A Multiple Gradient Descent Design for Multi-task Learning on Edge Computing: Multi-objective Machine Learning Approach

Xiaojun Zhou, *Member, IEEE,* Yuan Gao, Chaojie Li, *Member, IEEE* and Zhaoke Huang

Abstract—Multi-task learning technique is widely utilized in machine learning modeling where commonalities and differences across multiple tasks are exploited. However, multiple conflicting objectives often occur in multi-task learning. Conventionally, a common compromise is to minimize the weighted sum of multiple objectives which may be invalid if the objectives are competing. In this paper, a novel multi-objective machine learning approach is proposed to solve this challenging issue, which reformulates the multi-task learning as multi-objective optimization. To address the issues contributed by existing multi-objective optimization algorithms, a multi-gradient descent algorithm is introduced for the multi-objective machine learning problem by which an innovative gradient-based optimization is leveraged to converge to an optimal solution of the Pareto set. Moreover, the gradient surgery for the multi-gradient descent algorithm is proposed to obtain a stable Pareto optimal solution. As most of the edge computing devices are computational resource-constrained, the proposed method is implemented for optimizing the edge device's memory, computation and communication demands. The proposed method is applied to the multiple license plate recognition problem. The experimental results show that the proposed method outperforms state-of-the-art learning methods and can successfully find solutions that balance multiple objectives of the learning task over different datasets.

Index Terms—multi-task learning, multiple gradient descent, edge computing, multi-objective machine learning, deep neural network.

1 Introduction

Learning model is inherently a multi-objective task [1]. In machine learning model, on the one hand, learning methods usually perform model selection and parameter estimation based on multiple criteria. On the other hand, it is generally to train a model or multiple models to perform the required task. Machine learning methods are usually divided into single-objective machine learning and multi-objective machine learning [2]. Single-objective machine learning is a learning paradigm that optimizes only one objective function. The trained model between each objective is independent, and useful information is also contained only in training data for individual learning objective, so it cannot obtain more useful information from the learning process of other objectives. Hence, single-objective machine learning ignores feature sharing, subspace sharing and parameter sharing. Multi-objective machine learning is an inductive transfer method, which uses the domain information contained in the training data as the inductive bias to improve generalization. By learning objectives in parallel with shared representation, what is learned for each objective can help other objectives be learned better [3]. Many multi-objective machine learning methods have been proposed in recent years, and have achieved excellent performance in many fields such as computer vision [4], [5], multimedia learning [6], natural language processing [7], anomaly detection [8], speech recognition [9] and network calculus [10].

According to the relationship among multiple objectives, multi-objective machine learning can be divided into two categories: the traditional multi-objective learning and multi-objective machine learning based on deep learning. Multi-objective machine learning based on deep learning avoids the problem of insufficient information utilization in traditional multi-objective learning method, and it comprehensively considers the relationship between features and model parameters. Multi-objective machine learning based on deep learning is typically conducted via hard or soft parameter sharing. In hard parameter sharing, a subset of the parameters is shared among tasks while other parameters are task-specific. In soft parameter sharing, each task has its own model with its own parameters. The distance between the parameters of the model is then regularized in order to encourage the parameters to be similar. A new "cross-stitch" sharing unit was proposed in [11], which combines the activations from multiple networks and can be trained end-to-end. It can learn an optimal combination of shared and task-specific representations. An automatic approach for designing multi-task deep learning architectures was proposed in [12]. The approach started with a thin multi-layer network and dynamically widens it in a greedy manner during training. It can create a tree-like deep architecture by doing so iteratively. A deep convolutional network of multi-objective machine learning was proposed based on tensor normal priors, which can alleviate the dilemma of negative-transfer in feature layers and under-transfer in the classifier layers [13]. The parameters from all models were regularised by the tensor trace norm in [14], and the sharing strategy was learned in a data-driven way.

- *X. J. Zhou, Y. Gao and Z. K. Huang are with the School of Automation, Central South University, Changsha 410083, China, and X. J. Zhou is also with the State Key Laboratory of Synthetical Automation for Process Industries, Shenyang 110000, China. (email: michael.x.zhou@csu.edu.cn; gao_yuan@csu.edu.cn; huangzhaoke@csu.edu.cn)*
- *C. J. Li is with the School of Electrical Engineering and Telecommunications, The University of New South Wales, Kensington, NSW 2052, Australia. (email: chaojie.li@unsw.edu.au)*
Corresponding author: Chaojie Li (email: chaojie.li@unsw.edu.au)

Manuscript received XX, 2021; revised XX, 2021.

Multiple conflicting objectives often appear in multi-objective machine learning. It is impossible to find a solution to optimize all tasks. A common compromise is to optimize a proxy objective that minimizes a weighted linear combination of per-task losses. Linear weighted summation is a priori method, which needs to determine weights in advance. However, it is difficult to adjust the task weight manually, which is costly and highly subjective. At present, some methods are proposed to automatically adjust the weights by taking the weights of tasks as the parameters in the process of model training. A principled approach was proposed for multi-objective machine learning which weighs multiple loss functions by considering the homoscedastic uncertainty of each objective [15]. A gradient normalization algorithm that automatically balances training in multi-objective machine learning models by dynamically tuning gradient magnitudes was proposed in [16]. However, it is difficult to find the optimal solution of the multi-objective machine learning problem with these heuristic methods.

Multi-objective optimization addresses the problem of optimizing a set of possibly conflicting objectives. Population-based and gradient-free multi-objective evolutionary algorithms (MOEAs) are popular methods to find a set of well-distributed Pareto solutions in a single run. A collaborative multi-objective learning method, in which the multi-objective learning problem was expressed as a multi-objective optimization problem, and the multi-objective particle swarm optimization algorithm was adopted to solve the multi-objective learning problem in [17]. The equivalence relationship between clustered multi-task learning (CMTL) and alternating structure optimization was established in [18]. The combination of meta-learning with a modified multi-objective particle swarm optimization (MOPSO) which uses the crowding distance mechanism (MOPSO-CDR) is proposed to solve the multi-objective learning problems in [19]. A novel support vector machine multi-task multiple kernel learning (MT-MKL) framework was proposed in [20] that considered an implicitly defined set of conic combinations of task objectives, and the obtained solution corresponds to a single point on the Pareto Front (PF) of a multi-objective optimization problem. However, it can not be used for solving large scale and gradient-based multi-task learning problems. Therefore, it is necessary to find a multi-objective optimization method to solve these problems effectively and quickly.

Considering that MOEAs need to consume a lot of computing time in deep learning, and multi-objective gradient descent is an efficient approach for multi-objective optimization when gradient information is available; as a result, a multi-gradient descent algorithm (MGDA) is introduced, which adopts an innovative gradient-based optimization algorithm that is leveraged to converge to an optimal solution of the Pareto set. It can use the gradients of each task and solve an optimization problem to decide on an update over the shared parameters. By using a combination of gradients, a common descent direction for all objectives can be found. MGDA was compared with MOEAs in cost efficiency and was found to be suitable for multi-objective machine learning with deep networks [21].

In recent years, a new trend in computing is happening with the function of clouds being increasingly moving towards the network edges. Edge computing places compute nodes close to end devices, which can meet the high computation and low-latency requirements of deep learning on edge devices, and also provides better privacy, bandwidth efficiency, and scalability. For some computation-intensive and latency-critical tasks in multi-objective machine learning, devices are supposed to be deployed in a distributed manner and it is necessary to offload multiple objective tasks to multiple edge servers for optimizing the edge device's memory, computation and communication demands, which can reduce response time, provide more efficient processing, and alleviate the pressure on the network [22], [23], [24].

In this paper, a multiple gradient descent design for multi-task learning, i.e., a multi-objective machine learning approach, based on edge computing is proposed. The main contributions of this paper are summarized as follows: (1) To solve the multiple conflicting objectives problem in multi-task learning, the multi-objective machine learning problem is reformulated as the multi-objective optimization problem to solve the tradeoff problems among different objectives. A multi-task learning method based on multi-gradient descent algorithm is proposed for finding an optimal solution of the Pareto set; (2) To avoid the problem that traditional multi-gradient descent algorithm may converge to the two endpoints of the Pareto front, a gradient surgery for the multi-gradient descent algorithm is proposed to obtain a stable Pareto optimal solution; (3) As most of the edge computing devices are computational resource-constrained, the proposed method is implemented for optimizing the edge device's memory, computation and communication demands; (4) The proposed method is successfully applied to solve the multiple license plate recognition problem.

2 Proposed Multi-Objective Machine Learning Method for Edge Computing

2.1 Basic Definitions

Machine learning is inherently a multi-objective task, and multi-objective machine learning can be formulated as multi-objective optimization, which is optimizing a collection of possibly conflicting objectives. A multi-objective machine learning problem can be described by T correlated tasks with a loss vector:

$$\min_{\substack{\theta^{sh},\\ \theta^1,...,\theta^T}} \left(L^1\left(\theta^{sh},\theta^1\right),...,L^T\left(\theta^{sh},\theta^T\right)\right)^{\mathrm{T}} \tag{1}$$

where $L^t\left(\theta^{sh},\theta^t\right)$, $t=1,2,\cdots,T$, is the loss of the t-th task, θ^{sh} is the shared parameter, and θ^t is the task-specific parameter. Problem (1) is a multi-objective optimization problem (MOP). The resolution of a MOP yields a set of compromise solutions representing the optimal trade-offs among the different objectives.

There are several concepts related to multi-objective optimization in multi-objective machine learning.

Definition 1 (Pareto Dominance). Let θ and $\bar{\theta}$ be two points, θ is said to dominate $\bar{\theta}$, denoted as $\theta \prec \bar{\theta}$, if and only if $L^i\left(\theta^{sh},\theta^i\right) \leq L^i\left(\bar{\theta}^{sh},\bar{\theta}^i\right), \forall i \in \{1,2,...,T\}$ and $L^j\left(\theta^{sh},\theta^j\right) < L^j\left(\bar{\theta}^{sh},\bar{\theta}^j\right), \exists j \in \{1,2,...,T\}$.

Definition 2 (Pareto Optimality). A point $\theta^* \in \Omega$ is called Pareto optimal point if it is not dominated by any other point. The set of all Pareto optimal points is called the Pareto set.

2.2 Multi-gradient Descent Algorithm Based on Gradient Surgery

The multi-gradient descent algorithm usually adopts the following iterative formula to update the shared parameters θ^{sh}

$$\theta^{sh} = \theta^{sh} - \eta \sum\nolimits_{t=1}^{T} \alpha^t \nabla_{\theta^{sh}} L^t\left(\theta^{sh},\theta^t\right) \tag{2}$$

where α^t is the weight for the t-th task, and η is the learning rate.

Let's define the following constrained minimization problem:

$$\min_{\alpha^1,\ldots,\alpha^T} \left\| \sum_{t=1}^{T} \alpha^t \nabla_{\theta^{sh}} L^t\left(\theta^{sh}, \theta^t\right) \right\|_2^2$$
$$\text{s.t.} \quad \sum_{t=1}^{T} \alpha^t = 1, \alpha^t \geq 0, \forall t \in \{1, \cdots, T\}. \tag{3}$$

Next, the following theorem is given to show that the multi-gradient descent algorithm can generate a Pareto point to the multi-objective optimization problem (1).

Theorem 1. Supposing that $\alpha^t (t = 1, \cdots, T)$ is the solution to the constrained minimization problem (3) and $\sum_{t=1}^{T} \alpha^t \nabla_{\theta^{sh}} L^t\left(\theta^{sh}, \theta^t\right) = 0$, then $(\theta^{sh}, \theta^1, \cdots, \theta^T)$ is a Pareto optimal solution to problem (1).

Proof: Constructing the Lagrangian as follows

$$\mathcal{L} = \left\| \sum_{t=1}^{T} \alpha^t \nabla_{\theta^{sh}} L^t\left(\theta^{sh}, \theta^t\right) \right\|_2^2 + \lambda\left(\sum_{t=1}^{T} \alpha_t - 1\right) \tag{4}$$

then the optimality conditions to the constrained problem (3) are

$$\begin{aligned} &\frac{\partial \mathcal{L}}{\partial \alpha_t} = 0, \forall t \in \{1, \cdots, T\}; \\ &\frac{\partial \mathcal{L}}{\partial \lambda} = 0. \end{aligned} \tag{5}$$

By combing the Eq. (5) and $\sum_{t=1}^{T} \alpha^t \nabla_{\theta^{sh}} L^t\left(\theta^{sh}, \theta^t\right) = 0$, we have

$$\alpha^1 \nabla L^1(\theta^{sh}, \theta^1) + \cdots + \alpha^T \nabla L^T(\theta^{sh}, \theta^T) = 0 \tag{6}$$
$$\sum_{t=1}^{T} \alpha^t = 1, \alpha^t \geq 0, \forall t$$

which is equivalent to the optimal conditions of the multi-objective machine learning problem (1). This completes the proof. □

To calculate α effeciently, let's consider the case of two tasks ($T = 2$), and then the optimization problem can be defined as follows

$$\min_{\alpha \in [0,1]} \left\| \alpha u + (1 - \alpha) v \right\|_2^2. \tag{7}$$

Note that

$$\begin{aligned} f(\alpha) &= \|\alpha u + (1 - \alpha) v\|_2^2 \\ &= (\alpha u + (1 - \alpha) v, \alpha u + (1 - \alpha) v). \end{aligned}$$

By taking the derivative with respect to Eq. (7), it gives

$$f'(\alpha) = 2\left(u - v, \alpha(u - v) + v\right). \tag{8}$$

When $u \neq v$, we have

$$(\alpha u + (1 - \alpha) v) \cdot (u - v) = 0. \tag{9}$$

Thus,

$$\alpha = \frac{v \cdot (v - u)}{\|u - v\|^2} = \frac{\|v\|^2 - v \cdot u}{\|u\|^2 - 2u \cdot v + \|v\|^2}. \tag{10}$$

When $0 < \alpha < 1$, such that

$$\begin{aligned} 0 < \alpha < 1 \Leftrightarrow \quad & 0 < \|v\|^2 - v \cdot u < \|u\|^2 - 2u \cdot v + \|v\|^2, \\ \Leftrightarrow \quad & u \cdot v < \min\left(\|u\|, \|v\|\right)^2, \\ \Leftrightarrow \quad & \cos\widehat{(u, v)} < \frac{\min(\|u\|, \|v\|)}{\max(\|u\|, \|v\|)}, \\ \Leftrightarrow \quad & \widehat{(u, v)} > \cos^{-1} \frac{\min(\|u\|, \|v\|)}{\max(\|u\|, \|v\|)}. \end{aligned} \tag{11}$$

It can be seen from the above analysis that the angle between u and v is at least equal to a certain limit angle, and the value range should be $\left[0, \frac{\pi}{2}\right]$. Therefore, the sufficient condition is that the angle between u and v is an obtuse angle ($u \cdot v < 0$). Whenever the norms of u and v are very different, the limit angle will be close to $\pi/2$. Conversely, if the norms of the gradient vectors u and v are close to each other, the limit angle is small.

Thus, α can be solved as:

$$\alpha = \begin{cases} 0 & , \text{if} \;\; u^{\mathrm{T}} v \geq v^{\mathrm{T}} v, \\ \frac{v \cdot (v - u)}{\|u - v\|^2} & , \text{if} \;\; u^{\mathrm{T}} v < u^{\mathrm{T}} u \text{ and } u^{\mathrm{T}} v < u^{\mathrm{T}} u, \\ 1 & , \text{if} \;\; u^{\mathrm{T}} v \geq u^{\mathrm{T}} u. \end{cases} \tag{12}$$

When the numbers of tasks $T > 2$, the Frank-Wolfe convex optimization method [25] is adopted to solve the problem (3) quickly and effectively, and Eq. (12) is used as a subroutine for the line search. However, when $\alpha = 0$ or 1, MDGA will converge to the two endpoints of the Pareto front, which is easy to make the learning performance of one task better, while the learning performance of another task is relatively poor. Therefore, it is difficult for MDGA to stably converge to a point close to the middle position on the Pareto front of the above optimization problem.

In this study, the gradient surgery for the multi-gradient descent algorithm is proposed to obtain a stable Pareto optimal solution. The goal of gradient surgery is to modify the gradients for each task so as to aviod that MDGA may converge to the two endpoints of the Pareto front. The gradient information of multiple tasks is considered comprehensively to avoid some tasks dominating the whole gradient descent process. Suppose the gradient of the shared parameters for one task is u, the gradient of the shared parameters for the other task is v. When $u^{\mathrm{T}} v \geq v^{\mathrm{T}} v$ or $u^{\mathrm{T}} v \geq u^{\mathrm{T}} u$, the gradient of each task is projected onto the normal plane of the gradient of the other task. When $u^{\mathrm{T}} v < v^{\mathrm{T}} v$ and $u^{\mathrm{T}} v < u^{\mathrm{T}} u$, the gradient of each task remains unaltered. A pictorial description of gradient projection is shown in Fig. 1. And its procedures are given as below.

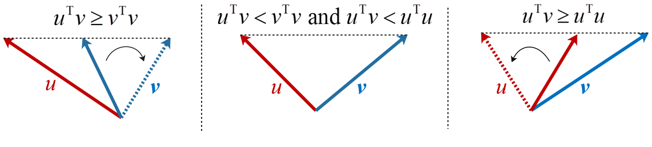
$u^{\mathrm{T}} v \geq v^{\mathrm{T}} v$
u
v
$u^{\mathrm{T}} v < v^{\mathrm{T}} v$ and $u^{\mathrm{T}} v < u^{\mathrm{T}} u$
u
v
$u^{\mathrm{T}} v \geq u^{\mathrm{T}} u$
u
v

Fig. 1: The diagram of gradient projection.

First, it determines whether to perform gradient surgery by comparing $u^{\mathrm{T}} v$ with $v^{\mathrm{T}} v$, and comparing $u^{\mathrm{T}} v$ with $u^{\mathrm{T}} u$.

Second, if $u^{\mathrm{T}} v \geq v^{\mathrm{T}} v$, then v is projected onto the normal vector of u, thus, $v = v + \frac{v \cdot u}{\|u\|^2} u$. If $u^{\mathrm{T}} v \geq u^{\mathrm{T}} u$, then u is projected onto the normal vector of v, and $u = u + \frac{u \cdot v}{\|v\|^2} v$. If $u^{\mathrm{T}} v < v^{\mathrm{T}} v$ and $u^{\mathrm{T}} v < u^{\mathrm{T}} u$, then u and v remain unaltered.

Third, to repeat this process across all of the other objectives sampled in random order from the current batch and perform

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2021.3067454, IEEE Transactions on Network Science and Engineering

the same procedure for all objectives in the batch to obtain their respective gradients.

2.3 Proposed Multi-task Learning Method with Multiple Gradient Descent Based on Edge Computing

According to the above analysis, the pseudo-code of the proposed method is shown in **Algorithm 1**. Edge computing enables a hierarchical architecture of end devices, edge compute nodes, and cloud data centers. In multi-objective machine learning, edge computing can provide computing resources and scale with the number of tasks, avoiding network bottlenecks at a central location. In this study, we consider a edge computing network composed by one cloud server, T edge servers and T mobile devices, as shown in Fig. 2. Without loss of generality, we assume that edge servers communicate with each other, and shared tasks are computed and processed by these edge devices. At the same time, each edge server has one specific task where each task can be computed by the mobile device itself or be offloaded to and processed by the edge servers or the cloud server.

This paper takes multi-license plate recognition as an example to elaborate on the proposed method. In the surveillance area with a large field of vision, multiple license plate recognition tasks often occur simultaneously. Multiple license plate recognition is more efficient and practical than single license plate recognition. A camera can monitor multiple lanes, and the captured images contain more vehicles, which can detect violations of traffic lights, vehicle line pressure, and other violations in real time. Therefore, multiple license plate recognition technology can effectively save manpower and money, reduce the use and maintenance of monitoring equipment, and provide efficient and practical management methods for intelligent transportation.

In deep neural networks, the commonly used multi-task learning method is usually to share hidden layer parameters. This study adopts a hard sharing mechanism of parameters, uses multi-objective machine learning strategies to mine valuable shared information among multiple license plate recognition tasks, and establishes the end-to-end license plate recognition model based on multi-objective machine learning. The structure of the model and the basic block are shown in Table 1 and Table 2, respetively. In the end-to-end license plate recognition model based on multi-objective machine learning, except for the last layer, the remaining intermediate layers (Layers 1-11) are all regarded as shared layers, which are used to extract common features. The last layer (Layer 12) is regarded as the task-specific layer, and which is set for each license plate recognition task to extract task-specific features. The end-to-end license plate recognition model architecture based on multi-objective machine learning is shown in Fig. 3.

Supposing that the blank placeholder is ϵ, by inserting ϵ at the first position of the license plate label sequence and after each character, the license plate label character library is defined as L, and after adding the blank placeholder, it is $L' = L \cup \{\epsilon\}$. In the license plate recognition, there is a timing problem that the length of the model output sequence is greater than the length of the label, because there are no real corresponding labels at some moments, or the output labels are blank placeholders.

Algorithm 1 The pseudo-code of the proposed method

```
Input: $\eta$: learning rate;
    Maxgen: Maximum Iterations;
    $B = \{T_t\}$: task minibatch
Output: $\theta^{sh}$: shared parameters;
    $\theta^t$: task-specific parameters
for $t$ = 1 to $T$ do
    $\theta^t = \theta^t - \eta \nabla_{\theta^t} L^t\left(\theta^{sh}, \theta^t\right)$
end for
$\boldsymbol{\alpha} = \left(\alpha^1, \ldots, \alpha^T\right) = \left(\frac{1}{T}, \ldots, \frac{1}{T}\right)$
$g_t \leftarrow \nabla_{\theta^{sh}} L^t(\theta^{sh}, \theta^t) \quad \forall t$
$g_t^{PC} \leftarrow g_t \quad \forall t$
for $T_i \in B$ do
    for $T_j \overset{uniformly}{\sim} B, T_i$ in random order do
        if $g_i^{\mathrm{T}} g_j \geq g_i^{\mathrm{T}} g_i$ then
            $g_j^{PC} = g_j^{PC} + \frac{g_j^{PC} \cdot g_i}{\|g_i\|^2} g_i$
        end if
        if $g_i^{\mathrm{T}} g_j \geq g_j^{\mathrm{T}} g_j$ then
            $g_i^{PC} = g_i^{PC} + \frac{g_i^{PC} \cdot g_j}{\|g_j\|^2} g_j$
        end if
    end for
end for
$g_i = \sum_i g_i^{PC}$, $g_j = \sum_j g_j^{PC}$
Compute $\boldsymbol{M}_{ij}$ = $g_i^{\mathrm{T}} g_j$
iter = 0
while iter $<$ Maxgen do
    $\hat{t} = \arg\min_{\gamma} \sum_t \alpha^t \mathbf{M}_{\gamma t}$
    $\hat{\gamma} = \arg\min_{\gamma} \left((1-\gamma)\boldsymbol{\alpha} + \gamma \mathbf{M}_{\hat{t}}\right)^{\top} \mathbf{M}((1-\gamma)\boldsymbol{\alpha} + \gamma \mathbf{M}_{\hat{t}})$
    $\boldsymbol{\alpha} = (1-\hat{\gamma})\boldsymbol{\alpha} + \hat{\gamma} \mathbf{M}_{\hat{t}}$
    iter = iter +1
end while
$\theta^{sh} = \theta^{sh} - \eta \sum_{t=1}^{T} \alpha^t \nabla_{\theta^{sh}} L^t\left(\theta^{sh}, \theta^t\right)$
```

TABLE 1: The structure of end-to-end license plate recognition model based on multi-objective machine learning

Layer	Stage	Channels	Filter size	Stride
1	Conv	64	3×3	1×1
2	Pool	64	3×3	1×1
3	Basic Block	128	3×3	1×1
4	Pool	64	3×3	2×1
5	Basic Block	128	3×3	1×1
6	Basic Block	128	3×3	1×1
7	Pool	64	3×3	2×1
8	Dropout	-	-	-
9	Conv	256	4×1	1×1
10	Dropout	-	-	-
11	Conv	68	1×13	1×1
12	Conv	68	1×1	1×1

Supposing that the arbitrary output sequence path of the model is $\pi = \{\pi_1, \pi_2, ..., \pi_n\}$, the label sequence corresponding to the license plate recognition task is $l = \{l_1, l_2, ..., l_m\}\,(m < n)$, Assuming that the output probability of each moment is independent of other moments. Given that the input sequence is x and the output sequence is π, the probability is expressed as follows:

$$P\left(\pi | x\right) = \prod_{t=1}^{T} y_{\pi_t}^{t} \tag{13}$$

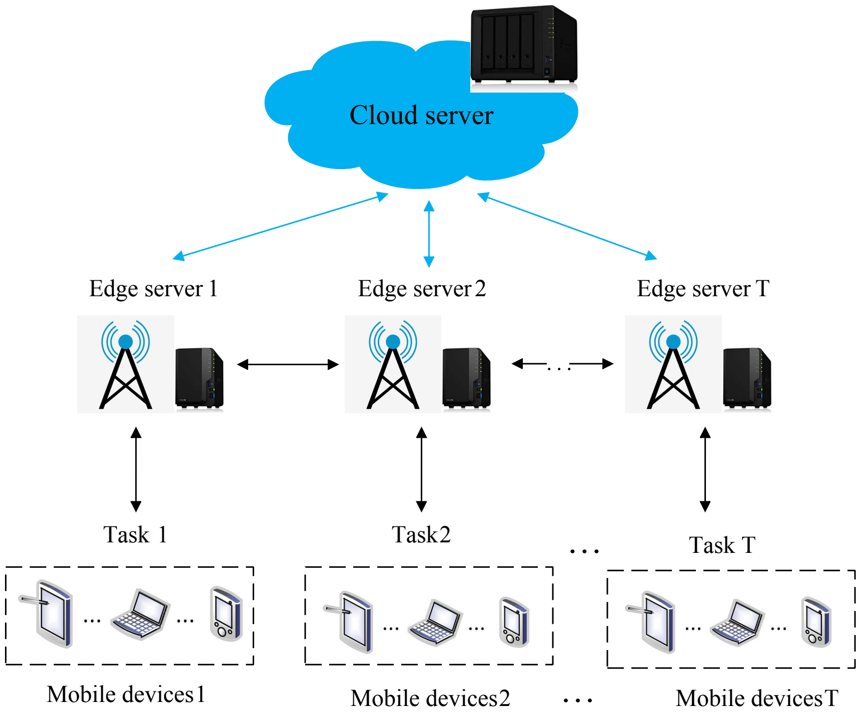

Fig. 2: The system model of multi-objective machine learning based on edge computing network.

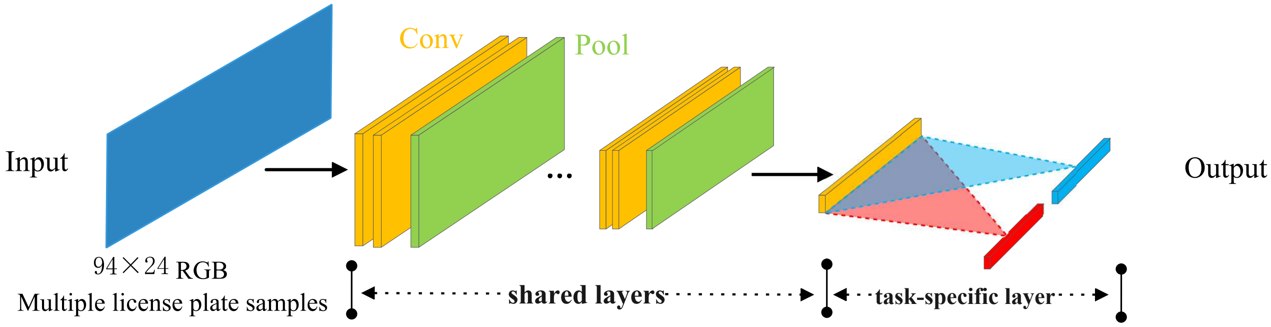

Fig. 3: An end-to-end license plate recognition model architecture based on multi-objective machine learning.

TABLE 2: Basic Block

Layer	Stage	Channels	Filter size	Stride
1	Conv	32	1×1	1×1
2	Conv	32	3×1	1×1
3	Conv	32	1×3	1×1
4	Conv	128	1×1	1×1

where $y_{\pi_t}^t$ represents the probability of outputting π_t at time t, and Eq. (13) represents the multiplication of the probabilities of all characters in a path.

Let's define a many-to-one mapping function $\varphi : L'^T \rightarrow L^T$, which means that a given input sequence x can be mapped to a collection of all possible output sequences on the label space L. The specific operations are given as follows: 1) De-duplicating consecutive identical license plate characters; 2) Removing blank placeholders. Although all paths are different, different paths can finally be mapped to the same label sequence l.

The conditional distribution probability of the label sequence l under the input sequence x is equal to the sum of the conditional probabilities of all paths that satisfy the mapping relationship, which can be expressed as follows:

$$P(l|x) = \sum_{\pi\in\varphi^{-1}(l)} P(\pi|x) = \sum_{\pi\in\varphi^{-1}(l)} \prod_{t=1}^{T} y_{\pi_t}^t \tag{14}$$

Eq. (14) represents the sum of all possible path probabilities. Minimizing the negative log likelihood is the goal of optimization during the training process of the end-to-end license plate recognition model based on multi-objective machine learning, which is given as below

$$\begin{aligned} L^t\left(\theta^{sh},\theta^t\right) &= \sum_{l,y\in\theta} -\ln\left(P\left(l|x\right)\right) \\ &= -\sum_{l,y\in\theta} \ln \sum_{\pi\in\varphi^{-1}(l)} \prod_{t=1}^{T} y_{\pi_t}^t \end{aligned} \tag{15}$$

The timing strategy gives a probability mapping from time-series input to the output sequence. The conditional probability distribution of the sequence of labels given model output is equal to the sum of conditional probability distribution satisfying the relationship path. Adopting this alignment style can make different paths mapped to the same label, and eventually obtain the maximum value of the sum of probabilities. This is no necessity for license plate character segmentation, and the output sequence and the sequence of labels can be automatically aligned, so that the timing problems can be well addressed.

In the decoding process, for a given input, the vehicle license plate character recognition results are obtained by decoding the model output according to the most probable path, which are given as follows

$$l^* = \arg\max_{l} P\left(l|x\right). \tag{16}$$

Therefore, taking multi-license plate recognition problem as an example, the detailed procedures of the proposed method are summarized as follows:

Step 1: For all tasks, the loss functions are calculated for all tasks $L^t\left(\theta^{sh}, \theta^t\right)$. The gradient of the shared parameters θ^{sh} is denoted as $\nabla_{\theta^{sh}} L^t\left(\theta^{sh}, \theta^t\right)$. The gradient of the task-specific parameters θ^t is denoted as $\nabla_{\theta^t} L^t\left(\theta^{sh}, \theta^t\right)$.

Step 2: Gradient descent is implemented on the task-specific parameters $\theta^t = \theta^t - \eta\nabla_{\theta^t} L^t\left(\theta^{sh}, \theta^t\right)$.

Step 3: Weight Initialization for all tasks. The gradient of the shared parameters θ^{sh} for the i-th and j-th task are denoted as g_i and g_j, respectively.

Step 4: By comparing $g_i^{\mathrm{T}} g_j$ with $g_i^{\mathrm{T}} g_i$, and comparing $g_i^{\mathrm{T}} g_j$ with $g_j^{\mathrm{T}} g_j$ to determine whether to perform gradient surgery.

Step 5: g_i and g_j is computed by gradient surgery. The gradient matrix for the shared parameters $\boldsymbol{M}_{ij}$ consisted of $g_i^{\mathrm{T}} g_j$ is computed.

Step 6: The optimal index $\hat{t}$ corresponding to the gradient matrix is computed.

Step 7: Eq. (12) is adopted as a subroutine for the line search to solve the constrained optimization problem (3).

Step 8: The solution $(\sum_{t=1}^{T} \alpha^t \nabla_{\theta^{sh}} L^t(\theta^{sh}, \theta^t))$ obtained by Step 7 as a gradient update applied to shared parameters.

Step 9: If the maximum number of iterations is reached, return to Step 1.

Step 10: For the given input data x, the output of the model is decoded according to Eq. (16) to obtain the recognition result of license plate characters.

3 EXPERIMENTS AND DISCUSSION

In this section, several experiments are conducted to verify the effectiveness of the proposed method. All the comparative experiments are conducted in Python 3.7 on NVIDIA Corporation GP102 [TITAN Xp].

3.1 Dataset

The large-scale Chinese City Parking Dataset (CCPD) included seven subdatasets, which consists of approximately 280,000 different license plate samples [26]. In order to solve the problem of multiple license plate recognition, a Multi-task convolutional neural network (MTCNN) [27] is adopted to detect and extract multiple license plate regions. Convolution operation of the Proposal Net (PNet) in MTCNN is adopted to replace the sliding window operation. It has the advantages of small size and fast speed to determine all candidate areas that may be license plates. Then, Output Net (ONet) is used to further refine and determine the areas of the license plates. Multiple license plates are automatically spliced together in a random manner in the left and right directions to form a multiple license plate training set. The multi-license plate dataset contains normal license plates, uneven lighting, shooting distances quite far or close, tilted license plates, extreme weather, and other challenging license plate images. Therefore, the recognition of multiple license plates in complex environments is considered in the experiments.

Table 3 lists the relevant descriptions of CCPD and the sizes of training sets and test sets. The size of each multi-license plate sample is 94 × 24. When considering a small-scale two-plate recognition problem, the recognition task of the left license plate is regarded as one task (task-L), and the recognition task of the right license plate is regarded as another task (task-R). When considering the problem of three license plate recognition, the recognition of the intermediate license plate is regarded as a task (task-M) on the basis of task-L and task-R.

Since CCPD contains various license plate samples with different angles, different distances, and different lighting. In this paper, spatial transformer networks (STN) [28] is adopted to preprocess the license plate samples before training. The structure of STN is shown in Table 4. STN can automatically learn transformation parameters, reduce the influence of the tilt and deformation of the license plate images, and enhance the robustness of license plate recognition.

In order to ensure fairness of the experiment, the settings of the main parameters in the experiment are the same as shown in Table 5. The above five main parameters used in this paper are general parameters for improving model training. The same model is used in the comparative experiments, and different comparison methods are used to adjust the weights of loss functions. In the process of adjusting the weight, there is no need to give fixed parameters such as the above five parameters in advance. Therefore, this parameter setting is fair to all comparison algorithms.

During the experiment, the learning rate is gradually decayed. In the early stage of model training, a larger learning rate is used to accelerate learning. As the number of iterations increases, the learning rate is gradually reduced to ensure that the model does not fluctuate too much in the later stages of training. This makes it easier to find the local or global optimal solution. The sizes of training sets and test sets are listed in Table 3.

The detailed process of applying the proposed method to the multi-license plate recognition experiment based on edge computing is as follows: First, in order to be able to use the deep learning model proposed in this paper on Android, we converted it to TorchScript format. Then, we add PyTorch Mobile to Gradle dependencies. Last, PyTorch Mobile is used to load models on mobile phones for license plate recognition.

TABLE 4: Parameters of STN

Layer	Stage	Channels	Filter size	Stride
1	Conv	32	3×3	1×1
2	Pool	32	2×2	2×2
3	Conv	32	5×5	1×1
4	Conv	32	3×3	3×3
5	FC	32	–	–
6	FC	6	–	–

TABLE 3: Information about sub-datasets in CCPD

Dataset	Description	Train set	Test set
CCPD-Base	Normal license plate image	149,999	49,999
CCPD-DB	The license plate image is unevenly lit, dark or bright	15,001	5,000
CCPD-FN	The distance from the license plate to the shooting location is relatively far or near	15000	4999
CCPD-Rotate	Horizontal tilt degree $20^\circ \sim 50^\circ$ and the vertical tilt degree varies from $-10^\circ \sim 10^\circ$	7,499	2,499
CCPD-Tilt	Horizontal tilt degree $15^\circ \sim 45^\circ$ and the vertical tilt degree varies from $15^\circ \sim 45^\circ$	7,500	2,500
CCPD-Weather	License plate images taken on rainy, snowy or foggy days	7,500	2,499
CCPD-Challenge	Other more challenging license plate images	7,503	2,503

TABLE 5: Parameters setting

Parameter	value
Batch size	128
Dropout rate	0.5
Weight decay	2e-5
Epoch	40
Learning rate	1e-2,1e-3,1e-4

3.2 Verification Experiments

In this section, the proposed method is compared with the single-objective machine learning method. The single-objective machine learning method is adopted to solve the license plate recognition task independently, which is represented in the statistical results as “Single task”. The proposed method is tested on multi-license plates datasets, and the single-objective machine method is tested on the single-license plates datasets.

Table 6 shows the performance comparison results on multi-license plate datasets containing two license plates. Accuracy-L and accuracy-R represent the license plate recognition accuracy of task-L and task-R, respectively. Compared with the single-task baseline, the proposed method can achieve higher recognition accuracy. The highest license plate recognition accuracy achieves up to 99.83$\%$ average precision on CCPD-Tilt, and the accuracy-L and accuracy-R of the proposed method are 3.26$\%$ and 2.73$\%$ higher than the single-task baseline, respectively.

As can be observed from Table 7, the performance comparison results on multi-license plate datasets containing three license plates show the license plate recognition accuracy of the proposed method are better than the single-task baseline on CCPD. To sum up, the proposed method can effectively solve the problem of multiple license plate recognition, and it outperforms baselines for all license plates recognition tasks and achieves comparable performance, indicating that the tasks cooperate with and help each other.

3.3 Comparative Experiments

In this section, the proposed method is compared with three representative multi-objective machine learning methods: 1) Uniform scaling: minimize the weighted sum of loss functions $\frac{1}{T}\sum_{t=1}^{T} L^t$; 2) Kendall’s method: using the uncertainty weighting [15]. 3) GradNorm: using the normalization [16].

The scatter plot of license plate recognition accuracy of the proposed method and comparison method on CCPD-Base is visualized in Fig. 4. The two solid yellow lines represent the accuracy-L and accuracy-R of the single-task baseline. The red shaded area represents that the license plate recognition accuracy of the proposed method outperforms the single-task baseline. It can be seen from Fig. 4 that the performance of the proposed method is better than other comparison methods on CCPD, which is located in the red shaded area. Kendall’s method and GradNorm find solutions that are distinctly better than uniform scaling. The same visualization method is used for the other six subdatasets, as shown in Fig. 5.

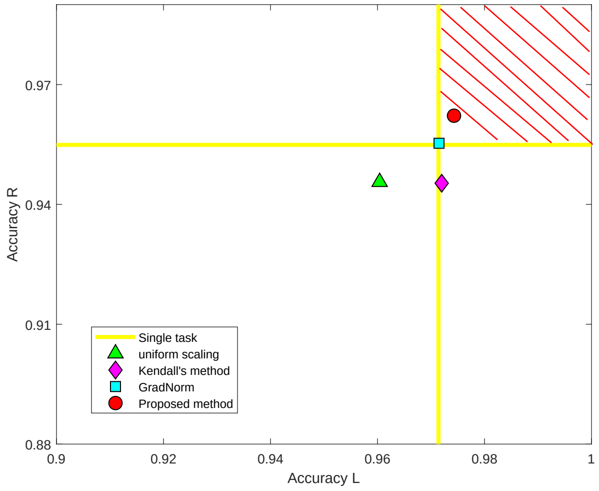

Fig. 4: Accuracy of different methods on CCPD-Base.

As shown in Table 8 and Fig.5, it can be found that the license plate recognition accuracy of the proposed method is better than other multi-objective machine learning methods on two license plate recognition tasks. The accuracy-L of Kendall’s method is the same as the proposed method are both 97.63$\%$ on CCPD-Weather. The accuracy-R of the proposed method is better than Kendall’s method. Although the accuracy-L of Kendall’s method and GradNorm are slightly better than the proposed method on CCPD-Challenge. For accuracy-R, the proposed method outperforms Kendall’s method and GradNorm. Moreover, for the license plate sample with large tilt and deformation (CCPD-Rotate), the accuracy-L of the method proposed in this paper is 4.14$\%$, 4.22$\%$ and 4.54$\%$ higher than other three methods, respectively. The accuracy-R of the proposed method is 5.15$\%$, 4.10$\%$ and 2.09$\%$ higher than other three methods.

Compared with the recognition task of two license plates, the recognition task of three license plates is more complicated and more difficult. It can be seen from Table 8 that the comparison algorithm has different degrees of decline in the accuracy of license plate recognition. The highest license plate recognition accuracy of the proposed method can reach 99.35$\%$ on CCPD-Base, and accuracy-R is 13.81$\%$, 10.56$\%$ and 8.63$\%$ higher than other three comparison algorithms on CCPD-Rotate, respectively.

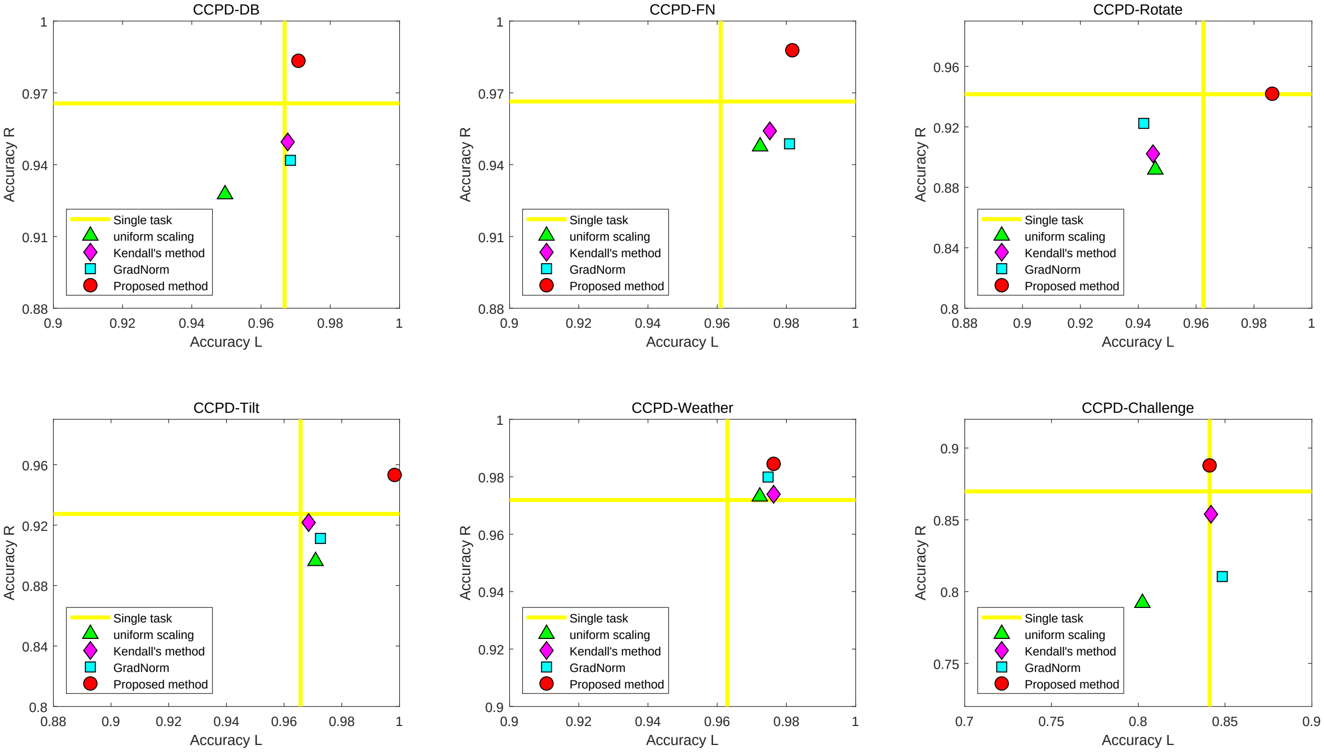

Fig. 5: Accuracy of different methods on CCPD.

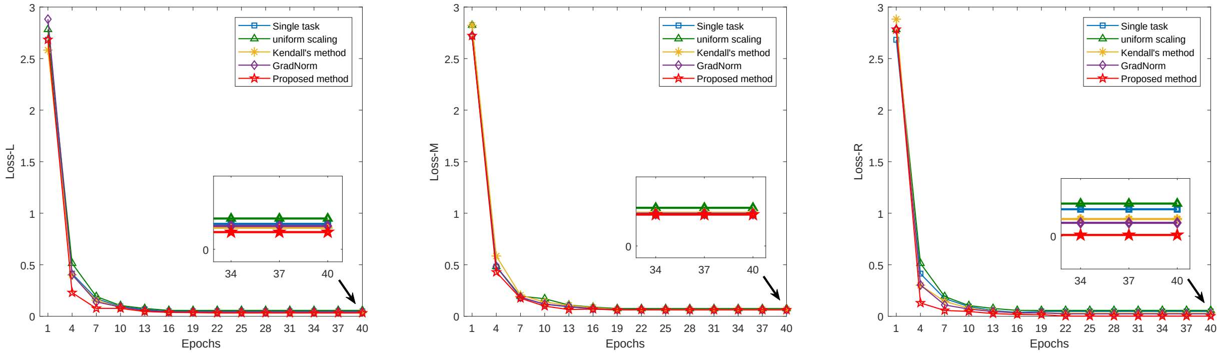

Fig. 6: Convergence curves of training loss function with different methods on CCPD-Base.

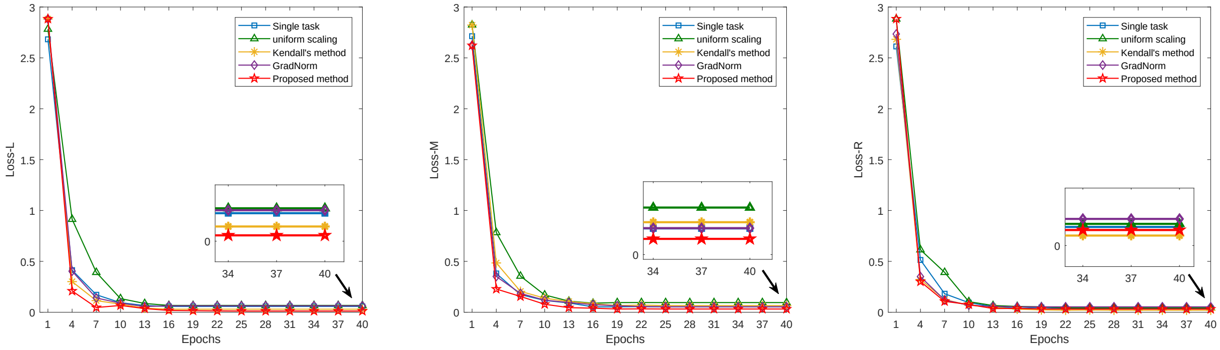

Fig. 7: Convergence curves of training loss function with different methods on CCPD-DB.

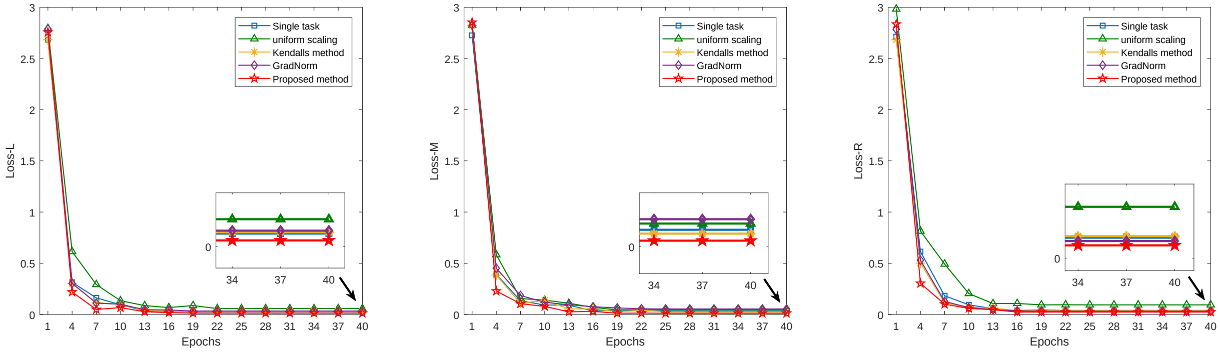

Fig. 8: Convergence curves of training loss function with different methods on CCPD-FN.

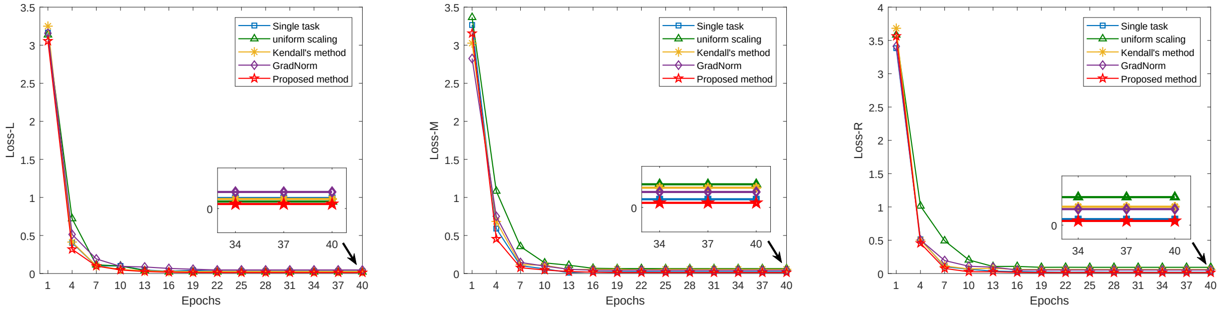

Fig. 9: Convergence curves of training loss function with different methods on CCPD-Rotate.

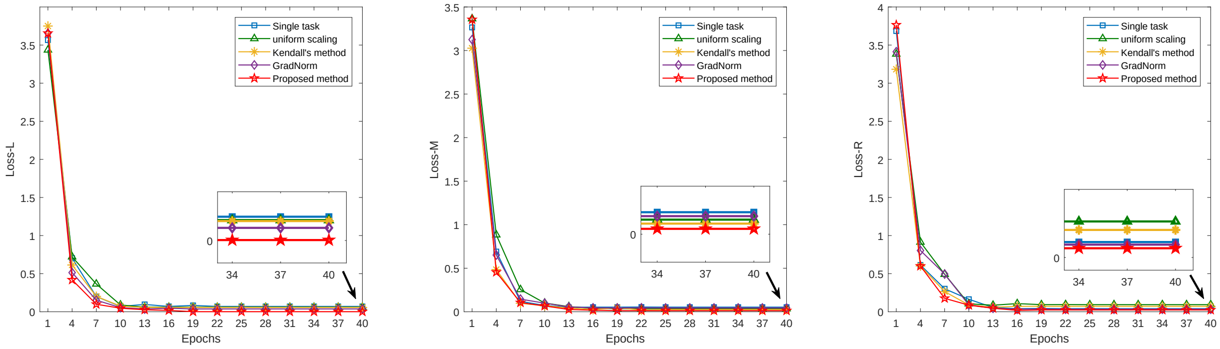

Fig. 10: Convergence curves of training loss function with different methods on CCPD-Tilt.

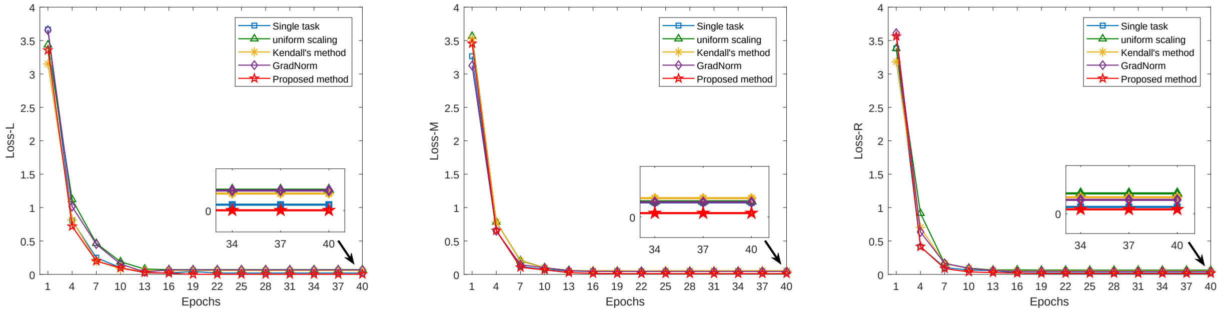

Fig. 11: Convergence curves of training loss function with different methods on CCPD-Weather.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2021.3067454, IEEE Transactions on Network Science and Engineering

TABLE 6: The performance of different methods on datasets with two license plates

Method	Evaluation index	Dataset						
		CCPD-Base	CCPD-DB	CCPD-FN	CCPD-Rotate	CCPD-Tilt	CCPD-Weather	CCPD-Challenge
Single task	Accuracy-L(%)	97.14	96.68	96.10	96.25	96.57	96.29	84.11
	Accuracy-R(%)	95.49	96.56	96.64	94.16	92.74	97.19	86.98
Proposed method	Accuracy-L(%)	**97.54**	**97.16**	**98.36**	**98.73**	**99.83**	**97.63**	**84.45**
	Accuracy-R(%)	**96.35**	**98.44**	**98.85**	**94.32**	**95.47**	**98.55**	**88.89**

TABLE 7: The performance of different methods on datasets with three license plates

Method	Evaluation index	Dataset						
		CCPD-Base	CCPD-DB	CCPD-FN	CCPD-Rotate	CCPD-Tilt	CCPD-Weather	CCPD-Challenge
Single task	Accuracy-L(%)	94.14	95.36	94.38	93.15	91.23	95.19	81.19
	Accuracy-M(%)	96.78	94.87	96.73	96.19	90.75	94.92	84.66
	Accuracy-R(%)	95.37	96.58	95.84	92.47	93.64	97.28	87.46
Proposed method	Accuracy-L(%)	**97.31**	**98.57**	**97.31**	**96.71**	**97.95**	**97.99**	**84.33**
	Accuracy-M(%)	**97.02**	**96.88**	**98.88**	**96.41**	**96.58**	**96.57**	**85.69**
	Accuracy-R(%)	**99.35**	**97.88**	**96.58**	**96.78**	**95.41**	**97.65**	**88.99**

TABLE 8: Accuracy of different methods on datasets with two license plates

Method	Evaluation index	Dataset						
		CCPD-Base	CCPD-DB	CCPD-FN	CCPD-Rotate	CCPD-Tilt	CCPD-Weather	CCPD-Challenge
uniform scaling	Accuracy L(%)	96.04	94.96	97.24	94.59	97.09	97.23	80.24
	Accuracy R(%)	94.56	92.76	94.76	89.17	89.62	97.31	79.21
Kendall's method	Accuracy L(%)	97.20	96.77	97.52	94.51	96.85	97.63	84.19
	Accuracy R(%)	94.53	94.95	95.41	90.22	92.17	97.39	85.39
GradNorm	Accuracy L(%)	97.15	96.85	98.09	94.19	97.26	97.47	**84.84**
	Accuracy R(%)	95.53	94.18	94.87	92.23	91.12	97.99	81.05
Proposed method	Accuracy-L(%)	**97.54**	**97.16**	**98.36**	**98.73**	**99.83**	**97.63**	84.45
	Accuracy-R(%)	**96.35**	**98.44**	**98.85**	**94.32**	**95.47**	**98.55**	**88.89**

TABLE 9: Accuracy of different methods on datasets with three license plates

Method	Evaluation index	Dataset						
		CCPD-Base	CCPD-DB	CCPD-FN	CCPD-Rotate	CCPD-Tilt	CCPD-Weather	CCPD-Challenge
uniform scaling	Accuracy L(%)	93.14	90.57	90.24	94.59	91.34	91.43	80.13
	Accuracy M(%)	92.86	87.34	95.14	91.35	96.37	94.23	75.19
	Accuracy R(%)	92.11	93.56	88.35	82.97	84.62	92.46	74.84
Kendall's method	Accuracy L(%)	96.23	96.32	94.34	93.37	91.46	93.25	82.24
	Accuracy M(%)	94.37	93.12	97.72	93.59	96.41	94.61	78.15
	Accuracy R(%)	95.84	97.88	90.42	86.22	87.43	93.89	82.56
GradNorm	Accuracy L(%)	97.01	95.35	94.14	92.57	94.24	91.49	76.67
	Accuracy M(%)	94.24	94.19	94.85	95.17	94.73	94.14	84.33
	Accuracy R(%)	96.47	90.91	96.10	88.15	94.26	94.27	84.22
Proposed method	Accuracy-L(%)	**97.31**	**98.57**	**97.31**	**96.71**	**97.95**	**97.99**	**84.33**
	Accuracy-M(%)	**97.02**	**96.88**	**98.88**	**96.41**	**96.58**	**96.57**	**85.69**
	Accuracy-R(%)	**99.35**	**97.88**	**96.58**	**96.78**	**95.41**	**97.65**	**88.99**

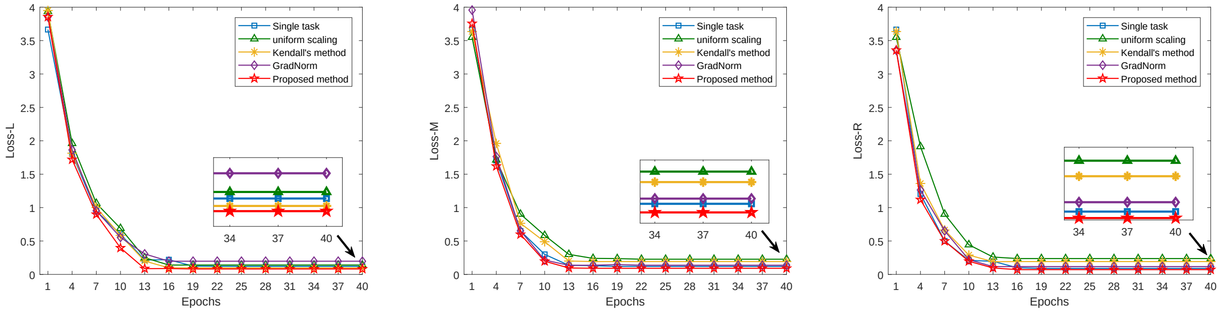

Fig. 12: Convergence curves of training loss function with different methods on CCPD-Challenge.

The accuracy-R of Kendall's method is the same as the proposed method are both 97.88% on CCPD-DB. The proposed method outperforms other comparison algorithms for the majority of tasks. The experiment results also show that the proposed method remains effective when the number of tasks is high.

Figs. 6-12 show the downward trend of loss on seven subdatasets during the training process, respectively, where loss-L, loss-M and loss-R represent the loss functions of the left, middle and right license plate recognition tasks. As the number of epochs increases, the loss function of all comparison algorithms finally decline rapidly and converge gradually. The loss decreases to a point of stability after the 13th epoch and fluctuates slowly. For license plate samples in different complex environments, the loss rates of different methods are different. Although on CCPD-Challenge, the loss of all methods in the training process decreases slowly. However, the proposed method has a faster decline on loss-L, loss-M and loss-R, and the final loss value is the smallest. Only on CCPD-DB, the loss-R obtained by the Kendall's method is smaller than that obtained by the proposed method. Compared with other algorithms, the loss of the proposed method decreases faster and the loss value is smaller.

Compared with the other state-of-the-art learning methods, it can be found that the proposed method can effectively improve the accuracy of license plate recognition. For various tilted and deformed license plate datasets with different viewing angles, different distances, and different lighting. The proposed method is superior to other multi-objective machine learning methods, and has good robustness and generalization performance.

4 Conclusion

In this paper, a novel multi-objective machine learning method was proposed. First, considering that the traditional weighted sum method is invalid when the objectives are competing. In view of the issues contributed by existing multi-objective optimization algorithms, a multi-gradient descent algorithm was introduced where an innovative gradient-based optimization was leveraged to converge to an optimal solution of the Pareto set. Second, in order to avoid converging to the two endpoints of the Pareto front, the gradient surgery for the multi-gradient descent algorithm was proposed to obtain a stable Pareto optimal solution. Third, as the most of edge computing devices are computational resource-constrained, the proposed method was implemented for optimizing the edge device's memory, computation and communication demands. Last, the proposed method was successfully applied to solve the multiple license plate recognition problem. The experimental results showed that the proposed method outperforms state-of-the-art learning methods and can successfully find solutions that balance multiple objectives of the learning task over different datasets.

Acknowledgments

The authors thank the National Natural Science Foundation of China (Grant No. 61873285), the National Key Research and Development Program of China (Grant No. 2018AAA0101603), International Cooperation and Exchange of the National Natural Science Foundation of China (Grant No. 61860206014), the Hunan Provincial Science and Technology Research Foundation of China (Grant No. 2019RS1003), the Australian Research Council through Grant (Grant Nos. DP200101197 and DE210100274) and the State Key Laboratory of Synthetical Automation for Process Industries for their funding support.

References

[1] K. H. Thung and C. Y. Wee, "A brief review on multi-task learning," *Multimedia Tools and Applications*, vol. 77, no. 22, pp. 29 705–29 725, 2018.

[2] Y. Zhang and Q. Yang, "An overview of multi-task learning," *National Science Review*, vol. 5, no. 1, pp. 30–43, 2018.

[3] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 3, pp. 397–415, 2008.

[4] S. Wen, H. Wei, Z. Yan, Z. Guo, Y. Yang, T. Huang, and Y. Chen, "Memristor-based design of sparse compact convolutional neural network," *IEEE Transactions on Network Science and Engineering*, 2019.

[5] I. Kokkinos, "Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6129–6138.

[6] S. Wen, W. Liu, Y. Yang, T. Huang, and Z. Zeng, "Generating realistic videos from keyframes with concatenated gans," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 8, pp. 2337–2348, 2018.

[7] S. Subramanian, A. Trischler, Y. Bengio, and C. J. Pal, "Learning general purpose distributed sentence representations via large scale multi-task learning," *arXiv preprint arXiv:1804.00079*, 2018.

[8] S. Xu, Y. Qian, and R. Q. Hu, "Data-driven edge intelligence for robust network anomaly detection," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, pp. 1481–1492, 2020.

[9] Z. Huang, J. Li, S. M. Siniscalchi, I.-F. Chen, J. Wu, and C.-H. Lee, "Rapid adaptation for deep neural networks through multi-task learning," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[10] F. Geyer and S. Bondorf, "Graph-based deep learning for fast and tight network calculus analyses," *IEEE Transactions on Network Science and Engineering*, pp. 1–1, 2020.

[11] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3994–4003.

[12] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. Feris, "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5334–5343.

[13] M. Long, Z. Cao, J. Wang, and S. Y. Philip, "Learning multiple tasks with multilinear relationship networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 1594–1603.

[14] Y. Yang and T. M. Hospedales, "Trace norm regularised deep multi-task learning," in *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[15] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7482–7491.

[16] Z. Chen, V. Badrinarayanan, C. Y. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *International Conference on Machine Learning*. PMLR, 2018, pp. 794–803.

[17] D. Zhou, J. Wang, B. Jiang, H. Guo, and Y. Li, "Multi-task multi-view learning based on cooperative multi-objective optimization," *IEEE Access*, vol. 6, pp. 19 465–19 477, 2017.

[18] J. Zhou, J. Chen, and J. Ye, "Clustered multi-task learning via alternating structure optimization," *Advances in Neural Information Processing Systems*, vol. 24, pp. 702–710, 2011.

[19] P. B. de Miranda, R. B. Prudêncio, A. C. P. de Carvalho, and C. Soares, "Combining a multi-objective optimization approach with meta-learning for svm parameter selection," in *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2012, pp. 2909–2914.

[20] C. Li, M. Georgiopoulos, and G. C. Anagnostopoulos, "Pareto-path multitask multiple kernel learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 51–61, 2014.

[21] J. A. Désidéri, "Multiple-gradient descent algorithm (mgda) for multiobjective optimization," *Comptes Rendus Mathematique*, vol. 350, no. 5-6, pp. 313–318, 2012.

[22] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[23] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, 2016.

[24] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.

[25] M. Jaggi, "Revisiting frank-wolfe: Projection-free sparse convex optimization," in *Proceedings of the 30th International Conference on Machine Learning*, no. CONF, 2013, pp. 427–435.

[26] Z. Xu, W. Yang, A. Meng, N. Lu, H. Huang, C. Ying, and L. Huang, "Towards end-to-end license plate detection and recognition: A large dataset and baseline," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 255–271.

[27] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[28] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.

Xiaojun Zhou (Member, IEEE) received his Bachelor's degree in Automation in 2009 from Central South University, Changsha, China and received the PhD degree in Applied Mathematics in 2014 from Federation University Australia. He is currently an Associate Professor in School of Automation, Central South University, Changsha, China. His main interests include uncertain optimization, multi-criteria optimization and decision-making, interpretable machine learning, and their applications in complex industrial processes.

Yuan Gao received her Bachelor's degree in Automation in 2019 from Xiangtan University, Xiangtan, China and she is currently a master student at Central South University, Changsha, China. Her main interests include multi-objective optimization, multi-task learning, optimization and control of complex industrial process.

Chaojie Li (Member, IEEE) received the B.Eng. and M.Eng. degrees from Chongqing University, Chongqing, China, in 2007 and 2011, respectively, and the Ph.D. degree from the School of Engineering, RMIT University, Melbourne, VIC, Australia, in 2017.

His current research interests include graph representation learning, distributed optimization and control in smart grid, neural networks, and their application.

Zhaoke Huang received the PhD degree in Control Science and Engineering in 2019 from Central South University, Changsha, China. He is currently a Postdoctoral Fellow in School of Automation, Central South University, Changsha, China. His main research interests include industrial big data analysis, data mining, optimization methods and their applications.