Semi-Supervised Knowledge Amalgamation for Sequence Classification

Jidapa Thadajarassiri, Thomas Hartvigsen, Xiangnan Kong, Elke Rundensteiner
Data Science Program and Computer Science Department, Worcester Polytechnic Institute
100 Institute Road, Worcester MA, USA 01609
{jthadajarassiri, twhartvigsen, xkong, rundenst}@wpi.edu

Abstract

Sequence classification is essential for domains from medical diagnosis to online advertising. In these settings, data are typically proprietary and annotations are expensive to acquire. Often times, so few annotations are available that training a robust model from scratch is impractical. Recently, knowledge amalgamation (KA) has emerged as a promising strategy for training models without this hard-to-come-by labeled training dataset. To achieve this, KA methods combine the knowledge of multiple pre-trained *teacher* models (trained on different classification tasks and proprietary datasets) into one *student* model that becomes an expert on the union of *all* teachers' classes. However, we demonstrate that the state-of-the-art solutions fail in the presence of *overconfident* teachers, which make confident but incorrect predictions for instances from classes upon which they were not trained. Additionally, to-date no work has explored KA for sequence models. Therefore, we propose and then solve the open problem of *semi-supervised* KA for sequence classification (SKA). Our SKA approach first learns to estimate how trustworthy each teacher is for a given instance, then rescales the predicted probabilities from all teachers to supervise a student model. Our solution overcomes overconfident teachers through careful use of a very small amount of labeled instances. We demonstrate that this approach beats eight state-of-the-art alternatives on four real-world datasets by on average 15% in accuracy with as little as 2% of training data being annotated.

Introduction

Background and Motivation. Sequence classification has a wide range of real-world applications. Examples include disease prediction (Razavian, Marcus, and Sontag 2016), activity recognition (Yang et al. 2020), and speech recognition (Graves, Jaitly, and Mohamed 2013). Conventional approaches to sequence classification mainly focus on supervised settings, which require a large amount of labeled data to train a model. Unfortunately, large datasets, such as health records or customer activities, are not always available to the public due to privacy concerns. Often too few annotations are available to train a robust model from scratch. Fortunately, some research groups may train a model on their private data (*e.g.*, medical records) and release the pre-trained models instead of releasing the training data to support reuse by other researchers or practitioners (Harutyunyan et al. 2019).

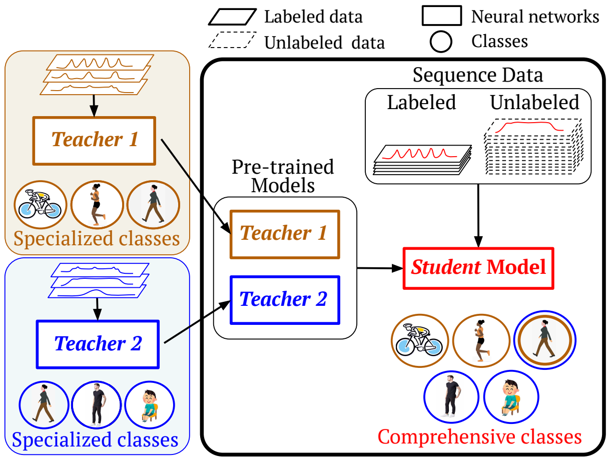
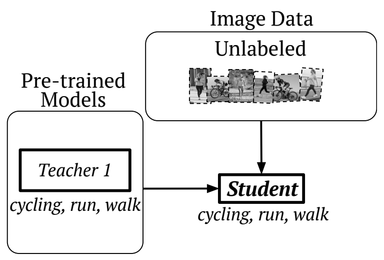


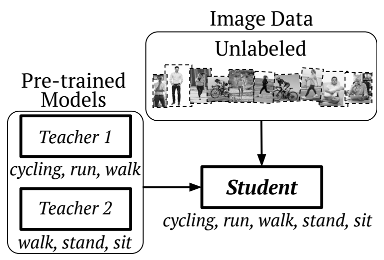
Figure 1: Semi-supervised knowledge amalgamation (SKA) for sequence classification. Given sequence data, a tiny subset of which is labeled, and a set of pre-trained models (*teachers*), the goal is to train a *student* model that can *amalgamate* the teachers' knowledge, learning to predict any of the classes in which the teachers specialize.

However, these pre-trained models may not be directly usable because (1) they may be too large to fit within the resources of some applications (e.g., sequence classification in mobile devices), or (2) there may be differences in the classification tasks of the teacher and student models (e.g., disease prediction with more disease classes than the pre-trained model). Many works have studied these two issues for image classification (Hinton, Vinyals, and Dean 2015; Shen et al. 2019a) while no works have yet studied such problems for sequence classification.

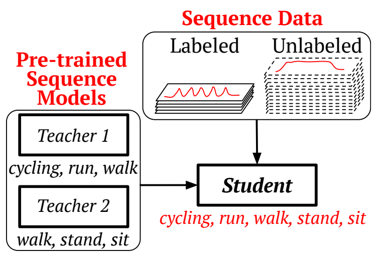
In image classification, Knowledge Distillation (KD) (Hinton, Vinyals, and Dean 2015) methods address problem (1) by training a small target model (the "*student*" model) that learns from *one* pre-trained, and possibly complex, model (the "*teacher*" model) with a set of unlabeled data. To address problem (2), Knowledge Amalgamation (KA) (Shen et al. 2019a) methods have been proposed to train a *student* model from *multiple* pre-trained *teacher* models, where different teacher models are pre-trained to classify distinct sets of classes. The student model in KA aims to cover the union



(a) Knowledge Distillation on Image Data (Hinton, Vinyals, and Dean 2015): learning from one teacher network on the same classification task.



(b) Unsupervised Knowledge Amalgamation on Image Data (Luo et al. 2019): learning from multiple teacher networks.



(c) Semi-supervised Knowledge Amalgamation on Sequence Data (this paper): learning from multiple teachers with access to a small set of labeled sequences.

Figure 2: Comparison of related problems. Each rectangular box represents a neural network model and the words *under* each box (e.g., "*cycling*" and "*run*") denote the classes that the model is trained for in the classification task.

of all classes specialized in by any of the teachers.

In this paper, we study the problem of KA for sequence classification. As an example of this setting, consider a human activity recognition problem to support health screening. To monitor patients' wellness, medical researchers may want to develop a model that can recognize patients' activities based on the sequence data collected by smart devices (Masum, Bahadur, and Ruhi 2020). As shown in Figure 1, suppose two pre-trained teacher models are released to the public: one teacher can detect *cycling*, *running* and *walking* classes from the patient's activities, while the other teacher detects *walking*, *standing* and *sitting*. If we want to build a model that can detect activities of all 5 classes, including *cycling*, *running*, *walking*, *standing*, and *sitting*, we need to learn a student model that can *amalgamate* the collective knowledge of these pre-trained teacher models.

State-of-the-Art. Conventional methods for knowledge amalgamation (KA) (Shen et al. 2019a,b; Ye et al. 2019; Luo et al. 2019; Vongkulbhisal, Vinayavekhin, and Visentini-Scarzanella 2019) focus on the unsupervised setting, where student models are trained by learning to imitate the teachers' outputs and/or intermediate layers from unlabeled data. For many sequence classification tasks, however, the classification performance of unsupervised KA methods suffers heavily when an overconfident teacher exists, which provides predictions that are inaccurate, yet have high confidence on instances about which it has no knowledge. For instance (Figure 1), due to similarities between *Cycling* and *Sitting*, Teacher 2 may be *overconfident* in predicting *Sitting* when a person is in fact *Cycling* because it is not trained to detect *Cycling*. Without access to any labels, this type of mistake is unavoidable. However, an overlooked opportunity is that in some sequence classification tasks we may be able to gain access to a small set of labeled data.

Problem Definition. In this paper, we thus propose the study of a new problem, called *semi-supervised knowledge amalgamation* (SKA) for sequence classification, as depicted in Figure 1. Given multiple pre-trained teacher models and sequence data, a tiny subset of which is labeled, the goal is to train a multi-class student model that covers the union of all classes predicted by the teachers. The teachers may have different architectures. SKA extends beyond classic KA, as shown in Figure 2, both by exploring sequence classification and by broadening the KA setting to the more realistic case when a few annotations are available, thereby handling overconfident teachers.

Challenges. The open SKA problem is challenging for the following reasons:

• *Very limited supervision*: Modern multi-class sequence classifiers notoriously require a large amount of labeled training data (Fawaz et al. 2018; Malhotra et al. 2017). With only a tiny amount of labeled data, the risk of overfitting, which inhibits generalization, is quite high.

• *Disparate teachers*: Teachers are typically trained on different, and often private, sets of training data. Thus, their internal representations are customized for their own respective tasks while their behavior and output for classes unknown to them is unpredictable and has no relation to outputs of other teachers. To amalgamate knowledge from such *disparate* teachers, we must find the middle ground between these heterogeneous sources of knowledge.

• *Overconfident teachers*: Individual teachers are experts on different sets of classes and so the *scales* of their soft predicted values may differ drastically. Further, given an instance from a class that a teacher was not have trained to predict, a teacher may be *overconfident* in its prediction, giving misleading information to a student. This is particularly prevalent when the sheer *number* of classes differs between teachers: having more classes tends to generate on average higher soft predicted scores. A good solution must filter out untrustworthy recommendations by teachers when supervising the student's training.

Proposed Method. To overcome these challenges, we propose the **T**eacher **C**oordinator (TC) training paradigm. TC consists of two novel components: A *Teacher Trust Learner* (TTL), which learns to estimate how trustworthy each teacher is for a given data instance, and a *Knowledge Amalgamator*, which combines the sets of class probabilities predicted by disparate teachers into one final probability distribution over the set of target classes. The TTL is modeled as a Recurrent Neural Network (RNN), trained using very limited labeled data to predict a probability distribution over the *teachers*, estimating for each teacher the likelihood that it should be trusted to predict the given class label. The Knowledge Amalgamator uses the TTL's prediction to normalize and combine the predicted probabilities from each teacher. This then serves as a surrogate class label used to train a student RNN model to classify the input sequences. Altogether, TC is the first solution to the open problem of SKA, which estimates for the first time the probability over all target classes by rescaling outputs of each teacher based on its trustworthiness.

Contributions. Our main contributions are as follows:

• We define the novel semi-supervised knowledge amalgamation (SKA) problem for sequence classification, which is to train a multi-class student model to cover all classes predicted by a set of disparate pre-trained sequence models.

• We propose the Teacher Coordinator (TC), the first solution to the SKA problem. Our approach learns to estimate the probability distribution over all target classes by amalgamating knowledge of pre-trained teachers. TC is the first to estimate the trustworthiness of each teacher for KA.

• We find that TC significantly outperforms eight state-of-the-art alternatives on four real datasets by an average of 15% in accuracy. Additionally, TC is effective even when as little as 2% of the training data are annotated.

Related Work

The open problem of SKA for sequence classification is related to knowledge extraction from pre-trained models.

Ensemble Learning. Ensemble learning methods combine the predicted *outputs* from multiple models. Classical methods use averaging or majority voting (Kittler et al. 1998; Dietterich 2000); recent methods relying on deep learning include Drop Connection (Wan et al. 2013), Stochastic Depth (Huang et al. 2016), and Swapout (Singh, Hoiem, and Forsyth 2016). These methods assume every teacher is responsible for exactly the same set of classes, which is a special case of our problem. Moreover, this approach requires all methods to be run on each instance as part of inference during classification, which could cause scalability issues.

Knowledge Distillation (KD). As originally proposed, KD (Hinton, Vinyals, and Dean 2015) overcomes this scalability problem by compressing a large teacher model into a smaller student model while aiming to retain the accuracy of the teacher. This is achieved by training the student to imitate the teacher's soft predictions (logits). More recent works extend this technique by imitating intermediate layers (Adriana et al. 2015; Wang et al. 2018) or distilling knowledge from multiple teachers (You et al. 2017; Fukuda et al. 2017; Chen, Su, and Zhang 2019). However, these works continue to assume the teacher and student have identical class sets.

Knowledge Amalgamation (KA). KA (Shen et al. 2019a) advances beyond KD by combining teachers that specialize in different class sets. This is a more general form of the KD problem. Most KA works strongly assume that all teachers and students have identical network architectures (Shen et al. 2019a; Ye et al. 2019; Shen et al. 2019b). Most-related to our work, Luo et al. (2019) and Vongkulbhisal, Vinayavekhin, and Visentini-Scarzanella (2019) relax this assumption and allow for multiple teachers with heterogeneous architectures and different specialties. Luo et al. (2019) train a student to imitate the concatenated teachers' soft targets while simultaneously imitating their final representation layers, trained in a new shared space. Vongkulbhisal, Vinayavekhin, and Visentini-Scarzanella (2019) train the student's predictions over all classes by optimizing multiple cross entropy losses that match each teacher's specialized class sets. As shown later in our experiments section, all of these KA methods suffer when teachers are *overconfident* in terms of the scales of their predicted probabilities. This can lead to an inflated impact of an erroneous teacher while training a student model. None of these methods incorporate any labeled data during training, which we will show can be a quite effective approach to mitigating such overconfidence. Moreover, these works have been developed exclusively for images and do not tackle the KA problem for sequences.

Problem Formulation

We address the problem of semi-supervised knowledge amalgamation (SKA) for sequence classification. In this problem, we denote X as a sequence, or instance, of length m and x_t is its value at timestep t. We are given a training dataset $\mathcal{D}$ consisting of a tiny subset of labeled data $\mathcal{D}_l$ and a proportionately large subset of unlabeled data $\mathcal{D}_u$. Each instance in $\mathcal{D}_l$ has an associated $y_j \in \mathcal{Y}$ where $\mathcal{Y} = \{y_j\}_{j=1}^{c}$. Thus, $\mathcal{D}_l = \{(X^l, y_j^l)\}_{l=1}^{n_l}$ while $\mathcal{D}_u = \{X^u\}_{u=1}^{n_u}$, and $\mathcal{D} = \mathcal{D}_l \cup \mathcal{D}_u$. We denote all n training sequence instances as $\mathcal{X} = \{X^l\}_{l=1}^{n_l} \cup \{X^u\}_{u=1}^{n_u}$ and $n = n_l + n_u$:

$$\mathcal{X} = \left.\underbrace{\begin{pmatrix} x_1^1 & x_2^1 & \cdots & x_m^1 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^n & x_2^n & \cdots & x_m^n \end{pmatrix}}_{\text{Sequence of length } m}\right\} n \text{ instances}$$

We also have access to a set of p powerful pre-trained sequence classifiers (*teachers*), $\mathcal{T} = \{T_k\}_{k=1}^{p}$, each of which specializes in classifying a set of classes $\mathcal{Y}_k$. We note that $\mathcal{Y} = \bigcup_{k=1}^{p} \mathcal{Y}_k$. The goal is to train a *student* model, which can accurately predict the probability that instance X is associated with any class y_j out of the c classes in $\mathcal{Y}$. We describe our method in terms of one instance to improve readability.

Proposed Method

We propose the **T**eacher **C**oordinator (TC), which tackles SKA by utilizing the relationship between two conditional probabilities: the probability a teacher T_k is an expert on an instance's true label, or $P(y_j \in \mathcal{Y}_k | X)$, and the probability an instance's label is y_j according to teacher T_k, or $P(y_j | y_j \in \mathcal{Y}_k, X)$. As shown in Figure 3, TC works in two steps. First, we use a tiny amount of labeled data to train a *Teacher Trust Learner* (TTL), which estimates the likelihood a given teacher is an expert for an instance. This information then informs the *Knowledge Amalgamator*, which rescales the probability outputs from each teacher, each of which predict over different class sets $\mathcal{Y}_k$, to be used as supervision for training the student model. We model the TTL and the student as RNNs with LSTM memory cells.

Recurrent Neural Network (RNN) Background

RNNs are the state-of-the-art solution to several sequence classification problems (Hartvigsen et al. 2019, 2020). RNNs model sequences by updating representation vectors as new timesteps are observed. For sequence classification, the final representation is used to predict the class label of a sequence. Traditional RNNs suffer from vanishing gradients when learning to represent long-term dependencies. Thus, most RNN-based models use gating mechanisms such as Long Short-Term Memory cells (LSTM) (Hochreiter and Schmidhuber 1997). The LSTM works by computing a hidden state $h_t \in \mathbb{R}^d$ at every step of a sequence, where d is the dimension of the hidden state. At timestep t, given the current input x_t and the previous hidden state h_{t-1}, the new hidden state for the current timestep, h_t, is computed as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{1}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2}$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{3}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{4}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \odot \tanh(c_t) \tag{6}$$

where σ is the sigmoid function, tanh is the hyperbolic tangent function, $\cdot$ is the dot product, and $\odot$ is the hadamard product. W's and b's are learnable matrices and vectors of parameters, respectively. For simplicity, we denote the collection of all such parameters as θ. After an entire sequence X has been observed and this process has been repeated m times, the final hidden state is used as input to a classifier. We denote this process as one function that outputs the final hidden state: $h = \text{LSTM}_\theta(X)$.

The Proposed Teacher Coordinator (TC)

The first step of TC is to train the **Teacher Trust Learner (TTL)**. The TTL estimates the probability a teacher T_k is an expert on a given instance's true label. The TTL is modeled as an LSTM network trained on the small set of labeled data $\mathcal{D}_l$. Given a sequence X, the task is to classify which teachers are experts on X's true label y_j. Multiple teachers may be experts simultaneously because their respective sets of predicted classes may overlap. Thus, the TTL's task is naturally multi-label: given an instance, predict one probability per teacher. This is achieved using Equations 7 and 8:

$$h_a = \text{LSTM}_{\theta_a}(X) \tag{7}$$

$$e = \sigma(W_a \cdot h_a + b_a) \tag{8}$$

where θ_a, W_a and b_a are learnable parameters. σ is the sigmoid function, which maps vector elements into the range [0, 1], thereby predicting one probability per teacher. Thus, the closer elements of e are to 1, the more likely the given instance is associated to the corresponding teachers. This vector is normalized through a softmax function to generate our target, a true distribution over all teachers:

$$P(y_j \in \mathcal{Y}_k | X) = \text{softmax}(e). \tag{9}$$

As shown in Figure 3, the TTL is trained separately in Step 1, and used statically to train the student model during Step 2. The training objective of the TTL is to iteratively update θ_a, W_a, and b_a, grouped into θ_{TTL}, by minimizing binary cross entropy:

$$J(\theta_{TTL}) = -\sum_{k=1}^{p} \left(\mathbb{1}_{T_k} \log(e_k) + (1 - \mathbb{1}_{T_k}) \log(e_k) \right) \tag{10}$$

where $\mathbb{1}_{T_k}$ is 1 if $y_j \in \mathcal{Y}_k$ and 0 otherwise.

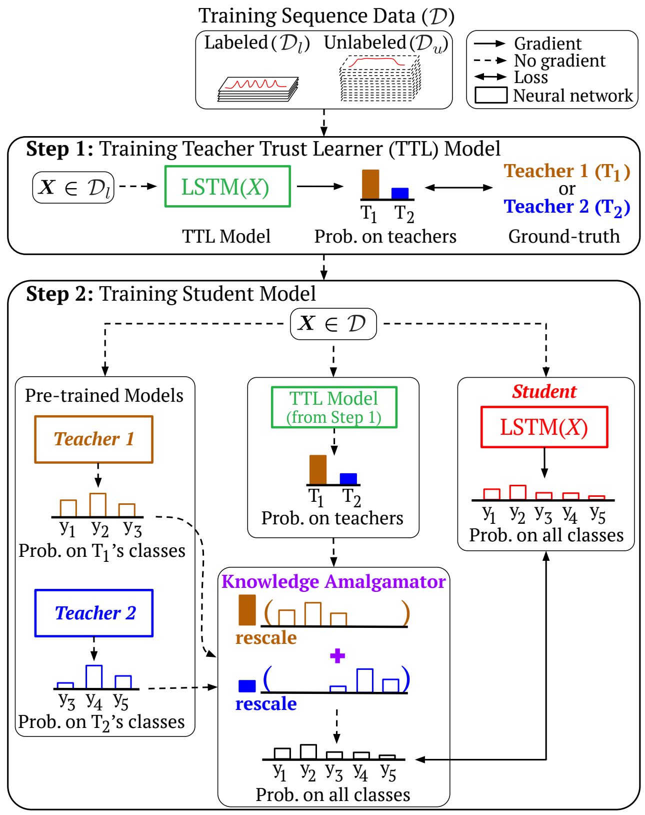


Figure 3: The architecture of our proposed Teacher Communicator. Training proceeds through both steps sequentially.

The second step of TC is the **Knowledge Amalgamator**, which produces a probability an instance's label is y_j. As shown below, this distribution over all classes in $\mathcal{Y}$ can be estimated using the relationship between two conditional probabilities: (i) the probability each teacher is an expert on X's true label as estimated by the TTL, and (ii) the probability X's label is y_j, given a teacher model.

Estimating the Class Probability. Given an input instance X, the goal is to estimate $P(y_j|X)$, where $y_j \in \mathcal{Y}$.

Given that the TTL has estimated $P(y_j \in \mathcal{Y}_k|X)$, let us consider the set of teacher models $\{T_k\}_{k=1}^p$, each of which specializes on a corresponding set of classes $\mathcal{Y}_k$. Each teacher thus generates a separate prediction

$$P(y_j | y_j \in \mathcal{Y}_k, X). \tag{11}$$

The target $P(y_j|X)$ can be estimated directly:

$$\begin{aligned} P(y_j|y_j \in \mathcal{Y}_k, X) * P(y_j \in \mathcal{Y}_k|X) \\ &= \frac{P(y_j, y_j \in \mathcal{Y}_k, X)}{P(y_j \in \mathcal{Y}_k, X)} * \frac{P(y_j \in \mathcal{Y}_k, X)}{P(X)} \\ &= \frac{P(y_j, y_j \in \mathcal{Y}_k, X)}{P(X)} \\ &= P(y_j, y_j \in \mathcal{Y}_k|X) \\ &= P(y_j|X) \end{aligned} \tag{12}$$

Finally, the **Student Network** estimates $Q(y_j|X)$, the probability of X's label being $y_j \in \mathcal{Y}$, by imitating $P(y_j|X)$, which has been produced by the Knowledge Amalgamator.

We model the student as an LSTM network:

$$h_S = \text{LSTM}_{\theta_S}(X) \tag{13}$$

$$Q(y_j|X) = \frac{\exp(W_S \cdot h_S + b_S)}{\sum_j \exp(W_S \cdot h_S + b_S)} \tag{14}$$

where $y_j \in \mathcal{Y}$, and θ_S, W_S and b_S are trainable parameters.

The student network is trained on all training sequences in $\mathcal{X}$ using $P(y_j|X)$ as a surrogate target. The goal is to iteratively update the collection of parameters: θ_S, W_S and b_S (collectively referred to as θ_{SN}) by minimizing the cross entropy between its own predicted probability over all classes and the amalgamated probability distribution $P(y_j|X)$:

$$J(\theta_{SN}) = -\sum_{j=1}^{c} P(y_j|X) \log(Q(y_j|X)). \tag{15}$$

Experiments

We evaluate our approach, TC, using three challenging settings of the SKA problem on four datasets. We compare the performance of TC to eight state-of-the-art alternatives.

Datasets

We focus our experiments on four well-known time series classification datasets below.

• *SyntheticControl* (SYN) (Alcock, Manolopoulos et al. 1999). These data contain control chart patterns of a machine parameter recorded over time consisting of 600 instances, each with 60 timesteps. The task is to classify six different patterns: normal, cyclic, increasing trend, decreasing trend, upward shift, and downward shift.

• *MelbournePedestrian* (PED) (Carter et al. 2020). This dataset consists of pedestrian counts at 10 locations in Melbourne, Australia throughout 2017 recorded over 3,633 instances with 24 length for each. The target is to detect from which location a given pedestrian's activity pattern comes, thereby supporting urban planning problems.

• *Human Activity Recognition Using Smartphones* (HAR) (Anguita et al. 2013). The task is to detect which action a person is performing, given their smartphone sensor data, out of the 6 actions: running, walking, sitting, lying down, going up stairs, and going down stairs. Each sample out of 10,299 instances has 561 timesteps.

• *ElectricDevices* (ELEC) (Lines and Bagnall 2014). Seven devices were monitored from 251 households in the UK in two-minute intervals over a month. The task is to detect patterns in device usage and ultimately anomalous behavior. This dataset contains 16,637 instances. Each of which has 96 length.

Note that we assume that each dataset ($\mathcal{D}$) used to train a student model consists of a small subset of labeled data ($\mathcal{D}_l$) and a proportionately large subset of unlabeled data ($\mathcal{D}_u$).

Compared Methods

As the SKA problem is newly proposed in this work, state-of-the-art approaches to KA are not designed directly for this setting. We divide eight alternative methods into three categories as follow.

Baselines: These methods assume access to no teachers.

• *Original Teachers*: Teachers are used independently. No predicted probabilities are assigned to classes outside of a teacher's specialties.

• *SupLSTM*: An LSTM is trained from scratch on the labeled data. This is standard supervised learning, though with few labeled instances.

• *SelfTrain* (Rosenberg, Hebert, and Schneiderman 2005): A classic semi-supervised approach that iteratively learns pseudo-labels for unlabeled data, adding high-confidence predictions to the training set.

Unsupervised KA methods: These methods use all instances in $\mathcal{D}$ and all teachers in $\mathcal{T}$ to train a student model.

• *KD* (Hinton, Vinyals, and Dean 2015): A student's soft targets (logits) are trained to imitate the average of all teachers' logits using Knowledge Distillation. For disjoint classes, the logits are concatenated.

• *CFL* (Luo et al. 2019): The student is trained to imitate each teachers' logits and final hidden features. Final features of the student and teachers are mapped to a common space in which similarity is encouraged.

• *UHC* (Vongkulbhisal, Vinayavekhin, and Visentini-Scarzanella 2019): The student's output is split into subsets corresponding to each teacher's class set. Then, each subset is trained to estimate its corresponding teacher's output.

Supervised KA methods: These methods use only the labeled instances in the small set $\mathcal{D}_l$ to train a student model.

• *SupKD*: A student is trained only on the small set of labeled data to imitate the teachers' logits and also to predict correctly such labels.

• *SupUHC*: UHC is extended by adding supervision from the small amount of given labeled data. This is achieved by adding a supervised objective to the original model.

Implementation Details

Each dataset in our study has pre-specified training and testing subsets, defined in the literature. In our experiments, the training file is used only for training teacher models, while the original testing file is split to train and evaluate the student network (70% of which for training, 10% for validation, and 20% for testing). All hyperparameters are tuned using the validation set. Once hyperparameters are set, we train the student model using both the training and validation sets and record accuracy on the testing set. Each experiment is replicated on three random seeds after which we report the mean and standard deviation of accuracy over all replications. All teacher/student models are LSTMs. Our model is implemented using PyTorch and optimized using Adam (Kingma and Ba 2014). All code and datasets are available at https://github.com/jida-thada/SKA.

Experimental Results

We investigate three key properties of the SKA task: the effects of using small labeled data, learning from disparate teachers, and the impact of overconfident teachers.

	SYN				PED				HAR				ELEC			
	Ratio of labeled data				Ratio of labeled data				Ratio of labeled data				Ratio of labeled data			
Methods	2%	4%	6%	8%	2%	4%	6%	8%	2%	4%	6%	8%	2%	4%	6%	8%
Teacher 1	.66±.00	.66±.00	.66±.00	.66±.00	.55±.00	.55±.00	.55±.00	.55±.00	.51±.00	.51±.00	.51±.00	.51±.00	.47±.00	.47±.00	.47±.00	.47±.00
Teacher 2	.64±.00	.64±.00	.64±.00	.64±.00	.49±.00	.49±.00	.49±.00	.49±.00	.58±.00	.58±.00	.58±.00	.58±.00	.41±.00	.41±.00	.41±.00	.41±.00
SupLSTM	.51±.11	.58±.10	.66±.12	.71±.06	.27±.05	.49±.03	.52±.06	.57±.06	.42±.07	.50±.12	.61±.03	.68±.07	.52±.06	.62±.05	.69±.04	.69±.01
SelfTrain	.58±.06	.65±.04	.75±.04	.79±.05	.48±.03	.65±.02	.65±.02	.68±.06	.46±.06	.54±.06	.64±.05	.65±.10	.56±.03	.62±.01	.70±.03	.69±.02
KD	.87±.01	.87±.01	.87±.01	.87±.01	.61±.03	.61±.03	.61±.03	.61±.03	.60±.01	.60±.01	.60±.01	.60±.01	.65±.01	.65±.01	.65±.01	.65±.01
CFL	.63±.01	.63±.01	.63±.01	.63±.01	.48±.02	.48±.02	.48±.02	.48±.02	.30±.00	.30±.00	.30±.00	.30±.00	.58±.02	.58±.02	.58±.02	.58±.02
UHC	.86±.01	.86±.01	.86±.01	.86±.01	.60±.03	.60±.03	.60±.03	.60±.03	.66±.10	.66±.10	.66±.10	.66±.10	.62±.01	.62±.01	.62±.01	.62±.01
SupKD	.55±.04	.61±.14	.67±.08	.69±.01	.51±.07	.61±.02	.64±.05	.67±.01	.48±.09	.58±.06	.64±.04	.64±.03	.45±.04	.64±.03	.69±.04	.68±.01
SupUHC	.47±.07	.58±.12	.66±.13	.70±.04	.46±.04	.58±.02	.65±.03	.64±.01	.41±.06	.46±.05	.49±.05	.65±.12	.52±.05	.61±.04	.62±.05	.63±.02
TC (Ours)	**.90**±.02	**.91**±.04	**.92**±.02	**.93**±.01	**.69**±.01	**.70**±.02	**.75**±.04	**.76**±.03	**.75**±.01	**.77**±.02	**.78**±.01	**.78**±.02	**.66**±.01	**.68**±.03	**.71**±.02	**.71**±.02

Table 1: Compared performance (Accuracy±SD) on varied tiny rates of available labeled data in the training data.

	Overlapping class sets			Exclusive class sets		
Methods	SYN	PED	HAR	SYN	PED	HAR
Teacher 1	.66±.00	.55±.00	.51±.00	.50±.00	.43±.00	.38±.00
Teacher 2	.64±.00	.49±.00	.58±.00	.47±.00	.44±.00	.45±.00
SupLSTM	.51±.11	.27±.05	.42±.07	.51±.11	.27±.05	.42±.07
SelfTrain	.58±.06	.48±.03	.46±.06	.58±.06	.48±.03	.46±.06
KD	.87±.01	.61±.03	.60±.01	.54±.04	.44±.03	.55±.02
CFL	.63±.01	.48±.02	.30±.00	.50±.03	.30±.05	.50±.08
UHC	.86±.01	.60±.03	.66±.10	.61±.02	.43±.02	.55±.01
SupKD	.55±.04	.51±.07	.48±.09	.48±.12	.48±.02	.50±.02
SupUHC	.47±.07	.46±.04	.41±.06	.53±.02	.43±.09	.22±.11
TC (Ours)	**.90**±.02	**.69**±.01	**.75**±.01	**.77**±.07	**.64**±.04	**.84**±.04

Table 2: Accuracy±SD when combining disparate teachers. Left: teachers are partially related, sharing 2 classes. Right: teachers are disjoint, sharing no classes.

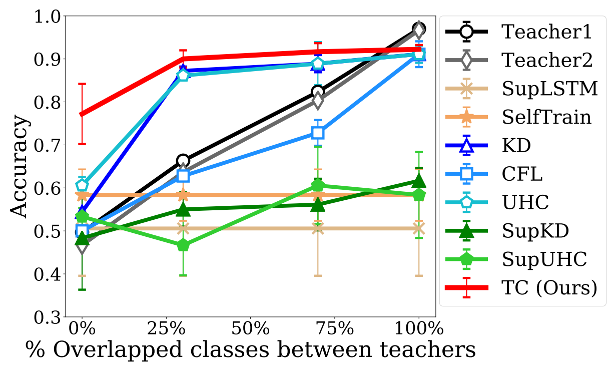


Figure 4: The shared classes in teachers are varied from 0% (fully disparate) to 100% (homogeneous).

Incorporation of limited labeled data. Effective use of very limited available annotations is essential to solving the SKA problem. We thus compare each method's performance using label proportions ranging from 2%-8% of the training data. In this setting, we use two teacher models, each with an equal number of classes, that have partially overlapped classes (one or two classes in common depending on the number of classes in each dataset).

The results of these experiments are in Table 1. By incorporating small labeled data effectively, even if such data are available only 2% of the training set, we observe that TC improves the classification accuracy significantly by an average of 6% over the second-best methods across the board. We notice that unsupervised KA methods (KD, CFL, and UHC) outperform baselines (SupLSTM and SelfTrain) and supervised KA methods (SupKD and SupUHC) when we have too few labels (2% and 4% annotated), as expected. This is because the baseline and supervised KA methods depend on sufficient labeling, though the impact is highly task-dependent. Once enough labels become available (6% and 8%), such baseline and supervised KA methods do begin to outperform unsupervised KA methods. This clearly shows the need for carefully incorporating labels into the KA problem, no matter how few are available. As shown in bold, our TC significantly outperforms all other compared methods on all label proportions for all datasets. As expected, in line with the observed trend of the supervised methods, as the proportion of labeled data increases, so does the performance of TC. This is because TC combines the benefits when labels are available with the strengths of the unsupervised KA methods when there are only few labels.

Amalgamating disparate teachers. Since teacher models are typically trained independently on their own tasks, their outputs are often unrelated to one another and are thus challenging to combine. To understand how this impacts the performance of all compared methods, we vary the *proportion of overlapping classes* among the teachers. Thereby we assume that the smaller the overlap between predicted classes, the more disparate are the teachers because they share less information. We conduct this experiment using two scenarios: (1) *Overlapping class sets* arise when the class sets of the teachers are at least partially shared. Here, both teachers share exactly two classes. (2) *Exclusive class sets* occur in the extreme case that the teachers' class sets are entirely disjoint. The latter is challenging because these teachers have no common class to anchor the scale of the predicted probabilities while supervising the student.

Table 2 shows all results on these settings. All methods have access to annotations for only 2% of the training data. We observe that all methods suffer when the teachers' class sets are mutually exclusive as the student models fail to fuse different knowledge from such disparate teachers. However, TC successfully overcomes this issue by rescaling each probability output of their disparate class sets to form their joint probability over all classes, resulting in 14% higher accuracy than the other methods on average.

Moreover, we extend this study to a wide variety of class-overlap ranges (0% to 100%) on the SYN dataset, the re-

Methods	SYN	PED	HAR	ELEC
Teacher 1	.33±.00	.30±.00	.28±.00	.32±.00
Teacher 2	.64±.00	.64±.00	.58±.00	.44±.00
SupLSTM	.51±.11	.27±.05	.42±.07	.52±.06
SelfTrain	.58±.06	.48±.03	.46±.06	.56±.03
KD	.66±.01	.66±.02	.57±.02	.37±.00
CFL	.64±.01	.63±.01	.53±.04	.36±.01
UHC	.66±.03	.64±.02	.49±.10	.36±.03
SupKD	.49±.03	.55±.04	.31±.13	.34±.06
SupUHC	.51±.05	.46±.06	.35±.07	.39±.05
TC (Ours)	**.85**±.07	**.70**±.08	**.75**±.01	**.64**±.01

Table 3: Learning from overconfident teachers. Teacher 2 has roughly twice as many classes as Teacher 1 for all tasks.

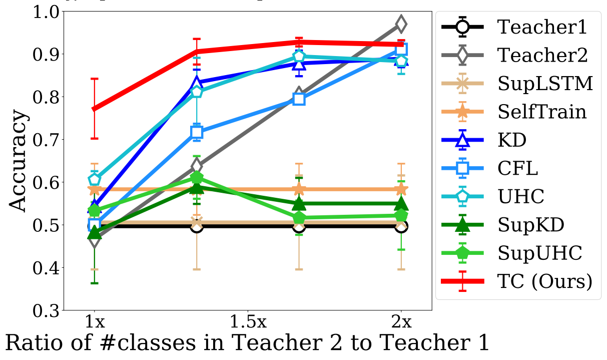


Figure 5: Increasing the proportion of all classes on which Teacher 2 is an expert, observed on the SYN dataset.

sults of which are shown in Figure 4. As expected, all models perform their worst when teachers have 0% overlap and improve gradually as teachers share more classes. Once the teachers' classes are 100% overlapping, each teacher is an expert on all classes and so no KA methods can surpass their performance. TC surpasses the other methods in all other cases, showing that rescaling the predicted scores from each teacher solves the *disparate teacher* challenge. Our publicly-available repository contains similar trends observed on the other datasets.

Overcoming overconfident teachers. Next, we study the effects of *overconfident* teachers, which make confident predictions, even when instances are actually from classes on which they are not experts. The sheer number of classes for which a teacher is an expert is a key contributor in the scale of its predictions. A teacher with more classes usually generates a large range of its logits to clearly distinguish the predictions over many classes. Such large logits can produce overconfident predictions for unseen instances for which the teacher is not an expert. This can therefore dominate the predictions of other teachers. We investigate this condition across datasets when the number of specialized classes in Teacher 2 is roughly twice that of Teacher 1 (the actual number ranges from 2 to 2.5, depending on the number of classes in each dataset). We once again assume only 2% labeling.

As shown in Table 3, we first notice that the accuracy of Teacher 2 is roughly twice as high as those of Teacher 1. Further, due to its overconfidence, the compared KA methods cannot improve beyond the performance of Teacher 2. In contrast, our TC is still able to combine the relevant knowledge from each teacher, boosting accuracy 13% higher.

Methods	3 teachers	4 teachers	5 teachers
Teacher 1	.35±.00	.30±.00	.19±.00
Teacher 2	.37±.00	.28±.00	.20±.00
Teacher 3	.36±.00	.26±.00	.18±.00
Teacher 4	NA	.26±.00	.20±.00
Teacher 5	NA	NA	.20±.00
SupLSTM	.27±.05	.27±.05	.27±.05
SelfTrain	.48±.03	.48±.03	.48±.03
KD	.30±.01	.30±.01	.21±.02
CFL	.27±.05	.21±.01	.21±.01
UHC	.32±.02	.42±.09	.22±.04
SupKD	.35±.06	.39±.04	.44±.01
SupUHC	.37±.10	.34±.04	.33±.10
TC (Ours)	**.71**±.04	**.67**±.05	**.62**±.04

Table 4: Learning from many teachers on the PED dataset.

To study this effect more deeply, in Figure 5 we illustrate the effect of different levels of overconfidence using the SYN dataset. The number of classes for Teacher 1 is fixed in all experiments and varied for Teacher 2, spanning from the same number as Teacher 1 (three classes per teacher) to twice as many classes (six classes for Teacher 2, three classes still for Teacher 1). Our results indicate that as the number of classes for Teacher 2 compared to Teacher 1 increases, all KA methods are controlled more by the predictions from Teacher 2 and thus the improvement beyond Teacher 2 decreases. However, TC achieves the largest improvements in all cases compared to other methods as it succeeds in informing accurate predictions from the correct and trustworthy teachers to the student model. At 2x, Teacher 2 specializes in all classes in the student's task, leading to a plateau in performance. Thus the students cannot improve beyond Teacher 2's accuracy. Similar results on the other datasets are shown in our publicly-available repository.

Amalgamating many teachers. Finally, we find that TC dramatically outperforms state-of-the-art KA methods when we use many teachers. To demonstrate this, we compare methods trained to amalagamate three, four, and five teacher models using the PED dataset. The PED dataset has 10 classes, allowing us to train more teachers specialized on different class sets. Our results, shown in Table 4, indicate that TC dramatically outperforms the other methods with an average improvement of 26%, even with as few as three teachers and especially when there are five teachers.

Conclusion

In this work, we introduce the new problem of semi-supervised knowledge amalgamation for sequence classification. We design the first solution, the Teacher Coordinator (TC), which tackles knowledge amalgamation learning from multiple pre-trained teachers on sequence data. TC overcomes the issue of not having enough annotations for learning multi-class tasks by training a student model using the knowledge of teachers with different specialties while requiring only a tiny set of annotations. Our approach effectively rescales the predicted output of all disparate teachers by estimating how trustworthy each teacher is for a given instance. Our experiments show that TC significantly outperforms other state-of-the-art methods with an average improvement of 15% in accuracy across a wide variety of tasks.

Acknowledgements

This work is supported by NSF grants IIS-2007932 and IIS-1718310 and the National Institute of Food and Agriculture under AFRI grant 1023720. We also thank the GAANN Fellowships in Computer and Information Sciences for AI from the U.S. Dept. of Education, P200A180088. Jidapa especially acknowledges the Royal Thai Government for her support.

References

Adriana, R.; Nicolas, B.; Ebrahimi, K. S.; Antoine, C.; Carlo, G.; and Yoshua, B. 2015. Fitnets: Hints for thin deep nets. In *Proceedings of ICLR*.

Alcock, R. J.; Manolopoulos, Y.; et al. 1999. Time-series similarity queries employing a feature-based approach. In *Hellenic conference on informatics*, 27–29.

Anguita, D.; Ghio, A.; Oneto, L.; Parra, X.; and Reyes-Ortiz, J. L. 2013. A public domain dataset for human activity recognition using smartphones. In *Proceedings of ESANN*, volume 3, 3.

Carter, E.; Adam, P.; Tsakis, D.; Shaw, S.; Watson, R.; and Ryan, P. 2020. Enhancing pedestrian mobility in Smart Cities using Big Data. *Journal of Management Analytics* 1–16.

Chen, X.; Su, J.; and Zhang, J. 2019. A Two-Teacher Framework for Knowledge Distillation. In *Proceedings of ISNN*, 58–66.

Dietterich, T. G. 2000. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, 1–15.

Fawaz, H. I.; Forestier, G.; Weber, J.; Idoumghar, L.; and Muller, P.-A. 2018. Transfer learning for time series classification. In *Proceedings of Big Data)*, 1367–1376.

Fukuda, T.; Suzuki, M.; Kurata, G.; Thomas, S.; Cui, J.; and Ramabhadran, B. 2017. Efficient Knowledge Distillation from an Ensemble of Teachers. In *Proceedings of Interspeech*, 3697–3701.

Graves, A.; Jaitly, N.; and Mohamed, A.-r. 2013. Hybrid speech recognition with deep bidirectional LSTM. In *Proceedings of IEEE workshop on automatic speech recognition and understanding*, 273–278.

Hartvigsen, T.; Sen, C.; Kong, X.; and Rundensteiner, E. 2019. Adaptive-Halting Policy Network for Early Classification. In *Proceedings of ACM SIGKDD*, 101–110.

Hartvigsen, T.; Sen, C.; Kong, X.; and Rundensteiner, E. 2020. Recurrent Halting Chain for Early Multi-label Classification. In *Proceedings of ACM SIGKDD*, 1382–1392.

Harutyunyan, H.; Khachatrian, H.; Kale, D. C.; Ver Steeg, G.; and Galstyan, A. 2019. Multitask learning and benchmarking with clinical time series data. *Scientific data* 6(1): 1–18.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* .

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8): 1735–1780.

Huang, G.; Sun, Y.; Liu, Z.; Sedra, D.; and Weinberger, K. Q. 2016. Deep networks with stochastic depth. In *Proceedings of ECCV*, 646–661.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Kittler, J.; Hatef, M.; Duin, R. P.; and Matas, J. 1998. On combining classifiers. *IEEE TPAMI* 20(3): 226–239.

Lines, J.; and Bagnall, A. 2014. Ensembles of elastic distance measures for time series classification. In *Proceedings of SDM*, 524–532.

Luo, S.; Wang, X.; Fang, G.; Hu, Y.; Tao, D.; and Song, M. 2019. Knowledge amalgamation from heterogeneous networks by common feature learning. In *Proceedings of IJCAI*, 3087–3093.

Malhotra, P.; TV, V.; Vig, L.; Agarwal, P.; and Shroff, G. 2017. TimeNet: Pre-trained deep recurrent neural network for time series classification. In *Proceedings of ESANN*.

Masum, A. K. M.; Bahadur, E. H.; and Ruhi, F. A. 2020. Scrutiny of Mental Depression through Smartphone Sensors Using Machine Learning Approaches. *International Journal of Innovative Computing* 10(1).

Razavian, N.; Marcus, J.; and Sontag, D. 2016. Multi-task prediction of disease onsets from longitudinal laboratory tests. In *Proceedings of MLHC*, 73–100.

Rosenberg, C.; Hebert, M.; and Schneiderman, H. 2005. Semi-Supervised Self-Training of Object Detection Models. In *Proceedings of IEEE WACV/MOTION*, 29–36.

Shen, C.; Wang, X.; Song, J.; Sun, L.; and Song, M. 2019a. Amalgamating knowledge towards comprehensive classification. In *Proceedings of AAAI*, 3068–3075.

Shen, C.; Xue, M.; Wang, X.; Song, J.; Sun, L.; and Song, M. 2019b. Customizing student networks from heterogeneous teachers via adaptive knowledge amalgamation. In *Proceedings of ICCV*, 3504–3513.

Singh, S.; Hoiem, D.; and Forsyth, D. 2016. Swapout: Learning an ensemble of deep architectures. In *Proceedings of NeurIPS*, 28–36.

Vongkulbhisal, J.; Vinayavekhin, P.; and Visentini-Scarzanella, M. 2019. Unifying heterogeneous classifiers with distillation. In *Proceedings of CVPR*, 3175–3184.

Wan, L.; Zeiler, M.; Zhang, S.; Le Cun, Y.; and Fergus, R. 2013. Regularization of neural networks using dropconnect. In *Proceedings of ICML*, 1058–1066.

Wang, H.; Zhao, H.; Li, X.; and Tan, X. 2018. Progressive Blockwise Knowledge Distillation for Neural Network Acceleration. In *Proceedings of IJCAI*, 2769–2775.

Yang, X.; Chen, Y.; Yu, H.; Zhang, Y.; Lu, W.; and Sun, R. 2020. Instance-Wise Dynamic Sensor Selection for Human Activity Recognition. In *Proceedings of AAAI*, 1104–1111.

Ye, J.; Wang, X.; Ji, Y.; Ou, K.; and Song, M. 2019. Amalgamating filtered knowledge: learning task-customized student from multi-task teachers. In *Proceedings of IJCAI*, 4128–4134.

You, S.; Xu, C.; Xu, C.; and Tao, D. 2017. Learning from multiple teacher networks. In *Proceedings of ACM SIGKDD*, 1285–1294.