IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 February 2024, accepted 9 April 2024, date of publication 11 April 2024, date of current version 18 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3387851

RESEARCH ARTICLE

ASPDD: An Adaptive Knowledge Distillation Framework for TSP Generalization Problems

SISI ZHENG AND RONGYE YE
School of Mathematics and Statistics, Huizhou University, Huizhou 516007, China
Corresponding author: Sisi Zheng (zhengss@hzu.edu.cn)

This work was supported in part by the 2022 Youth Project of Guangdong Basic and Applied Basic Research Fund under Grant 2022A1515110437, and in part by the 2018 Higher Education of Guangdong Key Platforms and Scientific Research Projects under Grant 2018KQNCX251.

ABSTRACT The traveling salesman problem (TSP) is a classic non-deterministic polynomial-hard (NP-hard) problem. Currently, almost all the research works utilizing Transformers to solve TSP problems employ supervised learning. However, it is extremely challenging to obtain accurate solution labels for the model in large-scale instances, resulting in a severe lack of scale generalization capability. Recent research combines knowledge distillation and Transformer to effectively address the distribution generalization issue. Nonetheless, if the framework is directly applied to the problem of scale generalization, the solution is not satisfactory. To address the aforementioned issues, we propose an adaptive soft probability distributed distillation (ASPDD) framework to improve Transformer scale generalization capability. The ASPDD framework uses a soft probability distributed distillation (SPDD) method to improve the knowledge interaction between the student and teacher models. In particular, ASPDD introduces an adaptive selection strategy, so that the student model can find weak points for improvement training in each training. This framework is utilized for training a model for a small-scale instance (TSP20) and deploying it to a large-scale instance. Extensive benchmarking (10000 instances) demonstrates that our ASPDD framework can achieve competitive results as compared to other Transformer baseline models and knowledge distillation frameworks. In addition, the ASPDD framework is applied to eleven publicly accessible benchmark datasets (TSPLIB). On six benchmark datasets, the experimental results demonstrate that our ASPDD framework outperforms previous knowledge distillation models.

INDEX TERMS Knowledge distillation, ASPDD, scale generalization, combinatorial optimization.

I. INTRODUCTION

The traveling salesman problem (TSP) has a wide range of applications in the fields of transportation, logistics distribution, etc. The research community has done a lot of research on TSP [1], [2], [3], [4]. In the early days, researchers used exact algorithms to solve the problem, but as the problem became more complex, the time required for the exact solution became prohibitive. The non-deterministic polynomial-hard (NP-hard) property of this problem makes its resolution in the field of theoretical computer science extremely challenging. As a result, researchers began employing heuristic algorithms to solve the TSP. Concorde [5] and Gourbi [6] are two powerful exact solvers; the Concorde solver can solve 85,900 city instances. In contrast, the current robust heuristic algorithm [7] can achieve superior results with millions of city nodes. Even though there are a large number of precise methods and heuristic algorithms to solve large-scale instances that can control error within a small range, these traditional solving methods require the artificial design of extremely complex constraint rules and have a huge computation time. As a result, it is often very difficult to directly extend the algorithm to practical application scenarios.

In recent years, many studies have utilized deep learning algorithms to solve TSP problems in order to address the aforementioned issues [8], [9], [10], [11], [12]. Usually, the deep learning algorithms do not require the design of complex

The associate editor coordinating the review of this manuscript and approving it for publication was Qingchao Jiang.

constraint rules and can directly solve the problems in an end-to-end manner. Consequently, this research direction continues to attract more and more attention. The Transformer model is currently one of the most prominent deep learning models, and its potential and application prospects in the fields of computer vision, natural language processing, etc. Some literature studies have been demonstrated [13], [14], [15], [16]. Recently, in the field of combinatorial optimization, there have been many studies on solving the TSP problems based on the Transformer model, and breakthrough results have been achieved. Kool et al. [17] introduced the Transformer learning paradigm in solving vehicle routing problem (VRP). Ma et al. [18] proposed a novel learning model (DACT) based on Transformer, which improves the Transformer's learning ability for such problems by combining the cyclic position coding method. The knowledge distillation method, which is an interactive learning paradigm of the teacher-student model, was first proposed by Hinton et al. [19]. This method can effectively transfer the knowledge representation of a large model to a small model, thereby achieving the goals of reducing the model's capacity and enhancing its generalization capability. Based on this, Bi et al. [20] introduced knowledge distillation in the TSP problem for the first time. They proposed adaptive multi-distribution knowledge distillation learning framework by using knowledge distillation combined with the Transformer model (AMDKD-AM). The experimental results demonstrated that the Transformer model introduced with knowledge distillation possessed a robust capacity for distribution generalization. Although knowledge distillation methods have achieved success in distribution generalization problems, as the scale of TSP instances increases, the cost of obtaining accurate labels for supervised learning (SL) models and the time overhead of instance training becomes unbearable in practical applications.

Given the aforementioned research and challenges, we pose the following question: Can the existing knowledge distillation combined with the Transformer learning framework train a small-scale instance model and deploy it in a large-scale TSP instance while maintaining a lower level of gap (i.e., scale generalization)? To accomplish this (as shown in Figure 1, we utilize the existing AMDKD-AM framework for experimental verification. The research results show that this framework cannot be used directly to complete the task of scale generalization (as shown in Table 1). Then, we asked how knowledge distillation and the Transformer model could be combined for scale generalization. In light of this, we propose a soft probability distributed distillation (SPDD) method for training a student model and scaling it to large-scale instances smoothly. In addition, we propose an adaptive selection strategy, specifically for the scale generalization problem to improve student model training. Finally, by combining the SPDD method with an adaptive selection strategy, we propose an adaptive soft probability distributed distillation (ASPDD) learning framework, based on which the student model

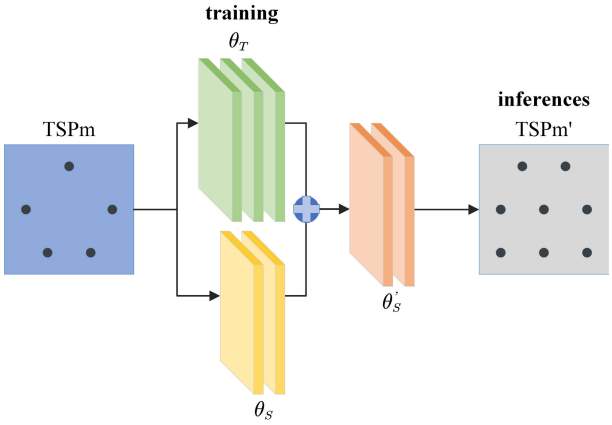

FIGURE 1. Generalization reasoning is performed on the TSP problem by means of knowledge distillation, where θ_T and θ_S represent the teacher and student models, respectively, and $m > m'$, deploy small models to larger instances through interactive learning.

can continuously approximate the geometric distribution in the teacher model space, and train a generalist student with strong academic ability. The following part of the article is structured as follows: Section II provides a brief review of related work on the study of supervised learning, deep reinforcement learning, and knowledge distillation. Section III introduces the background and motivation of ASPDD. Section IV presents the ASPDD method. Section V reports the experimental results and analyzes the reasons from different perspectives. The conclusion and future work are provided in Section VI. The code of this work is available at https://github.com/oooo111/ASPDD.

In summary, the main contributions of this paper are as follows:

- We propose a method of knowledge distillation to soften the probability distribution. The consistency of the geometric shape distribution of the student and teacher models in space is used as the learning objective, and the student model's learning ability is enhanced by modifying the learning function. This method ensures that the student model can interact more effectively with the teacher model and facilitate the expansion of the model, i.e., from a small-scale instance to a large-scale instance.
- We propose an adaptive selection strategy, which enables the student model to make adaptive adjustments based on the experimental results of different instances. In addition, it also enables us to make adjustments corresponding to the model's own weaknesses.
- We assess the proposed framework on a benchmark dataset and compare it with four baselines including LCP [21], DACT [18], AM [17], and AMDKD-AM [20]. The experimental results demonstrate that the proposed framework effectively improves the generalization ability of small-scale models, and achieves the state-of-the-art performance as compared to four baselines.

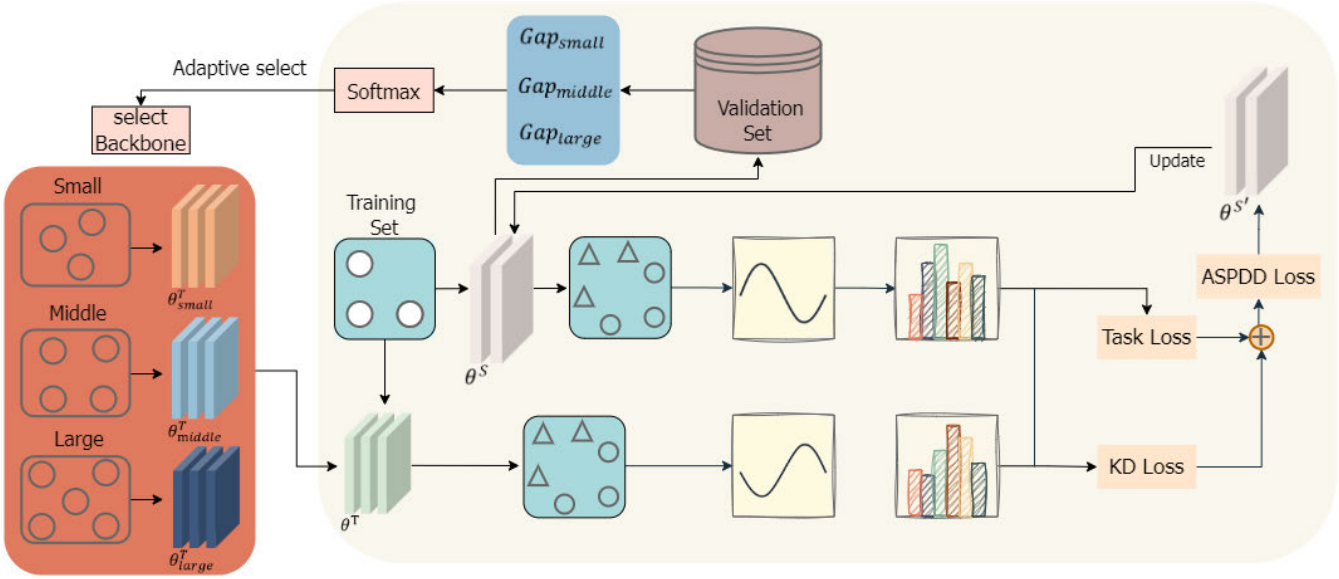

FIGURE 2. The training process of the ASPDD framework. Among them, circles and triangles represent pairs of samples. Small, middle, and large represent TSP20 TSP50, and TSP100, respectively.

II. RELATED WORKS

In this section, we present a brief literature review regarding the application of machine learning algorithms for addressing combinatorial optimization issues.

A. SUPERVISED LEARNING

Vinyals et al. [22] introduced a recurrent neural network (RNN) based on an encoder-decoder architecture to learn the conditional probability of output sequences, using attention as a pointer to select the input sequence, thereby resolving the issue of variable size output dictionaries. Their proposed pointer network (Ptr-Net) can solve the entire class of TSP instances offline; however, they only focused on small-scale TSP instances and did not generalize.

In recent years, scholars have started applying the attention mechanism to TSP issues. Kool et al. [17] improved the attention model [13] and used the deep learning training model of the simple greedy rollout baseline based on the Transformer method. This method yielded remarkable results on a number of VRPs and introduced a paradigm for Transformer training on route planning issues. The model is generalizable to TSP instances with 100 cities, but it does not generalize well to larger scales. Ma et al. [18] introduced a cyclic coding training method based on the Transformer model, which significantly enhanced the model's generalization capability. Xing and Tu [23] trained a graph neural network to capture graph motifs and vertex interactions, and combined Monte Carlo tree search (MCTS) to solve the classical TSP problem. Fu et al. [24] noted that the existing algorithms for solving TSP problems using SL severely lack generalization capability. Therefore, the authors first developed the graph sampling, graph converting, and heat maps merging techniques for TSP, and then combined deep learning and MCTS for searching for high-quality solutions, which significantly enhanced the training model's scale generalization capability. Due to MCTS, the computational cost was relatively large.

However, it was not possible to generalize the learned strategies to a larger instance on a real scale. Although these models utilized machine learning techniques, SL was still utilized. It was easy to obtain samples, but difficult to obtain labels for SL, which severely constrained the problem-solving scale and application degree. It became the bottleneck for the spread of machine learning-based combinatorial optimization problems.

B. DEEP REINFORCEMENT LEARNING

Aiming at the difficulty of label acquisition in SL, some researchers adopt deep reinforcement learning (DRL) methods and argue that DRL training may result in greater generalization than SL. Drawing from the studies of Vinyals et al. [22] and Bello et al. [25], a neural network and an actor-critic algorithm trained through reinforcement learning were proposed. Using the negative tour length as the reward signal, the strategy gradient method is used to optimize the parameters of the RNN. Finally, the Ptr-Net of TSP is trained. To solve NP-hard combinatorial optimization problems, Khalil et al. [26] proposed a framework based on reinforcement learning and structure2vec (S2V) graph embedding networks.

C. KNOWLEDGE DISTILLATION

Hinton et al. [19] formally proposed and promoted the knowledge distillation method. The main idea behind this method is to use a pre-trained teacher neural network to guide and supervise the training of a student neural network.

How to transfer knowledge from a large teacher model to a small student model is the central concept of this method. Bi et al. [20] noted that the current method for resolving VRP problems using machine learning is fundamentally based on the same distribution of instances for training and testing. Therefore, knowledge distillation was incorporated in the neural combinatorial optimization in order to enhance the cross-distribution generalization capability of solving VRP. It was proposed to use AMDKD to train lightweight models with good cross-distribution generalization performance, and it was applied to two representative deep models, i.e., AM [17] and POMO [27].

III. BACKGROUND

A. TSP PROBLEMS

This paper focuses on two-dimensional TSP problem, the TSP problem is defined as: $G = \{v, \xi\}$ is a directed graph with weights, $v_i \in v$ represents the location of each specific city, $e(v_i, v_j) \in \xi$ represents the edge connecting every two vertices, $C(e(v_i, v_j))$ represents the Euclidean distance between every two vertices, where all vertices are assumed to be uniformly distributed in two dimensions. The objective function of the TSP problem can be expressed as finding the shortest path starting from any vertex v_i, passing through all other vertices in the graph only once, and finally returning to the starting point v_i. That is, find the route θ^* (Equation (1)) with the smallest total cost among all the possible paths in the limited search space S.

$$\theta^* = \operatorname{argmin} L\left(\theta' \mid G\right) = \operatorname{argmin} \sum C(e(v_i, v_j)) \tag{1}$$

B. KNOWLEDGE DISTILLATION

Knowledge distillation is a training paradigm based on interactive learning between a teacher model and a student model. Recent research indicates that knowledge distillation can enhance the model's ability to generalize distributions and reduce the model capacity for TSP problems [20]. The mathematical expression of the loss function combined with the knowledge distillation method is given as follows:

$$\mathcal{L} = \beta \mathcal{L}_{Task} + (1 - \beta)\mathcal{L}_{KD} \tag{2}$$

where, β is a hyperparameter, $\mathcal{L}_{Task}$ represents the task loss of the student model, and $\mathcal{L}_{KD}$ is the distillation loss between the student and teacher models.

C. MOTIVATION

Classic NP-hard combinatorial optimization problems, such as TSP, have attracted the attention of many industry and academic researchers. On small-scale instances, the performance of the most advanced solution models is currently very close to that of conventional programming solvers. As the size of the instance increases, training typically requires more time and computing resources. However, the greater challenge is to pre-specify the label of the accurate solution, which is often difficult to achieve in practical application scenarios. Therefore, if the model trained on a small-scale instance is deployed on a large-scale instance while maintaining a certain level of performance, the applicability of the model in real-world scenarios can be significantly enhanced. Therefore, we require a method that can effectively transfer knowledge to enhance the model's generalizability. Knowledge distillation can effectively enhance the model's capacity for generalization and reduce its parameter capacity. Moreover, the Transformer model has great potential for solving combinatorial optimization problems such as TSP (see recent studies [17], [18]). On this basis, few researchers have recently combined Transformer with a knowledge distillation method, proposed a Transformer combined with a knowledge distillation framework, and discussed the generalization ability of the model under different distributions [20], which represents a significant advance. Unfortunately, if the framework is directly extended to scale generalization problems, it often results in large model prediction deviations. To overcome this technical obstacle, we propose an ASPDD framework to solve the model's scale generalization problem (see Section IV).

IV. METHODS

A. OVERALL FRAMEWORK OF ASPDD

First, we consider a practical teaching scenario. In a classroom, the students must first complete basic ability training. When students have attained a certain level of foundational knowledge, teachers will arrange tests to evaluate their abilities. These quizzes cover knowledge that students have not encountered in their studies. When the assessment is complete, the teacher will make corrections based on student errors and conduct targeted improvement training. Thus, after long-term training by teachers, students can develop strong learning abilities and eventually become generalists capable of solving a variety of problems. Inspired by this scenario, we propose an adaptive selection training strategy (see equation (Equation (12)) for details). In the TSP problem, the existing Transformer combined knowledge distillation framework is to measure the consistency between the teacher distribution and the student distribution by calculating the KL divergence. The specific calculation procedure is outlined below:

$$\mathcal{L}_{KD} = \frac{1}{N_T} \sum_{x \in \mathcal{X}} \sum_{i=1}^{N_T} KL\left[p_{\theta_i^T}(x), p_{\theta^S}(x)\right] \tag{3}$$

where, x represents the training sample. $p_{\theta^S}(x)$, $p_{\theta^T}(x)$ represent the spatial probability function distribution of the student model and the teacher model, respectively. The KL divergence function is defined as follows:

$$KL(p_{\theta^T}(x) \| p_{\theta^S}(x)) = p_{\theta^T}(x)(logp_{\theta^T}(x) - logp_{\theta^S}(x)) \tag{4}$$

The above formula shows that the student model is directly matched with the teacher model during the training process using the KL divergence function. However, the student model cannot learn the teacher model's deep semantic space distribution. Due to the difficulty of zero-shot generalization

problems, we need to train a model with strong knowledge reasoning and expansion capabilities. Therefore, we need a training method that can learn the latent information of the teacher model. Existing studies have demonstrated that training the interaction between data sample pairs can describe the geometric distribution rules of the feature space of the model [28]. Modeling the joint probability density function between sample pairs enables a more accurate description of the model's geometric space [29], [30]. The geometric distribution rules of the model contain important model distribution information [31]. We propose a soft probability distributed distillation (SPDD) method, which is influenced by the aforementioned techniques. Finally, we hope that the student model is able to think critically throughout the training process, and the student model cannot fully trust the answers given by the teacher model. Therefore, we soften the trained function obtained. When the model trained on small-scale instances is extended to larger-scale instances, the SPDD method can effectively address the issue of poor accuracy.

The overall diagram of the proposed ASPDD framework training is shown in Figure 2. Initially, the corresponding pre-trained teacher models are trained on three TSP20, TSP50, and TSP100 instance samples. After obtaining the pre-trained model, we sample and generate TSP20 instance samples for student model training. The student model is set to a standard AM model. Then, execute the adaptive selection strategy. In stage $E < E'$, students are trained using the teacher model pre-trained on TSP20 instances. In the $E > E'$ stage, the gap between the model solution and the standard solution is calculated on three verification sets (1000 instances) of TSP20, TSP50, and TSP100, respectively. The softmax value represents the probability of selecting the corresponding pre-trained teacher model during the next epoch.

B. ADAPTIVE SOFT PROBABILITY DISTRIBUTED DISTILLATION

In this paper, to address the aforementioned technical challenges, we propose an ASPDD framework. In this framework, the student model can learn more important and effective information by learning the geometric space distribution of the teacher model and combining adaptive selection strategies. The pseudo-code of the specific algorithm is shown in Algorithm 1.

(x_i, x_j) and (y_i, y_j) are the outputs of the teacher model and the student model, respectively. $i \in \{1, \ldots, N\}$, N is the size of the data, $x_i \in \mathbb{R}^{b\times n\times n}$, b is the batch size, and n is the size of the instance (e.g., the n of TSP50 is 50). The joint probability density function can be estimated in the function space through the kernel function [32]. The specific estimation method is as follows:

$$p_{ij} = p_{i|j}p_j = \delta\left(x_i, x_j\right) \tag{5}$$

$$q_{ij} = q_{i|j}q_j = \delta\left(y_i, y_j\right) \tag{6}$$

Algorithm 1 Adaptive Soft Probability Distribution Distillation(ASPDD)

Input: 1. Backbone model $\mathcal{M}$, train data $\mathcal{X}$(Uniform). 2. Pre-trained model($\mathcal{M}^{\mathcal{T}}$) and initialize model($\mathcal{M}^{\mathcal{S}}$).
Output: Update Student Model($\mathcal{M}^{\mathcal{S}'}$).
for *epoch=1:E* **do**
  Selecting $\mathcal{M}^{\mathcal{T}}$ based on adaptive select strategy;
  for *step=1:T* **do**
    Sampling the data of TSP20 from $\mathcal{X}$;
    Calculate $\nabla\mathcal{L}_{Task}$ (##$L(\pi)$ is total cost, $b(s)$ is standard solution);
    $\nabla\mathcal{L}_{Task} = E_{p_\theta(\pi|s)}(L(\pi) - b(s))\nabla log_{p_\theta(\pi|s)}$.
    Calculate $\nabla\mathcal{L}_{KD}$;
    $\nabla\mathcal{L}_{KD} = \tau^2\psi(\phi(p_{i|j}/\tau), \phi(q_{i|j}/\tau))$. Calculate global loss $\nabla\mathcal{L}_{SPDD}$;
    $\nabla\mathcal{L}_{SPDD} = \beta\nabla\mathcal{L}_{Task} + (1-\beta)\nabla\mathcal{L}_{KD}$.
    update the model parameter;
    $\mathcal{M}_t^S \leftarrow \mathcal{M}_{t-1}^S + \eta\nabla\mathcal{L}_{SPDD}$.
  end
end

where, q_{ij} and p_{ij} represent the joint probability density function of the student model and the teacher model in the space, respectively. (x_i, x_j) and (y_i, y_j) represent the output vectors of the teacher model and student model data sample pairs. $\delta(x, y)$ represents the kernel function. We transform the joint probability density function into a conditional probability density function using a Bayesian transformation. Among them, the conditional probability density functions of the teacher model and the student model are respectively as follows:

$$p_{i|j} = \frac{\delta\left(x_i, x_j\right)}{\sum_{n=1,n\neq j}^{N}\delta\left(x_n, x_j\right)} \tag{7}$$

$$q_{i|j} = \frac{\delta\left(y_i, y_j\right)}{\sum_{n=1,n\neq j}^{N}\delta\left(y_n, y_j\right)} \tag{8}$$

where, $p_{i|j}, q_{i|j} \in [0, 1]$. The cosine kernel function can estimate the function accurately without adjustment [28]. Therefore, we use the cosine kernel function as the kernel density function, and the specific calculation is as follows:

$$\delta_{cosine}(x, y) = \frac{1}{2}\left(\frac{x^T y}{\|x\|_2\|y\|_2} + 1\right) \tag{9}$$

where, $\delta_{cosine}(x, y) \in [0, 1]$. The objective of model training is to minimize the conditional probability distribution difference between the teacher model and the student model so that transferred samples have a high degree of consistency in the space of the teacher and student models. Finally, we soften the conditional probability density function to make the student model not fully trust the solution provided by the teacher model. The specific global loss function is defined as

TABLE 1. A comparison of the proposed ASPDD model with Transformer-derived models and AMDKD model.

Method	TSP20			TSP50			TSP100			TSP200			TSP1000		
	Cost	Gap	Time	Cost	Gap	Time	Cost	Gap	Time	Cost	Gap	Time	Cost	Gap	Time
Concorde	3.84	0.00%	(1m)	5.70	0.00%	(2m)	7.76	0.00%	(3m)	10.72	0.00%	(3.44m)	23.12	0.00%	(6.65h)
Gurobi	3.84	0.00%	(7s)	5.70	0.00%	(2m)	7.76	0.00%	(17m)	10.70	0.00%	(40.49m)	-	-	-
DACT	3.84	0.04%	(2m)	6.22	9.12%	(9m)	-	-	-	-	-	-	-	-	-
LCP{640,10}	3.84	0.00%	(2.3h)	5.95	4.39%	(3.17h)	8.97	15.59%	(3.33h)	13.95	30.13%	(57m)	39.02	68.77%	(2.13h)
AM(128)	3.85	0.34%	(0s)	5.95	4.39%	(0s)	9.00	16.00%	(3s)	13.93	29.94%	(1s)	38.64	67.12%	(17s)
AM(64)	3.85	0.16%	(0s)	5.97	4.74%	(0s)	9.04	16.49%	(1s)	14.06	31.16%	(0s)	39.76	71.97%	(17s)
AMDKD-AM$^{\diamond}$(64)	3.84	0.07%	(0s)	5.98	4.68%	(1s)	9.07	16.90%	(3s)	14.16	32.09%	(1s)	39.94	72.75%	(16s)
AMDKD-AM*(64)	3.85	0.22%	(0s)	6.00	5.26%	(1s)	9.03	16.37%	(3s)	13.83	29.01%	(1s)	38.23	65.35%	(16s)
AMDKD-AM$^{\diamond}$(128)	3.84	**0.04%**	(0s)	5.94	4.21%	(1s)	8.93	15.07%	(3s)	13.72	27.99%	(1s)	37.50	62.19%	(20s)
Ours(64)	3.84	0.05%	(1s)	5.93	**4.17%**	(1s)	8.89	**14.56%**	(3s)	13.61	**26.96%**	(1s)	37.79	63.45%	(17s)
Ours(128)	3.84	0.05%	(0s)	5.94	4.21%	(1s)	8.91	14.82%	(3s)	13.65	27.33%	(1s)	37.06	**60.29%**	(20s)

All tests are done based on the model trained on the TSP20 instance and gap is solved Concorde and Gurob. The test set comprises 10,000 instances for 20, 50, 100 and 2,000 instances for 200, 1000. AMDKD-AM* denotes the model trained using the multi-distribution teacher model described in the original paper. AMDKD-AM$^{\diamond}$ denotes the model trained using basic pattern.

follows:

$$\nabla \mathcal{L}_{KD} = \tau^2 \psi \left(\phi \left(\frac{p_{i|j}}{\tau} \right), \phi \left(\frac{q_{i|j}}{\tau} \right) \right)$$
$$\phi \left(z_i \right) = \log \left(\frac{e^{z_i}}{\sum_k e^{z_k}} \right) \tag{10}$$

where, τ represents the temperature variable during the distillation process. ψ represents the KL divergence function that measures the consistency of the teacher-student model. The $\nabla \mathcal{L}_{Task}$ uses the reinforce function rollout [17] as the task loss. The specific algorithm is presented as follows:

$$\nabla \mathcal{L}_{\text{Task}} = E_{p_\theta(\pi|s)}(L(\pi) - b(s)) \nabla \log_{p_\theta(\pi|s)} \tag{11}$$

Among them, $p_\theta(\pi|s)$ represents the probability distribution. $L(\pi)$ represents the solution predicted by the model, and $b(s)$ represents the standard solution.

The Adaptive Select Strategy: During training, we use the pre-trained TSP20 teacher model for basic training of students when $E < E'$, and the adaptive select strategy is turned off at this time. When $E \geq E'$, the probability of selecting each pre-trained teacher model is adaptively adjusted based on the performance of the student model on the TSP20, TSP50, and TSP100 validation sets. The gap value is solved by a powerful Gurobi solver, $AvgGap(S, M)$ represents the average gap between the solution $\pi_{tspm}(solver)$ of the solver at scale m and the solution $\pi_{tspm}(model)$ of the model at scale m and the specific adjustment strategy is as follows:

$$P = \begin{cases} \text{softmax}\left(AvgGap\left(S, M\right)\right) & \text{if } E \geq E' \\ [1, 0, 0] & \text{else} \end{cases} \tag{12}$$

where, $\pi_{tsp\,m}(solver)$ represents the standard solution on three instances of {20,50,100} by the Gurobi solver. P represents the adaptive probability. $\pi_{tsp\,m}(model)$ represents the solution of the student model on three instances of {20,50,100}. The adaptive selection strategy is turned off when $E < E'$ (i.e., select pre-trained TSP20 as the teacher model). In this study, our adaptive approach involves dynamically selecting teacher models with varying instance scales, which differs from the approach outlined in [20]. In [20], an adaptive method is used to select teacher models from different distributions. The adaptive technique applied in our paper represents a modification of the approach introduced in [20], specifically tailored to address the challenge of scale generalization. Although there might be similarities in the mathematical expressions, the intended applications are distinct. It's essential to underscore that this method primarily serves as a training technique in our work and isn't the primary innovation of this study.

V. EXPERIMENT

A. EXPERIMENTAL SETUP

We experimentally compare the proposed ASPDD framework with four transformer variants: 1) **DACT [18]**: a transformer model based on cyclic coding; 2) **LCP [21]**: a learning collaborative policies model; 3) **AM [17]**: a basic transformer model for the VRP problem; 4) **AMDKD-AM [20]**: a knowledge distillation framework combined with Transformer model. Specific model details can be found in the original literature. The main focus in this work is to enhance the scale generalization ability of Transformer. Therefore, we specifically compare the Transformer-based models only. The task loss adopts the commonly used rollout method [17], and the hyperparameters are consistent with the AMDKD-AM network fundamental architecture. The training strategy uses greedy approach. We propose an adaptive strategy for addressing the scale generalization problem, i.e., the selection of different pre-trained teacher models of varying scale only, while maintaining the training configuration of the student model similar to other comparison methods. This approach is similar to AMDKD-AM, which utilizes auxiliary teacher models pre-trained on different distributions. When using the teacher

TABLE 2. Effectiveness of ASPDD extensions in TSPLIB instances.

Instance	Opt.	AM [17]		AMDKD-AM [20]		ASPDD(Ours)	
		Gap	Cost	Gap	Cost	Gap	Cost
ts225	126,643	426%	665642	74.32%	220763	**49.82%**	189736
pr226	80369	534%	509503	60.19%	128799	**58.18%**	127124
berlin52	7542	165%	19953	**26.98%**	9578	87.32%	14128
lin318	42029	564%	278883	**185.00%**	119730	249%	146919
pcb442	50778	717%	414729	**175.00%**	139576	187.00%	145680
rd400	15281	1106%	184278	151%	38376	**129.00%**	35018
st70	675	358%	3092	**56.40%**	1056	74.96%	1181
rd100	7910	589%	54507	**82.07%**	14402	85.95%	14709
d493	35002	428%	184808	218.00%	111219	**124%**	78382
eil101	629	303%	2533	113.00%	1342	**79.65%**	1130
ch130	6110	603%	46606	124.00%	13658	**120%**	13442
avg.gap	0.00%		532.07%		115.09%		113.17%

All instance tests use models trained on TSP20 instances.

model for multi-scale training, it is essential to note that this approach is only feasible under the framework of knowledge distillation. Therefore, we solve the scale generalization problem of TSP under the knowledge distillation framework.

The model trained on the TSP20 instance is then applied to five datasets of TSP20, TSP50, TSP100, TSP200, and TSP1000. Among them, we use the standard baseline data set (10000 instances) [17] for the three scale instances, i.e., 20, 50, and 100, for testing. For instances of 200 and 1000 scales, we use Kool et al. [17] to generate 2000 instance nodes for testing. The gap is calculated using conventional programming solvers Concorde [5] and Gurobi [6]. All experiments are implemented on a signal GPU NVIDIA RTX 3090. The proposed ASPDD framework is implemented in Pytorch [33].

B. EFFECTIVENESS ANALYSIS OF ASPDD

In this experiment, we first compare the proposed ASPDD framework with the mainstream Transformer series models. In Table 1, the model trained on the TSP20 instance is deployed to five scale instances, and the model's index evaluation uses three indexes, including computational cost, gap, and computation time.

The performances of other existing models exhibit the following characteristics, as shown in Table 1: First, the performance of the DACT model in the scale generalization problem is nearly inferior to that of other models in terms of gap and time, and on a single-card GPU, memory overflow occurs when the test node exceeds 50. Second, the gap of the LCP model is as low as 0.00% on the TSP20 instance, but the time overhead is huge, i.e., the model requires time in exchange for gap optimization, and when the number of nodes exceeds 20, the model performance begins to decrease. Third, as the scale of the instance increases, the performance of the AM(64) model declines and AMDKD-AM(64) model becomes larger. Compared to the AMDKD-AM model, the proposed ASPDD framework has the following performance characteristics: First, it can improve the model performance while maintaining a relatively constant time consumption. After conducting experiments, we have observed an intriguing phenomenon where various AMDKD-AM training modes exhibit unique strengths and weaknesses across different scale instances. For the purpose of convenient discussion, this paper will primarily focus on the basic mode of AMDKD-AM. Second, on the scale of TSP20, the proposed 64-dimensional model achieves a 0.02% improvement in terms of gap as compared to the AMDKD-AM$^{\diamond}$ framework; on the scale of TSP50, it achieves a 0.51% improvement in terms of gap; on the scale of TSP100, it achieves a 2.34% improvement in terms of gap, and on the scale of TSP1000, it achieves a 9.3% improvement in terms of gap. Third, when the model node embedding dimension is 64, the proposed method significantly outperforms the AMDKD-AM and AM models of the same embedding dimension. However, as compared to other models with a node embedding dimension of 128, the proposed ASPDD (128) shows limited improvement. The powerful learning ability of the large model itself is primarily responsible for the limited improvement in model's performance. Fourth, as the model capacity of ASPDD increases, the gap only improves on the TSP1000 instance. This demonstrates that in the proposed ASPDD framework, the bigger model may not have a better effect. The aforementioned experimental results validate the effectiveness of the proposed ASPDD framework.

C. BENCHMARK DATASET EXTENSION

Next, in order to effectively validate the effectiveness of our ASPDD framework, we randomly select eleven benchmark datasets from TSPLIB. We train on data instances with a size of 20 and test with the trained model.

The experimental results are presented in Table 2. The AM model performs extremely poorly when faced with unknown distribution and scale-up instances. Relatively speaking, the performance of the AMDKD-AM model introduced with knowledge distillation and interactive learning has been greatly improved. For six benchmark datasets, the proposed

TABLE 3. The results of ablation experiments.

Method		Tsp20			Tsp50		
Adapative	Loss	Cost	Gap	Time	Cost	Gap	Time
✓	✓	3.84	**0.05%**	(1s)	5.93	**4.17%**	(1s)
×	✓	3.85	0.07%	(0s)	5.94	4.21%	(1s)
×	×	3.84	0.07%	(0s)	5.98	4.91%	(1s)

The loss represents the SPDD loss function.

TABLE 4. The results of ablation experiments(Epoch represents E').

Parameter	Tsp20		Tsp50		Tsp100	
Epoch	Cost	Gap	Cost	Gap	Cost	Gap
500	3.85	0.09%	5.95	4.61%	8.93	15.12%
600	3.85	0.07%	5.96	4.82%	8.96	15.50%
800	3.85	0.12%	5.96	4.82%	8.93	15.20%
1000	3.84	**0.05%**	5.93	**4.17%**	8.89	**14.56%**

ASPDD framework performs better than the AMDKD-AM framework.

D. ABLATION EXPERIMENTS

Finally, we conduct ablation experiments on ASPDD to effectively evaluate the performance of the proposed ASPDD framework. In particular, we select two scale data of TSP20 and TSP50 to conduct ablation experiments for validating the effectiveness of adaptive selection strategy and SPDD method. The experimental results are presented in Table 3. The model that employs the adaptive strategy and the SPDD method achieves the highest level of precision. The increase in the TSP20 instance scale is 0.02% when compared with no adaptive strategy. In the TSP50 instance scale, the improvement is 0.04% compared to the non-adaptive strategy and 0.74% compared to the non-SPDD method. In addition, with regard to the adaptive strategy, we conducted experimental analyses on the selection of different epochs.The experimental results are presented in Table 4.

VI. CONCLUSION AND FUTURE WORKS

In this paper, we propose an ASPDD framework for the TSP problem to address the lack of scalability of existing Transformer series models and knowledge distillation frameworks. Unlike the existing AMDKD-AM framework that solves the distribution generalization problem, our goal is to deploy the model trained on a small-scale instance to a larger-scale instance in order to solve the issues of long training time and difficulty in obtaining solution labels for the Transformer model in a large-scale instance. In order to achieve this, we propose the SPDD loss function and an adaptive adjustment strategy. In particular, we train a generalist student on small-scale instances by learning the geometric distribution of the teacher model combined with an adaptive selection strategy. In five instance scales, the quantitative experimental results demonstrate that the proposed ASPDD framework is significantly more competitive than the mainstream Transformer series models and AMDKD-AM framework. Moreover, in the largest TSP1000 instance, the proposed ASPDD (64) achieves a 9.3% improvement in terms of gap as compared to the AMDKD-AM(64). Overall, the proposed ASPDD framework improves model performance, while maintaining a roughly constant time overhead. Extended baseline experiments (TSPLIB) demonstrate that the proposed ASPDD framework outperforms AMDKD-AM on six baseline data and has a substantial gap optimization compared to the AM model without knowledge distillation. We will conduct the following research in future work:

- Integrate additional model types in the proposed ASPDD framework and extend them to a variety of combinatorial optimization problems.
- Introduce a reward mechanism to effectively train the student-teacher model using reinforcement strategies.

AUTHOR CONTRIBUTIONS

The authors confirm their contributions to the paper as follows: study conception and design: Sisi Zheng and Rongye Ye; data collection: Rongye Ye; analysis and interpretation of results: Rongye Ye; and draft manuscript preparation: Sisi Zheng and Rongye Ye. They reviewed the results and approved the final version of the manuscript.

DECLARATION OF CONFLICTING INTERESTS

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

REFERENCES

[1] L. Duan, Y. Zhan, H. Hu, Y. Gong, J. Wei, X. Zhang, and Y. Xu, "Efficiently solving the practical vehicle routing problem: A novel joint learning approach," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 3054–3063.

[2] G. D. Konstantakopoulos, S. P. Gayialis, and E. P. Kechagias, "Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification," *Oper. Res.*, vol. 22, no. 3, pp. 2033–2062, Jul. 2022.

[3] Z. Qin, X. Tang, Y. Jiao, F. Zhang, Z. Xu, H. Zhu, and J. Ye, "Ride-hailing order dispatching at DiDi via reinforcement learning," *INFORMS J. Appl. Anal.*, vol. 50, no. 5, pp. 272–286, Sep. 2020.

[4] R. Ye, R. Ye, and S. Zheng, "Machine learning guides the solution of blocks relocation problem in container terminals," *Transp. Res. Record, J. Transp. Res. Board*, vol. 2677, no. 3, pp. 721–737, Mar. 2023.

[5] C. Kwon. (2022). *Python Wrapper Around the Concorde TSP Solver*. Accessed: Mar. 27, 2023. [Online]. Available: https://github.com/jvkersch/pyconcorde

[6] Gurobi Optim. (2021). *Gurobi Optimizer Reference Manual*. Accessed: Mar. 27, 2023. [Online]. Available: https://www.gurobi.com

[7] É. D. Taillard and K. Helsgaun, "POPMUSIC for the travelling salesman problem," *Eur. J. Oper. Res.*, vol. 272, no. 2, pp. 420–429, Jan. 2019.

[8] P. Emami and S. Ranka, "Learning permutations with sinkhorn policy gradient," 2018, *arXiv:1805.07010*.

[9] Y. Kaempfer and L. Wolf, "Learning the multiple traveling salesmen problem with permutation invariant pooling networks," 2018, *arXiv:1803.09621*.

[10] C. K. Joshi, T. Laurent, and X. Bresson, "An efficient graph convolutional network technique for the travelling salesman problem," 2019, *arXiv:1906.01227*.

[11] X. Chen and Y. Tian, "Learning to perform local rewriting for combinatorial optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–12.

[12] W. Ouyang, Y. Wang, S. Han, Z. Jin, and P. Weng, "Improving generalization of deep reinforcement learning-based TSP solvers," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2021, pp. 01–08.
[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30, pp. 1–11.
[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.
[15] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, H. Lin, Z. Zhang, Y. Sun, T. He, J. Mueller, R. Manmatha, M. Li, and A. Smola, "ResNeSt: Split-attention networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2022, pp. 2736–2746.
[16] R. Ye, R. Wang, Y. Guo, and L. Chen, "SIA-unet: A unet with sequence information for gastrointestinal tract segmentation," in *Proc. Pacific Rim Int. Conf. Artif. Intell.* Cham, Switzerland: Springer, 2022, pp. 316–326.
[17] W. Kool, H. van Hoof, and M. Welling, "Attention, learn to solve routing problems!" 2018, *arXiv:1803.08475*.
[18] Y. Ma, J. Li, Z. Cao, W. Song, L. Zhang, Z. Chen, and J. Tang, "Learning to iteratively solve routing problems with dual-aspect collaborative transformer," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 11096–11107.
[19] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.
[20] J. Bi, Y. Ma, J. Wang, Z. Cao, J. Chen, Y. Sun, and Y. M. Chee, "Learning generalizable models for vehicle routing problems via knowledge distillation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 31226–31238.
[21] M. Kim and J. Park, "Learning collaborative policies to solve NP-hard routing problems," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 10418–10430.
[22] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, vol. 28, pp. 1–9.
[23] Z. Xing and S. Tu, "A graph neural network assisted Monte Carlo tree search approach to traveling salesman problem," *IEEE Access*, vol. 8, pp. 108418–108428, 2020.
[24] Z. H. Fu, K. B. Qiu, and H. Zha, "Generalize a small pre-trained model to arbitrarily large TSP instances," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 8, pp. 7474–7482.
[25] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," 2016, *arXiv:1611.09940*.
[26] E. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song, "Learning combinatorial optimization algorithms over graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30, pp. 1–11.
[27] Y.-D. Kwon, J. Choo, B. Kim, I. Yoon, Y. Gwon, and S. Min, "POMO: Policy optimization with multiple optima for reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 21188–21198.
[28] N. Passalis and A. Tefas, "Learning deep representations with probabilistic knowledge transfer," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 268–284.
[29] G. E. Hinton and S. Roweis, "Stochastic neighbor embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, vol. 15, pp. 1–8.
[30] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.
[31] H. Chen, P. Wang, F. Wang, W. Tian, L. Xiong, and H. Li, "EPro-PnP: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 2771–2780.
[32] D. W. Scott, *Multivariate Density Estimation: Theory, Practice, and Visualization*. Hoboken, NJ, USA: Wiley, 2015.
[33] Pytorch. (2023). *Tensors and Dynamic Neural Networks in Python With Strong GPU Acceleration*. Accessed: Mar. 27, 2023. [Online]. Available: https://github.com/pytorch/pytorch

SISI ZHENG received the Ph.D. degree in management from the South China University of Technology, in 2018. Since July 2018, she has been a full-time Teacher with the School of Mathematics and Statistics, Huizhou University, China. She has published more than 20 academic articles in mainstream journals in the field of management, of which eight are included in SCI, one is included in EI, and four are included in CSSCI core journals. She presided more than one Guangdong Provincial Fundamental and Applied Basic Research Fund, in 2022, and one Guangdong University Key Platform and Scientific Research Project Young Innovative Talent Project, in 2018, and participated in a number of provincial scientific research projects. She guided students to participate in subject competitions and won multiple international and national awards. She has won many awards, including the National Scholarship for Ph.D. Students.

RONGYE YE is currently pursuing the bachelor's degree with the School of Mathematics and Statistics, Huizhou University. He published three SCI articles in journals, such as *Transportation Research Record* and *PLOS One*. His research interests include deep learning, machine learning, and knowledge distillation. He has won the Silver Medal in the Kaggle Competition, the Second Prize in the National Teddy Cup Data Mining Competition, and other honors.