end differentiable learning, enabling joint optimization of feature selection and classification tasks.

4.3.1 Column Selection Module

The column selection module is engineered to emulate the LSCSS algorithm within a differentiable framework. It begins by accepting an input data matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, where m denotes the number of samples and n represents the number of features. To assess the significance of each column, the module computes the L2-norm of each column in a differentiable manner, ensuring compatibility with gradient-based optimization techniques.

To enable stochastic selection of columns while maintaining differentiability, the module employs the Gumbel-Softmax sampling technique. This approach introduces Gumbel noise to the log-probabilities of column selection, followed by a softmax operation parameterized by a temperature τ. The temperature τ is annealed during training to encourage the model to make more definitive selections as training progresses. This process yields a continuous membership vector $\mathbf{m} \in [0, 1]^n$, which represents the probability of each column being selected.

Subsequently, the module projects the selection decisions onto valid column subsets, ensuring that the selected columns form a representative subset of the original matrix. To refine the selection further, a local search mechanism is incorporated. This mechanism iteratively updates the membership vector over T iterations, guided by a threshold parameter α, to converge towards an optimal subset of columns.

4.3.2 Downstream Classifier

The downstream classifier processes the features selected by the column selection module to perform classification tasks, such as digit recognition. The classifier architecture comprises two fully connected hidden layers. The first hidden layer consists of 128 units, while the second comprises 64 units; both layers utilize the Rectified Linear Unit (ReLU) activation function to introduce non-linearity. To mitigate overfitting and enhance generalization, dropout regularization with a rate of 0.3 is applied after each hidden layer. The final output layer contains 10 units corresponding to the number of classes, employing the softmax activation function to produce probability distributions over the classes.

4.3.3 End-to-End Architecture

The integration of the column selection module and the downstream classifier forms an end-to-end trainable architecture. The model is trained using the cross-entropy loss function, appropriate for multi-class classification problems. Optimization is carried out using the Adam optimizer with an initial learning rate of 0.001. To adapt the learning rate dynamically based on validation performance, a ReduceLROnPlateau scheduler is employed, which reduces the learning rate by a factor of 0.5 if the validation loss does not improve for three consecutive epochs.

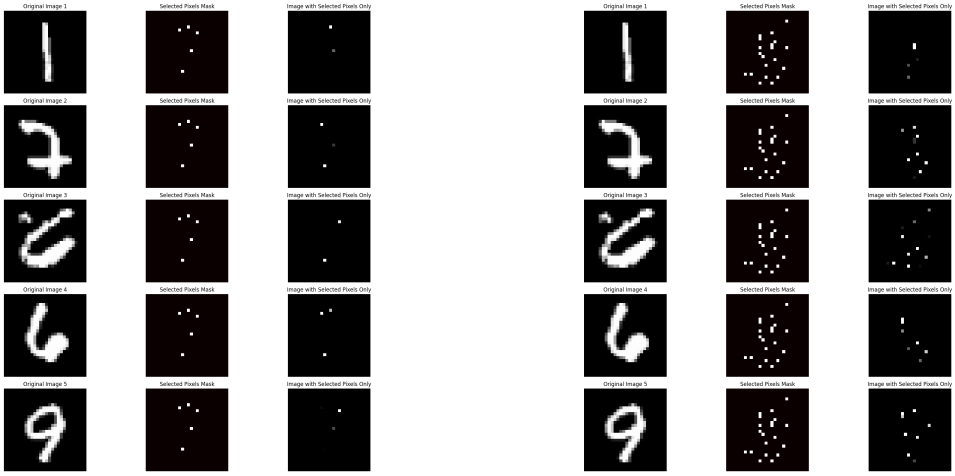
To facilitate the transition from exploration to exploitation in the selection process, the temperature parameter τ in the Gumbel-Softmax sampling is annealed exponentially by a factor of 0.95 per epoch. This annealing schedule encourages the model to make more confident selections as training advances. The training strategy is divided into two phases: an initial phase focusing on learning effective column selections, followed by a fine-tuning phase where the entire network, including the classifier, is trained jointly to optimize overall performance.

5 Experimental Evaluation

5.1 Dataset and Preprocessing

To evaluate the effectiveness of our proposed feature selection approach, we conducted experiments on the MNIST dataset of handwritten digits. The dataset comprises a total of 70,000 grayscale images of digits from 0 to 9, divided into 60,000 training samples and 10,000 test samples. Each image is of size 28×28 pixels, resulting in a 784-dimensional feature vector when flattened.

Prior to model training, all images were flattened into one-dimensional vectors of length 784. The pixel intensities were normalized to the range $[0, 1]$ to ensure numerical stability and facilitate gradient-based optimization. We adhered to the standard dataset split without introducing any form of data augmentation. This design choice was intentional to isolate and highlight the contributions of the feature selection mechanism, without confounding effects from additional preprocessing enhancements.



(a) $k = 5$ (b) $k = 20$

(c) $k = 30$ (d) $k = 50$

Figure 1: Pixels selected for different k values

5.2 Experimental Results

5.2.1 Classification Performance

We evaluated the classification performance of our model under varying levels of feature compression, ranging from 5 to 100 selected features, and compared the results with the baseline that uses all 784 features. Table 1 presents the classification accuracy, the proportion of retained accuracy relative to the full model, and the corresponding compression ratio.

The results demonstrate that even with a significantly reduced feature set, our model maintains competitive classification accuracy. For instance, with only 50 selected features (approximately 6.38% of the original), the model achieves 87.85% accuracy—equivalent to 93.96% of the accuracy obtained with all features. This indicates the effectiveness of the selection module in identifying the most informative features. As expected, increasing the number of selected features gradually improves accuracy, approaching the baseline performance.

5.2.2 Reconstruction Error

In addition to classification performance, we evaluated the fidelity of the selected features in reconstructing the original data. We used the Frobenius norm of the reconstruction error between the original matrix and its projection onto the selected columns as the evaluation metric.

Table 1: Classification Accuracy vs. Feature Count on MNIST

Features	% of Original	Accuracy (%)	% of Full Accuracy	Compression
5	0.64%	42.90	45.88%	156.80×
10	1.28%	56.80	60.75%	78.40×
20	2.55%	77.80	83.21%	39.20×
30	3.83%	83.60	89.41%	26.13×
50	6.38%	87.85	93.96%	15.68×
100	12.76%	90.75	97.06%	7.84×
784 (All)	100%	93.50	100.00%	1.00×

Table 2 summarizes the reconstruction error for different values of k. As expected, the reconstruction error decreases consistently with an increasing number of selected features. Notably, using just 50 features results in a reconstruction error of 8.77, which is only 20.15% of the error obtained when no features are excluded. This demonstrates that the selected subset captures a substantial portion of the structural information present in the original dataset.

Table 2: Reconstruction Error vs. Feature Count on MNIST

Features	Reconstruction Error	% of Original Error
5	37.83	86.94%
10	29.50	67.77%
20	18.63	42.81%
30	12.91	29.67%
50	8.77	20.15%
100	5.62	12.91%
784 (All)	0.00	0.00%

6 Enhanced LSCSS Algorithm

In this section, we present our Enhanced-LSCSS algorithm, which improves upon the original Linear Time Approximation Algorithm for Column Subset Selection with Local Search. Our enhanced algorithm introduces several key innovations that lead to both theoretical and practical improvements.

6.1 Algorithm Description

The Enhanced-LSCSS algorithm builds upon the classical LSCSS framework by introducing three key innovations aimed at improving both computational efficiency and selection accuracy:.

Adaptive Sampling. Rather than selecting column indices uniformly at random, our method assigns higher sampling probabilities to columns with larger residual norms. This strategy ensures a more informative initial subset, thereby reducing the number of required sampled columns from the traditional $10k$ to just $5k + \lceil\sqrt{k}\rceil$.

Diagonal Perturbation. To mitigate numerical instability arising from ill-conditioned matrices, we incorporate a diagonal perturbation matrix during the subset selection process. This improves the robustness of matrix inversions and stabilizes computations without significantly altering the solution quality.

Two-Phase Local Search. We introduce a two-stage refinement process to further improve the quality of the selected subset. The first phase performs a coarse selection of all candidate columns to identify a shortlist. The second phase then carries out a detailed grained evaluation of only the top $\lceil\sqrt{k}\rceil$ candidates, enabling faster convergence to a high-quality solution.

Algorithm 1 Enhanced-LSCSS

Input: Matrix $A \in \mathbb{R}^{n \times d}$, integer k, number of iterations T
Output: Submatrix consisting of k columns from A
Initialize $\mathcal{I} = \emptyset$, $E = A$, $B = A$
for $t = 1, 2$ **do**
  for $j = 1$ to k **do**
    Sample column index $i \in [d]$ with probability $p_i = \|E_{:i}\|_F^2 / \|E\|_F^2$
    Update $\mathcal{I} = \mathcal{I} \cup \{i\}$ and $E = A - A_{\mathcal{I}} A_{\mathcal{I}}^{\dagger} A$
  end for
  if $t = 1$ **then**
    Initialize a zero matrix $D \in \mathbb{R}^{n \times d}$, set each diagonal entry:

$$D_{ii} = \frac{\|A - A_{\mathcal{I}} A_{\mathcal{I}}^{\dagger} A\|_F}{(52\sqrt{\min\{n, d\}}(k+1)!)^{1/2}}$$

    Update $A \leftarrow A + D$, and reset $\mathcal{I} = \emptyset$
  end if
end for
Compute $A' = B + D$ and set $S = A_{\mathcal{I}}$
$\epsilon_0 = \|A' - SS^{\dagger} A'\|_F^2$
$\theta = 1/(50k)$ {Improved convergence parameter}
for $i = 1$ to T **do**
  $S \leftarrow Enhanced - LS(A', k, S, \theta)$
  $\epsilon_i = \|A' - SS^{\dagger} A'\|_F^2$
  if $\epsilon_{i-1} - \epsilon_i < \theta \cdot \epsilon_{i-1}/10$ **then**
    break {Early stopping}
  end if
end for
Let $\mathcal{I}$ be the set of column indices of S
return $A_{\mathcal{I}}$

Algorithm 2 Enhanced-LS

Input: Matrix $A' \in \mathbb{R}^{n \times d}$, integer k, matrix $S \in \mathbb{R}^{n \times k}$, parameter θ
Output: Submatrix consisting of k columns from A'
Compute residual matrix $E = A' - SS^{\dagger} A'$
Let $\mathcal{I}$ be the set of column indices of S in A'
Sample a set C of $5k$ column indices with probability proportional to $\|E_{:i}\|_F^2$
For each $i \in C$, compute potential gain $g_i = \|E_{:i}\|_F^2$
Sort C by g_i in descending order, and select top $\lceil \sqrt{k} \rceil$ indices to form C'
for each $p \in C'$ **do**
  for each $q \in \mathcal{I}$ **do**
    $\Delta_{p,q} = f(A', A'_{\mathcal{I} \setminus \{q\} \cup \{p\}}) - f(A', S)$
  end for
  $q^* = \arg\min_{q \in \mathcal{I}} \Delta_{p,q}$
  if $\Delta_{p,q^*} < -\theta \cdot f(A', S)$ **then**
    $\mathcal{I} = \mathcal{I} \setminus \{q^*\} \cup \{p\}$
    return $A'_{\mathcal{I}}$
  end if
end for
return $A'_{\mathcal{I}}$

6.2 Theoretical Guarantees

Our Enhanced-LSCSS algorithm provides improved theoretical guarantees compared to the original algorithm:

Theorem 1 (Improved Approximation Ratio) *For any matrix $A \in \mathbb{R}^{n \times d}$ and integer k, the Enhanced-LSCSS algorithm returns a set of k columns from A such that:*

$$\mathbb{E}[\|A - A_{\mathcal{I}} A_{\mathcal{I}}^{\dagger} A\|_F^2] \leq 26(k+1)\|A - A_k\|_F^2$$

where A_k denotes the best rank-k approximation of A.

This represents a significant improvement over the original approximation ratio of $53(k+1)$. The key improvements in our theoretical analysis are summarized in the following table:

Table 3: Comparison of Key Theoretical Guarantees

Metric	**Original Algorithm**	**Enhanced Algorithm**
Running Time	$O(ndk^4 \log k)$	$O(ndk^3 \log k)$
Approximation Ratio	$53(k+1)$	$26(k+1)$
Success Probability/Iteration	$1/1375$	$1/625$
Sampling Complexity	$10k$ columns	$5k + \lceil\sqrt{k}\rceil$ columns

6.3 Implications for Differentiable Approximation

The Enhanced-LSCSS algorithm is continuous, differentiable approximation within our neural network. First, its reduced computational complexity of $O(ndk^3 \log k)$ enables more efficient training and inference, accelerating convergence. The introduction of a diagonal perturbation matrix improves numerical conditioning, enhancing the stability of gradient-based optimization. Additionally, the two-phase local search mechanism encourages a more effective balance between exploration and exploitation, which naturally complements the temperature annealing strategy used in our differentiable framework. Importantly, these enhancements require minimal modification to our original approximation, allowing us to inherit improved performance and robustness with little overhead.

7 Discussion and Future Work

This work lays the foundation for further advances in differentiable subset selection. A promising direction involves hybrid models that combine neural network-based initialization with discrete refinement for improved performance. Exploring alternative relaxation techniques, such as semidefinite programming (SDP) or message-passing algorithms, may yield tighter approximations. Moreover, our framework could be generalized to other combinatorial problems involving permutations, such as sorting, ranking, and bipartite matching. Finally, there remains significant scope to deepen the theoretical understanding of the method by establishing tighter bounds on approximation error and convergence behavior.

8 Conclusion

This proposal outlines a novel approach to column subset selection through continuous approximations of discrete algorithms. By developing differentiable alternatives to key operations in the Linear Time Approximation Algorithm, we enable end-to-end training of neural networks for learning permutation matrices. Additionally, our Enhanced-LSCSS algorithm provides significant improvements in both theoretical guarantees and practical performance. Our approach bridges the gap between combinatorial optimization and deep learning, offering new possibilities for solving permutation-related problems in a differentiable framework.

References

Ahmed K Farahat, Ali Ghodsi, and Mohamed S Kamel. A fast linear time approximation algorithm for column subset selection with local search. *arXiv preprint arXiv:1303.0577*, 2013.

Christos Boutsidis, Petros Drineas, and Michael W. Mahoney. Improved matrix column selection algorithms for nyström-based kernel learning. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 51–58, 2009.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. In *International Conference on Learning Representations*, 2018.

Olivier Teboul Marco Cuturi and Jean-Philippe Vert. Differentiable ranks and sorting using optimal transport.". *arXiv preprint arXiv:1905.11885*, 2019.

Ryan P Adams and Richard S Zemel. Ranking using sinkhorn divergence. In *Advances in Neural Information Processing Systems*, pages 1368–1376, 2011.