Reinforced Knowledge Distillation for Time Series Regression

Qing Xu, Keyu Wu, Min Wu, *Senior Member, IEEE,* Kezhi Mao, Xiaoli Li, *Senior Member, IEEE,* and Zhenghua Chen *Senior Member, IEEE*

***Abstract*—As one of the most popular and effective methods in model compression, knowledge distillation (KD) attempts to transfer knowledge from single or multiple large-scale networks (i.e., *Teachers*) to a compact network (i.e., *Student*). For the multi-teacher scenario, existing methods either assign equal or fixed weights for different teacher models during distillation, which can be inefficient as teachers might perform variously or even oppositely on different training samples. To address this issue, we propose a novel reinforced knowledge distillation method with negatively correlated teachers which are generated via negative correlation learning. The negatively correlated teachers would encourage teachers to learn different aspects of data and thus the ensemble of them can be more comprehensive and suitable for multi-teacher KD. Subsequently, a reinforced KD algorithm is proposed to dynamically employ proper teachers for different training instances via Double Deep Q-Network (DDQN). Our proposed method complements the existing KD procedure on teacher generation and selection. Extensive experimental results on two real-world time series regression tasks clearly demonstrate that the proposed approach could achieve superior performance over state-of-the-art methods. The PyTorch implementation of our proposed approach is available at https://github.com/xuqing88/RL-KD-for-time-series-regression.**

***Index Terms*—Reinforcement learning, knowledge distillation, time series regression.**

I. Introduction

Time series regression has many potential use-cases in internet of things (IoT), such as indoor localization [1], occupancy estimation [2] and remaining useful life prediction [3], [4]. Recently, deep neural networks (DNNs) have exhibited remarkable capabilities in time series regression tasks. However, those DNNs with state-of-the-art (SOTA) performance often consist of billions of model parameters, which cannot be fully utilized considering that time series regression algorithms generally require to be deployed on IoT devices with limited computational resources. To address this issue, various cutting-edge deep model compression techniques have been developed to improve model efficiency without significantly compromising model accuracy, *e.g.,* network pruning and quantization [5], Network Architecture Search (NAS) [6] and Knowledge Distillation [7]. Among them, KD related approaches have drawn enormous attentions due to their effectiveness and flexibility on transferring knowledge from a complex teacher model to a compact student model.

In most of existing KD approaches, the knowledge either comes from a single teacher [8] or comes from an ensemble of multiple teachers [7]. For single teacher distillation scenario, it is common to employ a pre-trained large scale model with SOTA performance as the teacher. However, numerous empirical experiments have shown that a stronger teacher does not necessarily stand for a better teacher [9] due to the significant capacity gap between it and the compact student. Therefore, selecting a proper teacher could be very time-consuming and tedious in many applications.

Alternatively, one can use the ensemble of multiple teachers to omit the teacher selection procedure. On the one hand, it is common to train several models independently [10] or sequentially [11] to get multiple teachers and then combine them together as the ensemble. One limitation of these two methods is that there is no feedback from the combination stage to the individual training stage, which limits their performance. Moreover, the success of distilling knowledge for cross-modal data [12], [13] and multi-view data [14] in recent works have shown the necessity of diversifying teachers in the ensemble as it would provide different views of data for the student. On the other hand, simply taking average [7] or using fixed weights [14], [15] over all models in the combination stage could also be inefficient. The teacher models would generally converge to different local minima due to random initialization, different optimization trajectories or other factors. It is possible that some teachers would contribute less or even have the negative contributions to the performance on certain instances [16]. Hence, in order to achieve better distillation performance, it is necessary to dynamically select proper teachers for different instances during the entire distillation process.

Inspired by the above insights, we systematically study how to generate diverse teachers via designing a negative correlation learning (NCL) strategy and proposing a reinforced knowledge distillation approach to dynamically select different teachers during distillation via reinforcement learning. We verify the effectiveness of our approach on time series regression tasks which are commonly needed in flexible model compression techniques. In particular, we follow the same problem setup as [17], [18] and target at effectively trans-

Qing Xu is with Institute for Infocomm Research, A*STAR, Singapore and the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore (Email: Xu_Qing@i2r.a-star.edu.sg).

Keyu Wu, Min Wu and Zhenghua Chen are with Institute fro Infocomm Research, A*STAR, Singapore (Email: Wu_Keyu@i2r.a-star.edu.sg, wumin@i2r.a-star.edu.sg, chen0832@e.ntu.edu.sg).

Kezhi Mao is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore (Email: EKZMao@ntu.edu.sg).

Xiaoli Li is with Institute for Infocomm Research, A*STAR, Singapore and the School of Computer Science and Engineering, Nanyang Technological University, Singapore (Email: xlli@i2r.a-star.edu.sg).

ferring knowledge between disparate network architectures (*e.g.,* from LSTM-based teachers to a CNN-based student). We expect the CNN-based student to be capable of capturing the temporal information buried in the sequential data like LSTM but with less model complexity like CNN, thereby more friendly for deployment.

Our contributions are summarized as follows:

- We propose a reinforced KD approach based on reinforcement learning to dynamically select proper teachers for different instances, which significantly enhances the efficacy of multi-teacher KD.
- We propose to generate highly diverse teachers for efficient learning in multi-teacher KD via a NCL module.
- Extensive experiments on two time series regression tasks have demonstrated that the proposed method significantly outperforms other SOTA approaches.

II. Related Work

Hinton *et al.* was the first to term the soften logits of a cumbersome model as knowledge and then distilled it to a compact model [7]. Romero *et al.* further extended the knowledge from soften logits to the output of intermediate layers (i.e., feature distillation) in [8]. After that, various KD methods, such as [19], [20], [21], [17], [18], have been proposed to distill knowledge from a single teacher in various applications. However, the selection of teacher model is not explicitly stated in these works and it is commonly based on personal prior experience.

Considering the fact that better performance is often achieved by the ensemble of DNNs, several multi-teacher KD approaches have recently been proposed. Intuitively, taking average of outputs of all teachers is a good choice and has proved to be effective in [7], [22], [12]. Besides, Chebotar and Water adopted fixed weights for different teachers and used the grid search to identify the optimal weights [15]. Wu *et al.* also employed fixed weights for a three-teacher distillation on a video action recognition task but did not clearly state how to determine these weights [14]. Nonetheless, simply combining all teaches with equal or fixed weights is insufficient as each teacher may perform very differently from other teachers on certain samples. Fukuda *et al.* proposed two strategies for multi-teacher KD scenario. One is called switched-training which randomly selects a model from teachers' pool to guide student's training, the other one is called augmented-training which respectively updates student's parameters with the outputs of each teacher [23]. Our experimental results show that the switched-training method even underperforms the averaging ensemble and the augmented-training could be regarded as an extreme case of our proposed methods. Our proposed method is able to adaptively select proper teachers for different training instances and thus significantly boost student's performance via distillation.

Another research hot-spot is to combine reinforcement learning (RL) with knowledge distillation. Fan *et al.* proposed a policy gradient-based RL method to assign different weights to samples within homogeneous classes during distillation to handle the problems of multi-class imbalanced classification [24]. Wang *et al.* utilized the RL strategy to fine-tune the structure of a generative model, which is first simplified via the KD, to enhance the diversity of generated molecules [25]. Tsantekidis *et al.* trained the RL-based financial trading teacher agents under different environments and then distilled the knowledge to a student to improve its training reliability [26]. Liang *et al.* presented a RL-based framework to select appropriate unlabeled samples from target language when transferring knowledge from the source language [27]. Our work differs from all the aforementioned works in terms of how to integrate RL techniques and KD approaches.

One highly related work to ours is [28], which tried to assign different weights to multiple teacher models during knowledge transferring to the student. However, our work is different from it in the following aspects. Firstly, our problem formulation is totally different, not only in terms of the definition of state and reward, but also in the reinforced learning strategy. Specifically, Yuan *et al.* [28] adopted the standard policy gradient method while we propose an off-policy DQN algorithm to optimize the RL model. Our method is able to achieve more sample efficiency through reusing the collected experiences. Moreover, the Monte-Carlo based policy gradient method adopted in [28] uses an estimation of gradient generated by only a series of data during each gradient update. Therefore, their noise estimation can adversely affect the learning stability, while our DQN-based method can lead to more stable performance via target network updates. Secondly, to the best of our knowledge, we are the first to exploit negative correlation learning to generate highly diverse teachers for the multi-teacher KD scenario. With the proposed RL-based teacher selection method, the negatively correlated teachers are more efficient in student's learning than other types of teachers, such as independent teachers used in [28].

III. Methodology

In this section, we first introduce the negatively correlated teachers generated by designing a NCL module. Then, a RL-based teacher selection method is presented. Lastly, the distillation is performed between the selected teachers and the student. The overall framework for our proposed approach is illustrated in Fig. 1.

A. Negatively Correlated Teachers

As aforementioned, for independent-trained and sequential-trained models, there is no interaction between teacher models during training stage, as well as no feedback from the combination stage. Contrarily, negative correlation learning strategy [16] emphasises the interaction and cooperation among individual networks and encourages each network to learn different aspects of training data by introducing a penalty term into individual network training process. Specifically, for the i^{th} network in the ensemble, the loss function is defined as Equation (1). The first term is the empirical training loss function, *e.g., Mean Square Error*, between the ground truth y and the prediction $\hat{y}_i$ from the i^{th} teacher. The second term is the correlation penalty and λ is a hyper-parameter to adjust

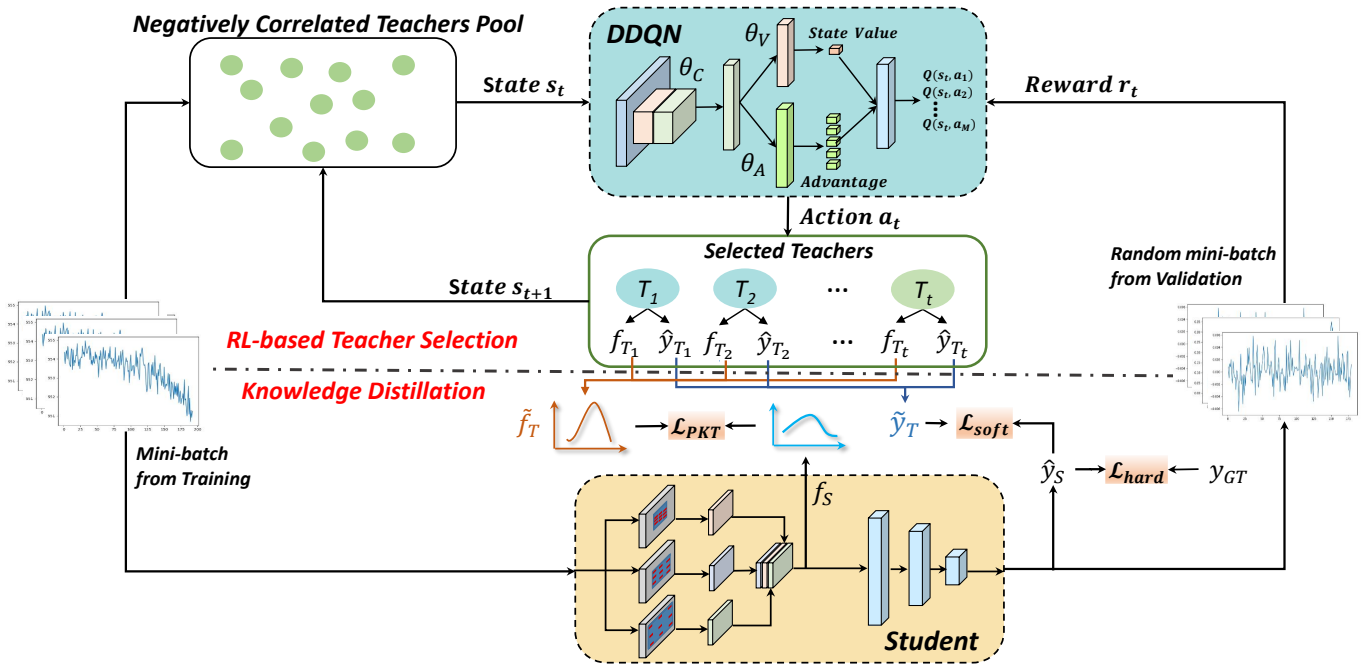


Fig. 1: The proposed RL-based KD framework. The upper part illustrates the RL-based teacher selector. We sample out one teacher from the pool at each step and aggregate them for knowledge transfer. The lower part is the knowledge distillation process. With the selected teachers, the student network is optimized via minimizing the loss which is the combination of PKT loss, soft loss and hard loss.

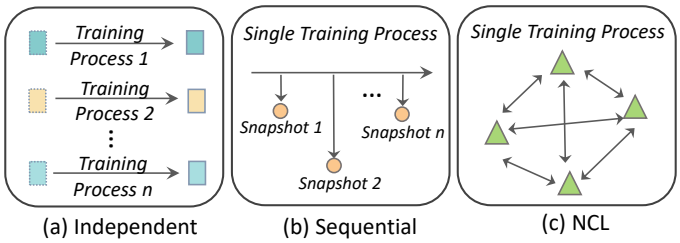


Fig. 2: Different Methods to get teacher models: (a) Each teacher is independently trained; (b) Teachers are sequential snapshots over single training process; (c) Teachers are simultaneously trained via negative correlation learning.

the strength of the correlation penalty. Here, $\tilde{y}$ is the average of predictions of all models in the ensemble.

$$\mathcal{L}_i = (\hat{y}_i - y)^2 + \lambda * (\hat{y}_i - \tilde{y}) * \sum_{j \neq i} (\hat{y}_j - \tilde{y}). \quad (1)$$

By minimizing the penalty term, the individual's error is negatively correlated to the errors of the rest in the ensemble, so that the diversity of the ensemble would be increased. Our experimental results show that comparing with teachers via independently or sequentially training, negatively correlated teachers are more knowledgeable for transferring the comprehensive knowledge to a student. Fig. 2 illustrates the above mentioned three different methods to generate multiple teachers.

B. RL-based Teacher Selection

When multiple teachers are available in teacher-student learning paradigm, since each individual teacher model may have inconsistent performance on different samples or mini-batches, dynamically employing proper teachers on instance level or mini-batch level could provide better understandings for specific data and thus effectively boost student's performance. Fig. 1 illustrates the overview of the proposed RL-based teacher selection process. Particularly, we utilize the Double Deep Q-Network (DDQN) [29] to obtain the optimal teacher selection policy for every training batch and sample out one teacher from the negatively correlated teachers pool at each step. Then the selected teacher at the current step participates in the knowledge distillation stage together with the other teachers sampled out at previous steps. The averaged outputs of intermediate layer and last layer of the selected teachers are regarded as the knowledge to guide student's training. Algorithm 1 presents the details of the proposed RL-based teacher selection process. In the following, the definitions of the state, action, reward and the optimization of DDQN are introduced in details.

State. In our proposed method, the state $s_t \in \mathbb{R}^{n_b \times k \times M}$ at any step $t \in [1, K]$, is formulated as the aggregation of feature maps from the intermediate layer of all negatively correlated teachers. Here, n_b is the number of samples in a mini-batch $\mathcal{D}_b \in \mathcal{D}$, k is the dimension of flattened feature maps from LSTM-based feature extractor, and M is the total number of negatively correlated teachers in the pool. Note that after a teacher is sampled out from the pool based on the optimal policy, the next state s_{t+1} is generated by zeroing out

the corresponding feature maps in s_t. To save computational cost, we only perform one-time inference for all negatively correlated teachers in mini-batch $\mathcal{D}_b$ as the initial state and conduct the zeroing-out operation at each step.

Action. For a specific teacher i in the pool, there are only two actions $a \in \{0, 1\}$ associated with it, selecting or not for current mini-batch. Specifically, $a_t^i = 1$ means to select teacher i at current step t and $a_t^i = 0$ means not to select it. Moreover, since we only select one teacher at each step and the output of the DDQN is a set of Q-values, the optimal action a^* at state s_t is then determined by Equation (2):

$$a^* = \underset{a}{\operatorname{argmax}}\, Q(s_t, a; \Theta_q). \tag{2}$$

Note that the argmax operation will only be applied to the remaining teachers in the pool.

Reward. The reward plays a crucial role on the learning of teacher selection policy as it could provide important feedback to DDQN for the value of choosing a particular action in current state. Motivated by [28], we use the student's performance on validation set to calculate the reward. However, we measure the linearized relative improvement of student's performance before and after distillation on a random batch from validation set every step as follows:

$$r = \max(-1, \min(\tanh(4 * \frac{P' - P}{P'}), 1)). \tag{3}$$

Here, P' and P represent the student's performance, *e.g.,* Root Mean Square Error (RMSE), before and after optimization with KD at one training step, respectively. We adopt the $tanh$ function to linearize the relative improvement on a random validation batch. Note that a positive value of $tanh$ means the performance get improved (*e.g.,* smaller RMSE). On the other hand, it is possible that the numerical value of relative improvement on RMSE is very small on a single optimization step due to a small learning rate. Therefore, we multiply the relative improvement by a coefficient to assign a good reward even when the improvement is marginal. What's more, we bound the reward between -1 and 1 so that the DDQN would be able to better converge.

Optimization of DDQN. The double DQN consists of two deep Q-networks, i.e., an online network $\mathcal{Q}$ with parameters Θ_q and a target network $\mathcal{Q}'$ with parameters Θ_q^-. $\mathcal{Q}$ and $\mathcal{Q}'$ share identical architecture and the initial Θ_q^- is a copy of Θ_q. As shown in Fig. 1, the DDQN network consists of three sub-network: a common CNN encoder with parameters θ_C which takes a 3-Dimension state matrix as input and outputs a vector, two separate streams (i.e., two stacked fully connected layers parameterized with θ_V and θ_A, respectively) to estimate the state-value $V(s; \theta_C, \theta_V)$ and advantages $A(s, a; \theta_C, \theta_A)$ for each action. Finally, the two streams are aggregated by Equation (4) to produce Q-values:

$$\begin{aligned} Q(s, a; \theta_C, \theta_V, \theta_A) =& V(s; \theta_C, \theta_V) + A(s, a; \theta_C, \theta_A) \\ & - \frac{1}{2} \sum_{a'} A(s, a'; \theta_C, \theta_A). \end{aligned} \tag{4}$$

Algorithm 1 RL-based Teacher Selection

Input: Negatively correlated teachers pool with M models. Student with Θ_s. An online network $\mathcal{Q}$ and a target network $\mathcal{Q}'$ initialized with Θ_q and Θ_q^-, respectively; Training data $\mathcal{D}$ and validation data $\mathcal{V}$; Epoch number L; Step number K; Replay Buffer $\mathcal{H}$.

for epoch $l \in [1, L]$ **do**
  Randomly shuffle $\mathcal{D}$
  for each mini-batch $\mathcal{D}_b \in \mathcal{D}$ **do**
    Calculate the initial state for $\mathcal{D}_b$
    for step $t \in [1, K]$ **do**
      Get state s_t and sample action $a_t \sim \mathcal{Q}(s_t)$.
      Pick out a teacher from the pool based on a_t.
      Update next state s_{t+1}.
      Update Θ_s with selected teachers via KD.
      Calculate r_t using a random batch from $\mathcal{V}$.
      Set $\beta = 0$ if step end, otherwise $\beta = 1$
      Store the tuple (s_t, a_t, r_t, s_{t+1}, β) to $\mathcal{H}$.
    end for
  end for
  Update $\mathcal{Q}$ and $\mathcal{Q}'$ via Algotithm 2.
end for

Algorithm 2 presents the details of the optimization process for the DDQN. To be specific, a random batch of entries (s_t, a_t, r_t, s_{t+1}, β) is sampled out from replay buffer $\mathcal{H}$. Then, Θ_q is optimized via minimizing the Huber loss between the estimated Q-values and the target Q-values which are calculated from Equation (5):

$$Q_{tar} = r_t + \beta * \gamma * Q(s_{t+1}, \underset{a_{t+1}}{\operatorname{argmax}}\, Q(s_{t+1}, a_{t+1}; \Theta_q); \Theta_q^-). \tag{5}$$

Algorithm 2 Optimization of DDQN

Input: An online network $\mathcal{Q}$ and a target network $\mathcal{Q}'$ initialized with Θ_q and Θ_q^-; Replay Buffer $\mathcal{H}$.

1: Randomly sample a mini-batch of (s_t, a_t, r_t ,s_{t+1}, β) from $\mathcal{H}$.
2: Calculate the estimated Q-value $Q(s_t, a_t; \Theta_q)$ via Equation (4).
3: Calculate the target Q-value via Equation (5).
4: Calculate the Huber loss between the estimated Q-value and target Q-value.
5: Update Θ_q via minimizing the Huber loss.
6: Update Θ_q^- via Equation (6).

Here, $\beta \in \{0, 1\}$, $\beta = 0$ means next step $t+1$ is the terminal step and 1 means the opposite. $\gamma \in [0, 1]$ is the discount factor that provides a trade-off between the immediate and future rewards. Lastly, the parameters Θ_q^- of target network $\mathcal{Q}'$ is updated via a moving average as shown in Equation (6):

$$\Theta_q^- \leftarrow \delta * \Theta_q^- + (1 - \delta) * \Theta_q. \tag{6}$$

C. Knowledge Distillation

LSTM-based networks have demonstrated superior performance over CNN-based networks on time series data analytics [30]. Yet, they are generally more complex than CNN-based networks. In this work, we intend to distill knowledge from a powerful and complex LSTM to an efficient CNN for time series regression. Here, the knowledge from teachers are formulated as two parts: feature maps from the intermediate layers and final predictions. In particular, to transfer the knowledge from intermediate layers, we adopt the Probabilistic Knowledge Transfer (PKT) [20] to match the probability distribution of the data in the feature space between teacher and student.

For multiple teachers scenario, we first average the feature maps over all selected teachers and then calculate the PKT loss by measuring the Kullback-Leibler (KL) divergence between two distributions as Equation (7).

$$\mathcal{L}_{PKT} = D_{KL}(P\|Q) = \sum_{i=1}^{n_b} \sum_{j=1, j\neq i}^{n_b} p_{j|i} \log(\frac{p_{j|i}}{q_{j|i}}), \quad (7)$$

where $p_{j|i}$ and $q_{j|i}$ are the conditional probability distributions of the teacher and student with respect to sample x_i and x_j in the mini-batch. Equation (8) presents the calculation of $p_{j|i}$. Here, $\widetilde{f}_{x_i}$ and $\widetilde{f}_{x_j}$ are the averages of flattened feature maps over all selected teachers with sample x_i and x_j as inputs. $cos(\widetilde{f}_{x_i}, \widetilde{f}_{x_j})$ measures the biased cosine similarity between $\widetilde{f}_{x_i}$ and $\widetilde{f}_{x_j}$ as shown in Equation (9). Comparing with the general calculation of cosine similarity, the biased cosine similarity shifts the value to make sure that the conditional probability distribution is positive.

$$p_{j|i} = \frac{cos(\widetilde{f}_{x_i}, \widetilde{f}_{x_j})}{\sum_{k=1, k\neq j}^{n_b} cos(\widetilde{f}_{x_k}, \widetilde{f}_{x_j})}, \quad (8)$$

$$cos(\widetilde{f}_{x_i}, \widetilde{f}_{x_j}) = \frac{1}{2}\left[\frac{(\widetilde{f}_{x_i})^T(\widetilde{f}_{x_j})}{||\widetilde{f}_{x_i}||_2||\widetilde{f}_{x_j}||_2} + 1\right]. \quad (9)$$

By minimizing the $\mathcal{L}_{PKT}$, we encourage the student to learn feature representation with similar probability distribution as the teachers.

Meanwhile, we also utilize teachers' predictions as the soft labels and define the soft loss $\mathcal{L}_{soft} = \frac{1}{n_b}\sum_1^{n_b}(\widetilde{y}_T - \hat{y}_S)^2$ as the distance between the averaged predictions $\widetilde{y}_T$ from all selected teachers and the prediction of student $\hat{y}_S$. In addition to the above two distillation losses, a general loss between student's prediction $\hat{y}_S$ and ground truth y is also adopted in student's training process, denoted as $\mathcal{L}_{hard} = \frac{1}{n_b}\sum_1^{n_b}(\hat{y}_S - y)^2$. The final loss is defined as the combination of above three losses as follows:

$$\mathcal{L} = \alpha_1 * \mathcal{L}_{hard} + \alpha_2 * \mathcal{L}_{soft} + \alpha_3 * \mathcal{L}_{PKT}, \quad (10)$$

where α_1, α_2 and α_3 are hyper-parameters to balance the three losses. Fig. 3 illustrates the changing curve of α_1, α_2 and α_3 over different training epochs. Particularly, the weight α_3 for PKT loss is gradually decreased while the weight α_1, α_2 for hard loss and soft loss are increased during the whole training process. The summation of α_1, α_2 and α_3 always equals to 1 at any training step. The motivation is that: at the early training stage of student, the feature maps of the selected teachers could provide more information than a scalar, like soft label and ground truth. It would help the student to learn similar feature representations as the teachers first. As the training goes on, the weights of soft loss and hard loss would gradually increase, which allows the student to focus more on the regression target.

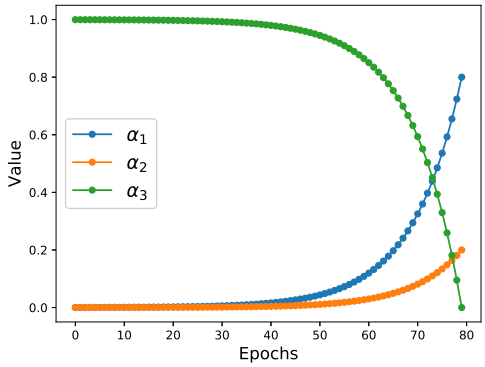


Fig. 3: Values of α_1, α_2 and α_3 at different training epochs.

IV. Experiments

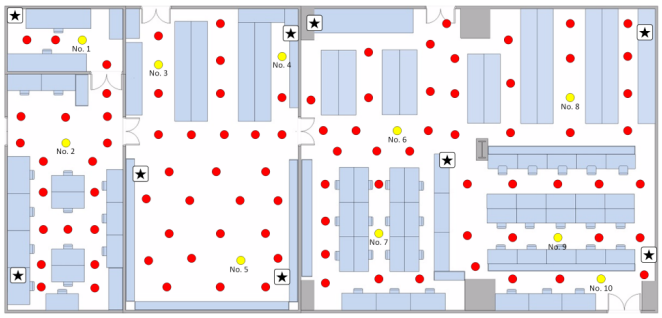
In this section, we evaluate the effectiveness of our proposed approach on two real-world time series regression tasks, i.e., machine remaining useful life (RUL) prediction and indoor localization, which consists of six public datasets.

A. Experimental Setup

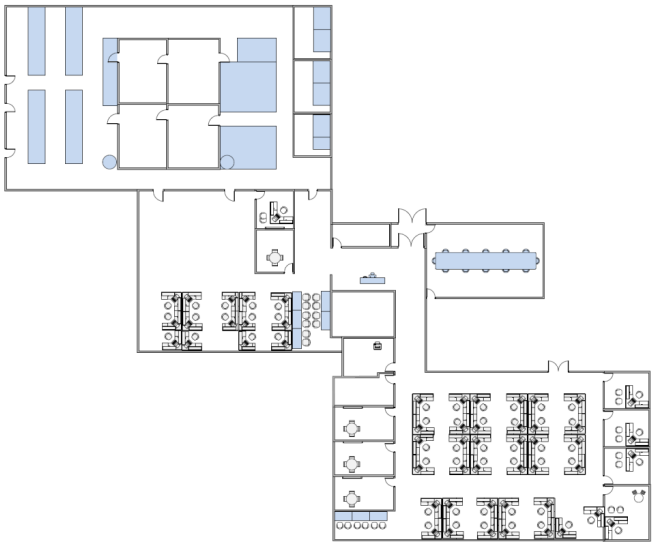
1) Machine RUL Prediction: We adopt the NASA's turbofan engine data for the RUL prediction task, which is also known as C-MAPSS dataset [31]. The objective is to accurately estimate the RUL of turbofan engines via 21 time series sensor measurements. It consists of four datasets, namely, FD001, FD002, FD003 and FD004. The operating conditions and fault modes of engines in each dataset are different, as shown in Table I. Each dataset contains several training and test trajectories. Each trajectory in training set represents a turbofan engine with different initial states and contains 21 time-series sensor measurements recording engine degradation process. On the contrary, the trajectories in test sets are the engine measurements at certain degradation stage. The objective of this dataset is to accurately estimate the RUL of turbofan engines via these sensor measurements. In our experiments, the training sets are further divided into training and validation set with a ratio of $9:1$ in terms of trajectories. For instance, we randomly select 90 trajectories from FD001 for model training and the rest 10 trajectories for validation. Then, the same data pre-processing method has been applied to training, validation and test sets as [18]. Moreover, same as previous works [17], [30], [32], the root mean square error

TABLE I: Summary of C-MAPSS Dataset

Dataset	FD001	FD002	FD003	FD004
Operating Conditions	1	6	1	6
Fault Modes	1	1	2	2
Train Traj.	100	260	100	249
Test Traj.	100	259	100	248



(a) Lab



(b) Office

Fig. 4: Layout of two indoor localization environments.

(RMSE) and Score function are employed as the evaluation metrics, which are defined as

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}, \tag{11}$$

$$Score = \begin{cases} \sum_{i=1}^{N}(e^{-\frac{\hat{y}_i - y_i}{13}} - 1), & \text{if } \hat{y}_i < y_i, \\ \sum_{i=1}^{N}(e^{\frac{\hat{y}_i - y_i}{10}} - 1), & \text{otherwise}, \end{cases} \tag{12}$$

where $\hat{y}_i$ and y_i are the predicted RUL and the ground truth RUL, and N is the total number of samples. The lower RMSE and Score are, the better performance the model would achieve. Particularly, the Score function places more penalty on late prediction than early prediction, as the late prediction may cause worse catastrophe in real world applications.

2) Indoor localization: Another typical time series regression task that we utilized to evaluate our method is indoor localization with WiFi RSSI (Received Signal Strength Indicator) values [33]. It consists of two datasets collected under different settings. Specifically, the dataset was collected by using off-the-shelf WiFi routers in two different environments: a research lab (35.3 m $\times$ 16.0 m) with 9 WiFi routers and an office (55.0 m $\times$ 50 m) with 20 WiFi routers as depicted in Fig.4. To be specific, 102 reference points are selected in the research lab, which are uniformly distributed. At each point, 2000 consecutive RSSI values are collected from all 9 routers (star symbols in Fig. 4(a)). Then, data from 92 randomly selected reference points (red circles) are used for model training and the rest 10 reference points (yellow circles) for testing. In the office, 353 uniformly distributed reference points are selected and 30 of them are randomly selected for testing and the rest for training. More detailed discriptions of the dataset can be found in [33]. We adopt mean localization error (MLE) as the evaluation metric following [33]. The MLE is defined as

$$MLE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}[(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2]}, \tag{13}$$

where $(\hat{x}_i, \hat{y}_i)$ are predicted location coordinates and (x_i, y_i) are the ground truth.

3) Model Implementation: For fair comparison with existing works, we adopt the same network architectures for LSTM-based teacher and CNN-based student as [17], [18] for RUL prediction task. Particularly, the teacher model consists of 5 LSTM layers (32 hidden units in each layer) as the feature extractor and 2 fully connected layers as the regressor. The student model consists of 3 parallel dilated CNN branches as the feature extractor and also 2 fully connected layers (but with less hidden units) as the regressor. For the indoor localization task, we employ a model consisting of 2-layer LSTM and 2 fully connected layers as the teacher (same as [33]) and use the same CNN-based student as RUL prediction task but only modify the last layer from single scalar output to a tuple output ((x, y) coordinates for the localization). Moreover, we employ a 3-layer CNN as the common encoder mentioned in Section III and 2 fully connected layers for each stream. Table II gives the details of the DDQN architecture. The output shape is calculated based on $M = 10$ and $n_b = 64$. "Conv2D(32,7,3)" refers to a 2-D convolutional operation with $output\_channels = 32$, $kennel\_size = 7$ and $stride = 3$. "FC(2034, 512)" refers to a fully connected layer with $input = 2034$ and $output = 512$. Layer 1 to 4 is the common CNN encoder. The left branch of layer 5 and 6 is the state-value stream and the right branch is the advantage stream. The final output is the aggregation of the two streams.

4) Training Details: To generate the negatively correlated teachers pool, we first initialize $M = 20$ teachers with different random seeds and then simultaneously train them with the NCL strategy. Specifically, following [16] we configure the correlation penalty weight $\lambda = 0.5$ and set $n_b = 64$, learning rate $lr = 1e-3$. The Adam optimizer is adopted during training. In RL-based KD stage, totally K teachers are selected

TABLE II: Network Details of DDQN

Layers	Operations		Output Shape
Input	-		(None, 10, 64, 64)
Layer 1	Conv2D(32,7,3) + Relu		(None, 32, 20, 20)
Layer 2	Conv2D(64,5,2) + Relu		(None, 64, 8, 8)
Layer 3	Conv2D(64,3,1) + Relu		(None, 64, 6, 6)
Layer 4	Flatten		(None, 2304)
Layer 5	FC(2034, 512) + Relu	FC(2034,512) + Relu	(None, 512), (None, 512)
Layer 6	FC(512, 1)	FC(512, 10)	(None, 1), (None, 10)
Output	Aggregation as Equation (4)		(None, 10)

TABLE III: Performance Comparison with Benchmark Approaches

Tasks		Machine RUL Prediction								Indoor Localization	
Scenario	Methods	FD001		FD002		FD003		FD004		Lab	Office
		RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score	MLE	MLE
No Teacher	Student Only	14.64	392.23	16.15	1281.07	15.34	602.15	17.38	1760.41	1.78	2.12
Single Teacher	Standard KD	13.82	320.4	15.59	1131.33	14.16	521.16	16.86	1549.98	1.76	1.95
	AT	13.91	344.05	15.38	1034.96	14.75	593.28	17.02	1570.05	1.70	2.01
	RKD-DA	13.67	309.69	15.18	1008.52	14.04	480.25	16.55	1418.40	1.73	2.02
	PKT	13.57	332.28	14.41	996.04	13.17	350.86	15.94	1291.87	1.66	1.92
	KDnet-RUL	13.68	362.08	14.47	929.2	12.95	327.27	15.96	1303.19	1.51	1.83
	CA-KD	13.41	293.82	14.23	975.96	12.95	325.29	15.85	1256.82	1.53	1.79
Multiple Teachers	Random-T	14.15	314.85	15.41	990.02	14.09	450.72	16.89	1590.01	1.58	1.85
	W-Averaging	13.76	309.22	15.56	1012.82	13.94	468.64	16.60	1460.99	1.60	1.89
	U-Averaging	13.60	306.59	15.27	983.48	14.12	477.57	16.42	1308.48	1.58	1.87
Reinforced Multi-Teacher	Independent	13.55	310.53	15.10	1054.82	13.72	426.76	16.53	1340.85	1.50	1.78
	Sequential	13.71	312.81	15.03	972.74	13.87	404.48	16.41	1385.85	1.52	1.84
	Proposed	**13.07**	**288.82**	**14.22**	**901.74**	**12.82**	**311.55**	**15.71**	**1241.31**	**1.47**	**1.76**

out for each mini-batch. Note that student's performance varies with the number of K selected teachers for each set. To be specific, we set $K = 3$ for FD001 and FD003 in machine RUL prediction task and indoor localization task, $K = 5$ for FD002 and FD004. More experimental results regarding the K will be presented in Section IV-D. The student is trained with $lr = 1e-3$ with a step decay schedule and Adam optimizer. For the DDQN optimization, we follow [29] and set $\gamma = 0.9$ and $\delta = 0.999$. The DDQN is trained with a batch size of 32, $lr = 1e-4$ and Adam optimizer.

B. Results and Discussion

1) Comparison with Benchmark Approaches: To verify the effectiveness of the proposed approach, extensive experiments have been conducted to compare our method with SOTA methods. Table III presents the evaluation results of different KD methods on two tasks. In particular, we compare student's performance under four scenarios: no teacher, single teacher, multiple teachers and reinforced multi-teacher. For no teacher case, we train the student with the ground truth only, denoted as "**Student Only**". We take it as the baseline to other KD methods. For the single teacher case, we employ the teacher with best performance on the validation set from our negatively correlated teachers pool to perform distillation with different methods. We compare three different feature distillation approaches including **AT** [19], **RKD-DA**[21] and **PKT** [20]. We also include the results from two recent SOTA works for C-MAPSS dataset targeting at transferring knowledge between disparate network architecture: **KDnet-RUL** [17] and **CA-KD** [18]. Meanwhile, we apply these two SOTA approaches in indoor localization task. Particularly, for **KDnet-RUL**, we follow [17] and apply sequential distillation upon identical architecture knowledge transferring. For **CA-KD**, we set the number of negative samples in contrastive learning to 512 same as [18]. Besides, we compare three widely used ensemble methods for multi-teacher scenario: randomly selecting one teacher from all teachers (denoted as **Random-T** [23]), using fixed weights for all teachers (denoted as **W-Averaging** [15]) and using uniform distributed weights for all teachers (denoted as **U-Averaging** [7]). For **W-Averaging**, we assign weights based on their performance on validation set. The better performance a teacher can achieve, the higher weight it will be assigned to. The summation of all weights equals to 1. From Table III, some observations are as follows.

First, all KD methods, either using single teacher or multiple teachers, could enhance student's performance in both machine RUL prediction task and indoor localization task. It indicates the effectiveness of KD in model compression.

Secondly, in single teacher KD scenario, among three different feature distillation methods, (*i.e.,* AT, RKD-DA and PKT), PKT outperforms other two feature distillation methods in terms of transferring knowledge from single teacher. Therefore, in our multi-teacher scenario, PKT is employed to match the probability distribution of the feature maps between student and the average of the selected teachers.

Besides, KDnet-RUL and CA-KD have superior performance than others for most of subsets in both tasks. One possible reason is that the knowledge transferring schemes in these two methods are specially designed for disparate network architectures. However, introducing the adversarial, sequential or contrastive learning strategies on feature distillation would significantly increase training efforts.

Thirdly, in multiple teachers KD scenario, the simple averaging strategy over all teachers (*i.e.*, U-Averaging) could lead to better performance than the other two ensemble methods (Random-T and W-Averaging) for most of subsets in both machine RUL prediction task and indoor localization task. In other words, assigning more weights to a stronger teacher based on its performance on the whole training samples does not necessarily improve student's performance. This observation reveals the importance of properly choosing 'decent' teachers on instance level.

Lastly, our proposed RL-based knowledge distillation method significantly and also consistently outperforms all aforementioned methods. It demonstrates the superiority of our method over other ensemble strategies and reveals that our method could help the student to better learn feature representations via properly selecting suitable teachers for different instances. Besides, our method eliminates the process of teacher selection based on prior experience, hence could be more efficient.

2) Different Teachers: Furthermore, we explicitly show how well three different types of teachers (as depicted in Fig. 2) could transfer the knowledge to a student. All these three types of teachers have exactly the same network architecture but with different training strategy. For the independent teachers, each individual was randomly initialized and trained independently. For the snapshot teachers, we adopted a cyclical learning rate schedule with warm restart method to improve the convergence rate and saved the intermediate network (snapshot) every 10 epochs along single training process. Other training parameters for independent and sequential teachers, like learning rate, optimizer and configuration of the RL-based KD stage, are the same as negatively correlated teachers for fair comparison. The last three rows in Table III show the student's performance with different types of teachers. On the one hand, comparing with other ensemble strategies the proposed RL-based teacher selection could also boost student's performance even with independent and sequential teachers. It reveals that our proposed RL-based teacher selection method can be applied to any other KD-related applications using multiple teachers. On the other hand, it is obvious that the students trained with negatively correlated teachers have consistently superior performance on all datasets. It indicates that the negatively correlated teachers are more suitable for our proposed RL-based KD framework.

C. Ablation Study

To investigate how each KD component contributes to student's final performance, we conducted the ablation study on machine RUL prediction task as shown in Fig. 5. Fig. 5(a)∼(d) show the RMSE and Score results from FD001 to FD004. "RL+KD" represents the student trained with ground truth labels and soft labels for KD. Similarly, "RL+FT" means that the student is trained with ground truth labels and feature distillation with PKT. The teachers in "RL+KD" and "RL+FT" are selected via the proposed RL-based teacher selection method. It can be clearly seen from Fig. 5 that each component in KD could boost student's performance. It is also not surprising that "RL+FT" outperforms "RL+KD" since the soft label in the regression task is just a scalar, which in general is a probability distribution in classification tasks. The knowledge provided by a scalar is very limited. However, the feature distillation and soft labels from teacher are complementary to each other. Combining them together could yield a better performance as our proposed method.

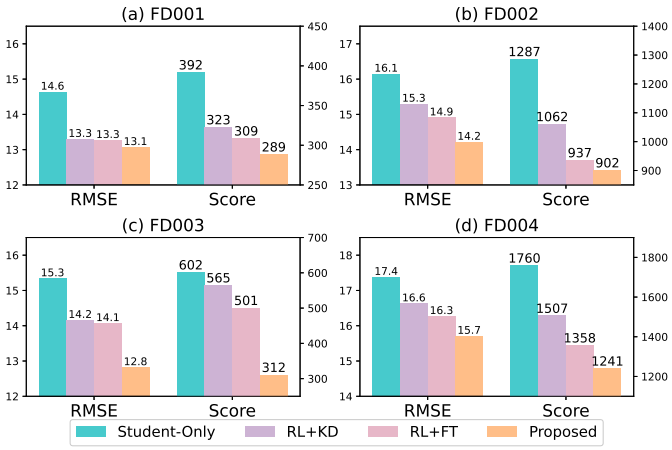


Fig. 5: Analysis of Different Components in the Proposed Method on C-MAPSS.

D. Sensitivity Analysis

One of the key parameters in our proposed reinforcement learning based KD method is the number of K teachers selected for each batch training. We conducted a detailed investigation on how the K value affects student's performance on machine RUL prediction task. The results for four subsets from C-MAPSS are shown in Fig. 6.

The optimal K for each subset are different. For simple datasets (*e.g.,* FD001 and FD003), a set of $K = 3$ teachers would produce good student. However, for the complex datasets (*e.g.,* FD002 and FD004), they need more teachers ($K = 5$) to participate in distillation stage. Furthermore, student's performance would gradually decrease when there are more teachers involving into distillation. The reason is that involving unknowledgable teachers for certain specific samples would introduce noise into student's learning process.

V. Conclusion

In this paper, we show that teachers trained via negative correlation learning are more comprehensive and suitable for multi-teacher KD scenario. A reinforcement learning based teacher selection approach is then proposed to dynamically select different teachers to participate in the distillation for

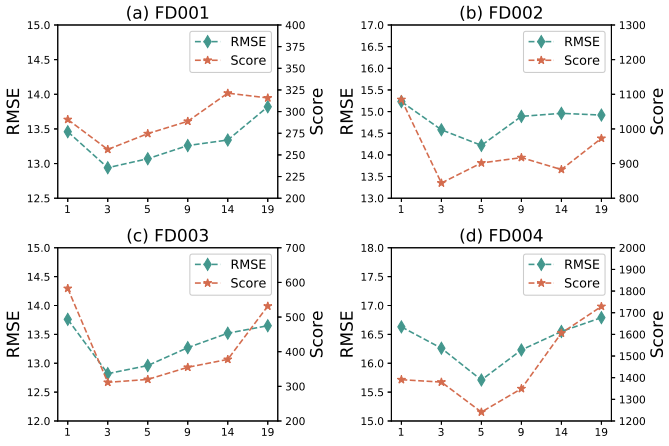


Fig. 6: Sensitivity of K Selected Teachers on C-MAPSS.

different training samples. The experiment results demonstrate that our proposed method significantly outperforms other SOTA KD methods and prove the effectiveness of our method for time series regression.

In our future work, we intend to explore more possibilities of combining reinforcement learning and KD techniques. Currently, we propose a RL strategy to iteratively select a few teachers from pools. Alternatively, we could also assign different weights for all teachers on different training instance via designing different RL strategies to fully utilize the knowledge learned by all teachers. Besides, considering that the network architecture of student models is generally pre-defined, RL-based method could also be employed to automatically optimize the structure of the student models.

REFERENCES

[1] X.-Y. Liu and X. Wang, "Real-time indoor localization for smartphones using tensor-generative adversarial nets," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3433–3443, 2020.

[2] S. Golestan, S. Kazemian, and O. Ardakanian, "Data-driven models for building occupancy estimation," in *Proceedings of the Ninth International Conference on Future Energy Systems*, 2018, pp. 277–281.

[3] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2306–2318, 2016.

[4] Y. Gao, Y. Wen, and J. Wu, "A neural network-based joint prognostic model for data fusion and remaining useful life prediction," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 117–127, 2020.

[5] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, "Pruning and quantization for deep neural network acceleration: A survey," *Neurocomputing*, vol. 461, pp. 370–403, 2021.

[6] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 1997–2017, 2019.

[7] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *NIPS Deep Learning and Representation Learning Workshop*, vol. 1050, p. 9, 2015.

[8] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *in International Conference on Learning Representations (ICLR)*, 2015.

[9] J. H. Cho and B. Hariharan, "On the efficacy of knowledge distillation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4794–4802.

[10] A. J. C. SHARKEY, "On combining artificial neural nets," *Connection science*, vol. 8, no. 3-4, pp. 299–314, 1996.

[11] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, "Snapshot ensembles: Train 1, get m for free," *arXiv preprint arXiv:1704.00109*, 2017.

[12] Y. Liu, K. Wang, G. Li, and L. Lin, "Semantics-aware adaptive knowledge distillation for sensor-to-vision action recognition," *IEEE Transactions on Image Processing*, 2021.

[13] Y. Tian, D. Krishnan, and P. Isola, "Contrastive representation distillation," *arXiv preprint arXiv:1910.10699*, 2019.

[14] M.-C. Wu, C.-T. Chiu, and K.-H. Wu, "Multi-teacher knowledge distillation for compressed video action recognition on deep neural networks," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2202–2206.

[15] Y. Chebotar and A. Waters, "Distilling knowledge from ensembles of neural networks for speech recognition." in *Interspeech*, 2016, pp. 3439–3443.

[16] Y. Liu and X. Yao, "Simultaneous training of negatively correlated neural networks in an ensemble," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 6, pp. 716–725, 1999.

[17] Q. Xu, Z. Chen, K. Wu, C. Wang, M. Wu, and X. Li, "Kdnet-rul: A knowledge distillation framework to compress deep neural networks for machine remaining useful life prediction," *IEEE Transactions on Industrial Electronics*, 2021.

[18] Q. Xu, Z. Chen, M. Ragab, C. Wang, M. Wu, and X. Li, "Contrastive adversarial knowledge distillation for deep model compression in time-series regression tasks," *Neurocomputing*, 2021.

[19] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," *5th International Conference on Learning Representations(ICLR)*, 2017.

[20] N. Passalis and A. Tefas, "Learning deep representations with probabilistic knowledge transfer," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 268–284.

[21] W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3967–3976, 2019.

[22] S. Park and N. Kwak, "Feature-level ensemble knowledge distillation for aggregating knowledge from multiple networks," in *ECAI 2020*. IOS Press, 2020, pp. 1411–1418.

[23] T. Fukuda, M. Suzuki, G. Kurata, S. Thomas, J. Cui, and B. Ramabhadran, "Efficient knowledge distillation from an ensemble of teachers." in *Interspeech*, 2017, pp. 3697–3701.

[24] S. Fan, X. Zhang, and Z. Song, "Reinforced knowledge distillation: Multi-class imbalanced classifier based on policy gradient reinforcement learning," *Neurocomputing*, vol. 463, pp. 422–436, 2021.

[25] J. Wang, C.-Y. Hsieh, M. Wang, X. Wang, Z. Wu, D. Jiang, B. Liao, X. Zhang, B. Yang, Q. He *et al.*, "Multi-constraint molecular generation based on conditional transformer, knowledge distillation and reinforcement learning," *Nature Machine Intelligence*, vol. 3, no. 10, pp. 914–922, 2021.

[26] A. Tsantekidis, N. Passalis, and A. Tefas, "Diversity-driven knowledge distillation for financial trading using deep reinforcement learning," *Neural Networks*, vol. 140, pp. 193–202, 2021.

[27] S. Liang, M. Gong, J. Pei, L. Shou, W. Zuo, X. Zuo, and D. Jiang, "Reinforced iterative knowledge distillation for cross-lingual named entity recognition," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 3231–3239.

[28] F. Yuan, L. Shou, J. Pei, W. Lin, M. Gong, Y. Fu, and D. Jiang, "Reinforced multi-teacher selection for knowledge distillation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 14 284–14 291.

[29] K. Wu, M. Wu, J. Yang, Z. Chen, Z. Li, and X. Li, "Deep reinforcement learning boosted partial domain adaptation," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Z.-H. Zhou, Ed. International Joint Conferences on Artificial Intelligence Organization, 8 2021, pp. 3192–3199. [Online]. Available: https://doi.org/10.24963/ijcai.2021/439

[30] Z. Chen, M. Wu, R. Zhao, F. Guretno, R. Yan, and X. Li, "Machine remaining useful life prediction via an attention based deep learning approach," *IEEE Transactions on Industrial Electronics*, pp. 1–1, 2020.

[31] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 international conference on prognostics and health management*. IEEE, 2008, pp. 1–9.

[32] G. S. Babu, P. Zhao, and X.-L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life,"

International conference on database systems for advanced applications, pp. 214–228, 2016.

[33] Z. Chen, H. Zou, J. Yang, H. Jiang, and L. Xie, "Wifi fingerprinting indoor localization using local feature-based deep lstm," *IEEE Systems Journal*, vol. 14, no. 2, pp. 3001–3010, 2019.