The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Learning with Single-Teacher Multi-Student

Shan You,[†] Chang Xu,[‡] Chao Xu,[†] Dacheng Tao[‡]
[†]Key Lab. of Machine Perception (MOE), Cooperative Medianet Innovation Center,
School of EECS, Peking University, China
[‡]UBTECH Sydney AI Centre, SIT, FEIT, University of Sydney, Australia
youshan@pku.edu.cn, c.xu@sydney.edu.au
xuchao@cis.pku.edu.cn, dacheng.tao@sydney.edu.au

Abstract

In this paper we study a new learning problem defined as "Single-Teacher Multi-Student" (STMS) problem, which investigates how to learn a series of student (simple and specific) models from a single teacher (complex and universal) model. Taking the multiclass and binary classification for example, we focus on learning multiple binary classifiers from a single multiclass classifier, where each of binary classifier is responsible for a certain class. This actually derives from some realistic problems, such as identifying the suspect based on a comprehensive face recognition system. By treating the already-trained multiclass classifier as the teacher, and multiple binary classifiers as the students, we propose a gated support vector machine (gSVM) as a solution. A series of gSVMs are learned with the help of single teacher multiclass classifier. The teacher's help is two-fold; first, the teacher's score provides the gated values for students' decision; second, the teacher can guide the students to accommodate training examples with different difficulty degrees. Extensive experiments on real datasets validate its effectiveness.

Introduction

Multiclass classification (MC) (Aly 2005; Bishop 2006; Nie, Wang, and Huang 2017) considers that an instance can be affiliated to one of multiple (≥ 2) classes. Many real applications can be cast into MC problem. Taking the face recognition (He et al. 2017) for example, the task is required to distinguish multiple people based on their face photos, *e.g.* 126 individuals in AR Face Database (Martinez 1998). Other MC applications refer to image classification (Guo 2017), text categorization (Xu et al. 2017), information retrieval (Yang et al. 2017), multi-view classification (Xu, Tao, and Xu 2013) *et.al.*

To accommodate multiple classes, the multiclass classifiers usually adopt a score-based prediction mechanism. In detail, they can generate a score vector in terms of all classes as the output, then the class with the maximum score is regarded as the prediction. Some of the widely-used multiclass classifiers actually follow this mechanism, such as One-vs-All (Vapnik 2013), structural SVM (Tsochantaridis et al. 2004), rank SVM (Joachims 2002), *et.al.* As a result, the multiclass classifiers usually have larger model size than the binary classifiers, and more expensive inference cost as well.

Suppose we have a multiclass classifier (*e.g.* structural SVM) which has been already trained well. It allows us to identify a class from multiple candidates. In some real circumstances, however, the task in hand may only require a binary classification task, *i.e.* identifying whether a test instance belongs to one specific class instead of what class it belongs to. For example, in news categorization a multiclass classifier is implemented to recognize what category a news item is related to, such as tech news, entertainment news, economic news, *et.al.* However, when a user only requests for the tech news, the system only needs to classify whether the news items are tech-related or not. Another example refers to the suspect tracking in criminal detection. Usually, a face recognition system has already existed whose function is to identify a person's ID according to his/her face. This system may be capable of distinguishing millions of people. Nevertheless, for suspect tracking, we only concern whether the testing faces belong to the suspect or not.

Of course, we can still use the multiclass classifier to directly recognize whether the instance is a tech news item (or the suspect). If the instance is related to other news categories (or persons), we think it is no tech news (or suspect). However, this practice will always involve the inference of all the other classes; thus for a binary classification, the inference cost is too expensive. Or just using the training data to learn the binary classifiers is also an intuitive method. Nevertheless, this can not take advantage of the existing multiclass classifier which may provide some useful information. Therefore, it is of the need to learn a new binary classifier with much faster inference speed than the multiclass classifier, and higher classification accuracy than that of simple learning with training data.

In our paper, we suggest learning the binary classifiers by virtue of the existing multiclass classifier. In fact, this can be likened vividly to the basketball team training by a single teacher (or coach) (He, Eisner, and Daume 2012; Hinton, Vinyals, and Dean 2015; You et al. 2017b; Zhu, Liu, and Lopes 2017). An experienced teacher has extensive knowledge about how to be a good forward, center and guard. What the teacher wants is to train a team and each player has his/her own speciality; some specialize in the forward while others are more suitable for centers or guards. Our problem can also be regarded as learning from a single teacher (*i.e.* the multiclass classifier) to obtain multiple students (*i.e.* binary classifiers), each of which is responsible for a certain class. To the best of our knowledge, this is the first attempt to study this type of problem, which we define as "Single-Teacher Multi-Student" (STMS) problem.

Different from our prior work (You et al. 2017b) aiming at "Multi-Teacher Single-Student" problem where multiple teachers have the same functions, our work highlights multiple students and they possess different functions from each other (*i.e.* covering different binary classes). To solve the STMS problem, we propose a gated support vector machine (gSVM) to accomplish the learning of a series of binary classifiers with the help of a teacher multiclass classifier. Treating the teacher's score as auxiliary information, we formulate the students' learning as a skiping out of the gate game where the gate is set by the teacher. The gate (*i.e.* teacher's score) reflects how strongly the teacher has confidence about an instance's belonging to the target class. If a student can not skip out of the gate, then he/she has to stick to the teacher's decision, and also thinks the instance belongs to the target class. On the contrary, if the student does skip out of the gate, then he/she can reject the opinion of the teacher, and votes against belonging to the target class. During the training, we encourage the magnitude order between student's prediction value and teacher's gated value match with the ground-truth class response. By further studying the teacher's score matrix on the training examples, we introduce a difficulty degree on each example to measure how difficult the example is for the students. This prior difficulty information given by the teacher is to further guide and benefit the students during the learning. We embed this privileged information (Du, Xu, and Tao 2017; You et al. 2017a) by rescaling the slack variables, and formulate the whole model in an SVM fashion, which we name gated SVM. A majorization-minimization algorithm is adopted to optimize the gSVM, with an iteratively-reweighted least-squares method presented. Experimental results on extensive benchmark datasets validate the effectiveness of our gated SVM.

Problem Formulation

In this section, we detail our solution to the Single-Teacher Multi-Student problem. Focusing on classification problems, we investigate how to use a multiclass classifier (teacher) to develop a variety of binary classifiers (students) corresponding to different classes. For simplicity, we take the structural support vector machines (SSVMs) as our teacher model for example, but our solution can be extended to other score-based multiclass classification approaches.

Teacher Model: SSVM

Typical SVM is designed for binary classification problems with the output being 1 or -1. In contrast, multiclass classification problems involve multiple classes (*e.g.* K classes), the output space $\mathcal{Y}$ changes from $\{1, -1\}$ to $\{1, 2, ..., K\}$. To accomodate this output space or even more complicated one, structual support vector machine (SSVM) is developed (Tsochantaridis et al. 2004; Yu and Joachims 2009; Joachims, Finley, and Yu 2009). An SSVM is parameterized as a function of a problem-specific joint feature map $\psi(\mathbf{x}, y) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^M$, where $\mathcal{X}$ is the feature space and $\mathcal{Y}$ is the output space (*e.g.* $\mathcal{Y} = \{1, 2, ..., K\}$ for multi-class classification). Then the SSVM can be learned from the following model with training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$:

$$\begin{aligned} &\min_{\boldsymbol{w},\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ &\text{s.t. } \Delta(y, y_i) + \langle \boldsymbol{w}, \psi(\mathbf{x}_i, y)\rangle - \langle \boldsymbol{w}, \psi(\mathbf{x}_i, y_i)\rangle \le \xi_i \\ &\qquad \xi_i \ge 0,\ \forall y \in \mathcal{Y}, i = 1, ..., n, \end{aligned} \tag{1}$$

where $\langle\cdot,\cdot\rangle$ is the Euclidean inner product of two vectors or matrixes and $\Delta(\cdot,\cdot)$ is a distance measure for two outputs. In this way, for a new instance $\mathbf{x}$, its prediction is determined to be the class with highest score, *i.e.*

$$\hat{y} = \arg\max_{y\in\mathcal{Y}} \ \langle \boldsymbol{w}^*, \psi(\mathbf{x}, y)\rangle \tag{2}$$

Note that this inference needs to traverse *all* classes for the maximization operator over all socres.

Student Model: Gated SVM

Suppose we have a well-trained multiclass SSVM model $\mathcal{T}$ as the teacher, the aim is to learn a series of binary classifiers each of which is responsible for a certain class. Given the training dataset $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]$ of n examples and their ground-truth labels $\mathbf{y} \in \mathcal{Y}^n$, the prediction scores of all examples on all classes can form a score matrix $S \in \mathbb{R}^{n\times K}$ given by the teacher model $\mathcal{T}$, where (i, j)-th element S_{ij} refers to the i-th example's prediction score on the j-th class. For SSVM, the score is calculted as Eq.(2),

$$S_{ij} = \langle \boldsymbol{w}^*, \psi(\mathbf{x}_i, j)\rangle . \tag{3}$$

The score matrix reflects the teacher's acceptance confidence on each of classes the examples belong to. Without the loss of generality, we focus on the learning for the y-th class in $\mathcal{Y}$. Then the y-th column $\mathbf{s}^y$ of S indicates the teacher $\mathcal{T}$'s confidence on the class y. Considering that the teacher is well-trained and predict correctly in most times, then the scores of those with positive labels are usually the maximum values and vice versa. Then to identify the class y, an intuitive idea is to present the rejection confidence, namely, how confidently we think an instance does not belong to this class. Suppose the rejection confidence of class y on instance $\mathbf{x}$ is parameterized as $\varphi(\mathbf{x}; \boldsymbol{\theta})$ with parameter $\boldsymbol{\theta}$. The predicted result $\hat{z}^y$ on the class y follows

$$\hat{z}^y = \begin{cases} +1, & \text{if } \langle w^*, \psi(\mathbf{x}, y)\rangle \ge \varphi(\mathbf{x}; \boldsymbol{\theta}) \\ -1, & \text{otherwise} \end{cases} \tag{4}$$

where $+1$ indicates the instance $\mathbf{x}$ belongs to class y while -1 means the otherwise. Eq.(4) serves as the decision rule; what a student has to learn is the rejection confidence function $\varphi(\mathbf{x}; \boldsymbol{\theta})$. Once the $\varphi(\mathbf{x}; \boldsymbol{\theta})$ is learned, the inference (testing) will go as following: given an example, a student estimates the rejection confidence by calculating the function φ, then the student will compare it to the acceptance confidence given by the teacher. If the student has a larger value than that of the teacher, then he/she will reject the teacher and predict the label to be negative.

Table 1: The difficulty of examples reflected by the differences of teacher's scores. $\uparrow$ indicates that the larger the value is, the more difficult the example is. $\downarrow$ means the smaller the value is, the more difficult the example is.

Case	Relation	Prediction	Difference		Fixed Difference	
(i)	$z^y=+1; \hat{z}^y=+1$	correct	$\mathcal{S}^y-\mathcal{S}_2 \geq 0$	$\downarrow$	$z^y(\mathcal{S}^y-\mathcal{S}_2) \geq 0$	$\downarrow$
(ii)	$z^y=+1; \hat{z}^y=-1$	incorrect	$\mathcal{S}^y-\mathcal{S}_1 \leq 0$	$\downarrow$	$z^y(\mathcal{S}^y-\mathcal{S}_1) \leq 0$	$\downarrow$
(iii)	$z^y=-1; \hat{z}^y=+1$	incorrect	$\mathcal{S}^y-\mathcal{S}_2 \geq 0$	$\uparrow$	$z^y(\mathcal{S}^y-\mathcal{S}_2) \leq 0$	$\downarrow$
(iv)	$z^y=-1; \hat{z}^y=-1$	correct	$\mathcal{S}^y-\mathcal{S}_1 \leq 0$	$\uparrow$	$z^y(\mathcal{S}^y-\mathcal{S}_1) \geq 0$	$\downarrow$

This prediction process resembles the student's skiping out of the gate game where the gate is set by the teacher. The gate (*i.e.* teacher's score) reflects how strongly the teacher has confidence about an instance's belonging to the target class. And this gate actually serves as a threshold for the student's judgement. This is different from the SVM models where all examples have the same hard threshold 0, and it is more adaptive than SVM since each example has own different threshold given by the teacher. Besides, in SVM scores larger than the threshold are predicted to be positive while in Eq.(4) to cohere with the teacher's maximization prediction rule, scores smaller than the "gate" are considered to be with positive responses (*i.e.* labels). In this way, the prediction is mutually determined by the cooperation of the teacher and the student.

Now we investigate how to learn a good rejection confidence estimator for the students. Suppose the ground-truth of all training examples on class y is $\mathbf{z}^y \in \{+1,-1\}^n$, then if $y=\hat{y}$ in Eq.(2), $z_i^y=+1$; otherwise, $z_i^y=-1$, $\forall\, i=1,...,n$. Assume the rejection confidence estimator is a linear function, *i.e.* $\varphi(\mathbf{x};\boldsymbol{\theta})=\langle\boldsymbol{\theta},\phi(\mathbf{x})\rangle$, where $\phi(\mathbf{x})\in\mathbb{R}^m$ is the feature vector of $\mathbf{x}$. For each example $\mathbf{x}_i$, the rejection confidence should observe the inequality

$$z_i^y\left\langle\boldsymbol{\theta},\phi(\mathbf{x}_i)\right\rangle \leq z_i^y\left\langle\boldsymbol{w}^*,\psi(\mathbf{x}_i,y)\right\rangle \tag{5}$$

which means that if $\mathbf{x}_i$ belongs to class y ($z_i^y=+1$), then we should have $\langle\boldsymbol{\theta},\phi(\mathbf{x}_i)\rangle \leq \langle\boldsymbol{w}^*,\psi(\mathbf{x}_i,y)\rangle$; otherwise, $\langle\boldsymbol{\theta},\phi(\mathbf{x}_i)\rangle \geq \langle\boldsymbol{w}^*,\psi(\mathbf{x}_i,y)\rangle$. To expect a significant gap (or margin) between the inequalities, we push the inequalies harder a bit by introducing a constant ε,

$$\varepsilon + z_i^y\left\langle\boldsymbol{\theta},\phi(\mathbf{x}_i)\right\rangle \leq z_i^y\left\langle\boldsymbol{w}^*,\psi(\mathbf{x}_i,y)\right\rangle \tag{6}$$

This order inequality is consistent with the prediction rule Eq.(4); however, Eq.(6) serves as a hard constraint and may be easily destroyed by the outliers in examples or fatal errors from teacher. To enhance the robustness, we introduce a slack variable $\xi_i \geq 0$ to control the tolerance for prediction mistakes, *i.e.*

$$\varepsilon + z_i^y\left\langle\boldsymbol{\theta},\phi(\mathbf{x}_i)\right\rangle \leq z_i^y\left\langle\boldsymbol{w}^*,\psi(\mathbf{x}_i,y)\right\rangle + \xi_i. \tag{7}$$

Our goal is to minimize the prediction mistakes for the student, then the leanring model can be formulated in SVM style:

$$\begin{aligned}&\min_{\boldsymbol{\theta},\boldsymbol{\xi}}\ \frac{1}{2}\left\|\boldsymbol{\theta}\right\|_2^2+\frac{C}{n}\sum_{i=1}^{n}\xi_i\\ &\text{s.t. } \varepsilon + z_i^y\left\langle\boldsymbol{\theta},\phi(\mathbf{x}_i)\right\rangle \leq z_i^y\left\langle\boldsymbol{w}^*,\psi(\mathbf{x}_i,y)\right\rangle + \xi_i.\\ &\quad \xi_i \geq 0,\ \forall\, i=1,2,...,n.\end{aligned} \tag{8}$$

Algorithm 1 Difficulty Coding with Single-Teacher

```
Input: Teacher model (SSVM) with classifier $\boldsymbol{w}^*$ and joint feature operator $\psi(\mathbf{x},y)$, an example $\mathbf{x}$ with the gound-truth response $z^y$ for class $y$.
1: Calculate the teacher's prediction score vector $\mathcal{S}$ as Eq.(3), with the maximum score and second maximum score being $\mathcal{S}_1$ and $\mathcal{S}_2$, repectively.
2: Denote the score of class $y$ as $\mathcal{S}^y$, and calculate the teacher's prediction response $\hat{z}^y$.
3: Calculate the difference $\delta$ as Eq.(9).
4: Calculate the difficulty degree $d$ as Eq.(11).
Output: Example $\mathbf{x}$'s difficulty degree $d$ in Eq.(10).
```

The first $(1/2)\left\|\boldsymbol{\theta}\right\|_2^2$ is a regularization term; the second term is actually the ℓ_1 norm of slack variable vector $\boldsymbol{\xi}=[\xi_1,...,\xi_n]^T$ to encourage the sparisity. A sparse $\boldsymbol{\xi}$ implies that only a fraction of examples make predition mistakes. $C>0$ is a constant to balance both two terms.

Going Further: Difficulty among Examples

In Eq.(8) for each example, the teacher's score on the target class serves as the gated information for the student's prediction. In the following, we reveal that the teacher's scores on the other classes also have beneficial information for the student's learning the target class.

For other non-target classes, an intuition is that if their score values are much closer to that of target class, then we think the example is more difficult for the teacher to distinguish whether it belongs to the target class or not. Moreover, since the teacher is not an Oracle, he/she also makes mistakes in the prediction. Thus with the ground-truth label responses, the incorrectly predicted examples are regarded more difficult than those corrected predicted for the teacher. We shall measure the difficulty of examples from both two aspects in the sequel, *i.e.* the correctness and closeness.

For target class y and an example $\mathbf{x}$, its ground-truth response and the prediction response of the teacher $\mathcal{T}$ are denoted as z^y and $\hat{z}^y$, respectively. Denote the score vector on all K classes as $\mathcal{S}$, with the score of target class as $\mathcal{S}^y$. There are totally four cases for z^y and $\hat{z}^y$.

(i) $z^y=+1; \hat{z}^y=+1$. In this case, the teacher predicted correctly, with the $\mathcal{S}^y$ being the maximum value of $\mathcal{S}$. Then the difference between $\mathcal{S}^y$ and the second largest score $\mathcal{S}_2$ reflects the closeness. The smaller (closer) the difference $\mathcal{S}^y-\mathcal{S}_2 \geq 0$ is, the more difficult the example $\mathbf{x}$ is for the teacher.

(ii) $z^y=+1; \hat{z}^y=-1$. The teacher predicts incorrectly.

Algorithm 2 Training for gated SVM (gSVM) with Majorization Minimization optimization

```
Input: Training data: for each example $\mathbf{x}_i$ its feature vector $\phi(\mathbf{x}_i)$ and their labels $\mathbf{y} \in \mathcal{Y}^n$. A teacher model $\mathcal{T}$. Learning
    parameters: $C > 0$.
 1: Calculate the teacher $\mathcal{T}$'s prediction $\hat{\mathbf{y}}$ and the score matrix $S$.
 2: for each $y$ in $\mathcal{Y}$ do
 3:     For class $y$, calculate the ground-truth response $\mathbf{z}^y$, the teacher's prediction $\hat{\mathbf{z}}^y$ and the each example $\mathbf{x}_i$'s score $\mathcal{S}_i^y$ in the
        score matrix $S$.
 4:     Calculate the difficulty degree $d_i$ of each example according to Algorithm 1.
 5:     Initialize the classifier $\boldsymbol{\theta}$ for class $y$
 6:     while not convergence do
 7:         Calculate the residual $\vartheta_i = z_i^y \phi_{\mathbf{x}_i}^T \boldsymbol{\theta} - z_i^y \mathcal{S}_i^y$ for each example $\mathbf{x}_i$.
 8:         Update the classifier $\boldsymbol{\theta} \leftarrow \frac{C}{2n}(\mathbf{I} + \frac{C}{2n}\Phi Z^y D^{-1}|\Upsilon|^{-1} Z^y \Phi^T)^{-1} \Phi Z^y D^{-1} |\Upsilon|^{-1} (\boldsymbol{\mu}^y - |\boldsymbol{\vartheta}|)$
 9:     end while
10:     Output the binary classifier $\boldsymbol{\theta}^y \leftarrow \boldsymbol{\theta}$ for class $y$.
11: end for
```

The maximum score of $\mathcal{S}$ is not $\mathcal{S}^y$. Then the difference between $\mathcal{S}^y$ and the maximum score $\mathcal{S}_1$ mirrors the degree of the incorrectness. The larger the absolute of the difference $\mathcal{S}^y - \mathcal{S}_1 \leq 0$, the more difficult the example is.

(iii) $z^y = -1; \hat{z}^y = +1$. The teacher predicts incorrectly. The maximum score predicted by the teacher should not the in the top position. A larger difference $\mathcal{S}^y - \mathcal{S}_2 \geq 0$ indicates a more difficult example.

(iv) $z^y = -1; \hat{z}^y = -1$. The teacher predicts correctly. The difference $\mathcal{S}^y - \mathcal{S}_1 \leq 0$ corresponds to the closeness. The larger its abosolute is, the more difficlut the example is.

All four cases have their own depiction of the difficulty of examples. We clarify them in Table 1. In addition, it is our consensus that incorrectly predicted examples are assumed to be more difficult than those correctly predicted. To make the differences consistent with this assumption, we introduce a fixed difference in Table 1. As in Table 1, the correctly predicted examples always have non-negative differences while the incorrectly predicted examples have non-positive ones. Thus the differences of incorrect examples are smaller than those of correct examples. Moreover, the smaller the differences are, the more difficult the examples are supposed to be. To make the differences more compact, we rewrite them into one expression,

$$\delta(\mathbf{x}; \mathcal{T}) = \frac{\mathcal{S}^y - \hat{z}^y \min(\hat{z}^y \mathcal{S}_1, \hat{z}^y \mathcal{S}_2)}{z^y |\mathcal{S}^y|}. \tag{9}$$

which is consistent with the Table 1, and the absolute of $\mathcal{S}^y$ in the denominator serves as a normalization constant.

The difference δ is capable of reflecting the difficulty of examples for the teacher. A student can use this difficulty information to regularize his/her learning the target class. In particular, the slack variable ξ_i in Eq.(7) indicates the tolerance for the mistakes. Generally speaking, if an example is more difficult, then we have more tolerance for the mistake; otherwise, we have less tolerance. Thus we can embed the difficulty degree into the student's learning via rescaling the slack variable,

$$z_i^y \langle \boldsymbol{\theta}, \phi(\mathbf{x}_i) \rangle \leq z_i^y \langle \boldsymbol{w}^*, \psi(\mathbf{x}_i, y) \rangle + d_i \xi_i. \tag{10}$$

where the d_i is the difficulty degree presented by the teacher. The difficult examples are expected to have large difficulty degree d. However, the difference δ can be positive or negative, we propose to generate the difficulty degree d by directly mapping the difference in the interval [0,1]. In detail, a monotonically decreasing non-negative coding function $f(\cdot)$ is needed, for example, a filpped clip function, *i.e.*

$$d = f(\delta) = \begin{cases} 1 & \text{if } \delta < -1 \\ \frac{\epsilon - 1}{2}\delta + \frac{\epsilon + 1}{2}, & \text{if } -1 \leq \delta \leq 1 \\ \epsilon, & \text{if } \delta > 1 \end{cases} \tag{11}$$

where $0 \leq \epsilon \leq 1$ is a constant controlling the decay. We show the coding function in Figure 1 and elobarate the difficulty coding process in Algorithm 1.

Embedding the difficulty degree into Eq.(8), we obtain the gated SVM, *i.e.*

$$\begin{aligned} \min_{\boldsymbol{\theta}, \boldsymbol{\xi}} \ & \frac{1}{2} \|\boldsymbol{\theta}\|_2^2 + \frac{C}{n} \sum_{i=1}^n \xi_i \\ \text{s.t. } & \varepsilon + z_i^y \langle \boldsymbol{\theta}, \phi(\mathbf{x}_i) \rangle \leq z_i^y \langle \boldsymbol{w}^*, \psi(\mathbf{x}_i, y) \rangle + d_i \xi_i. \\ & \xi_i \geq 0, \ \forall\, i = 1, 2, ..., n. \end{aligned} \tag{12}$$

As in Eq.(12), the more difficult (lager d_i) examples will be allowed for larger mistake errors. In this way, the student can learn more adaptively with different examples. To distinguish Eq.(12) from Eq.(8) which is with no difficulty coding, we call Eq.(12) gSVM and Eq.(8) gSVM-0.

Remark 1. Substituing $d_i \xi_i$ with υ_i, the objective of Eq.(12) will be rewritten as $\frac{1}{2} \|\boldsymbol{\theta}\|_2^2 + \frac{C}{n} \sum_{i=1}^n d_i^{-1} \upsilon_i$ in term of variables $\boldsymbol{\theta}$ and $\boldsymbol{\upsilon}$. This formulation is similar with the idea of weighted SVM (Lapin, Hein, and Schiele 2014), where each of example is assigned an importance weight d_i^{-1}.

Remark 2. Since the teacher model has k times as many classifiers than a binary SVM, the computation cost is linear with the class number k. Note that the teacher model can be speeded up for its binary inference via a traverse technique: as soon as any class with a larger value than that of the target class is found, one can stop traversing the classes in Eq.(2).

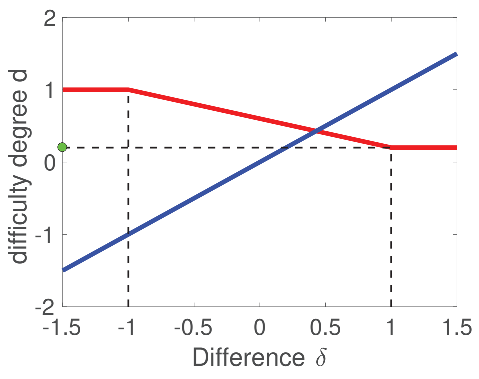

Figure 1: Difficulty coding function from the difference δ. The blue line is the original difference values, and the red line is the corresponding difficulty degrees. The green mark indicates the constant ϵ in Eq.(11).

But this only works on those which are predicted to be -1 by the teacher. For those +1 predicted examples, we still have to traverse all classes to make sure the target class has the largest score. Even for the -1 predicted examples. the worst case still needs to traverse all classes. We will show this in our experiments.

Optimization

In this section, we elaborate how to solve the proposed gated SVM Eq.(12). gSVM is a convex problem, which can be efficiently solved by many off-the-shelf toolboxes, such as mosek (Mosek 2010) and cvx (Grant, Boyd, and Ye 2008). By dint of Eq.(12), we rewrite it into an empirical risk minimization problem,

$$\min_{\boldsymbol{\theta}} \frac{1}{2}\|\boldsymbol{\theta}\|_2^2+\frac{C}{n}\sum_{i=1}^{n} d_i^{-1}[\varepsilon+z_i^y\langle\boldsymbol{\theta},\phi(\mathbf{x}_i)\rangle-z_i^y\mathcal{S}_i^y]_+ \quad (13)$$

where $\mathcal{S}_i^y = \langle \boldsymbol{w}^*, \psi(\mathbf{x}_i, y)\rangle$ and $[x]_+ = \max(x, 0)$ is a rectified linear unit (ReLU) function. Eq.(13) is unconstrained; however, the loss function is not smooth and non-differentiable. Majorization minimization (MM) algorithms substitute a simple optimization problem for a difficult optimization problem (Hunter and Lange 2004; Mairal 2015; Lange 2016). In the sequel, we present how to optimize Eq.(13) with an MM algorithm. First, we throw light on the basic paradigm of MM.

Majorization Minimization (MM)

MM liberates the optimization of a non-comvex or/and non-smooth objective function by iteratively minimizing a majorization suggogate function. During the optimization, it actively constructs a majorizer $\mathcal{M}(\boldsymbol{\theta};\boldsymbol{\theta}^{(k)})$ at the corrent iterate $\boldsymbol{\theta}^{(k)}$, which is assumed to be easy to manipulate. Suppose the objective function is $g(\boldsymbol{\theta})$. The majorizer $\mathcal{M}$ is supposed to satisfy two properties:

$$g(\boldsymbol{\theta}) \le \mathcal{M}(\boldsymbol{\theta};\boldsymbol{\theta}^{(k)}),\ \forall\, \boldsymbol{\theta} \quad (14)$$

$$\boldsymbol{\theta}^{(k)} = \arg\min_{\boldsymbol{\theta}}\ \mathcal{M}(\boldsymbol{\theta};\boldsymbol{\theta}^{(k)}) - g(\boldsymbol{\theta}). \quad (15)$$

Then the next iterate is presented by minimizing the majorizer instead of the original objective:

$$\boldsymbol{\theta}^{(k+1)} = \arg\min_{\boldsymbol{\theta}}\ \mathcal{M}(\boldsymbol{\theta};\boldsymbol{\theta}^{(k)}). \quad (16)$$

As a result, we can have a sequence of iterates enabling the previous objective $g(\boldsymbol{\theta})$ to keep non-increasing.

Iterative Update

The difficulty of optimizing Eq.(13) comes from the non-smoothness of ReLU function. In our paper, we adopt a majorizer proposed in (Nguyen and McLachlan 2017), presented in Lemma 1.

Lemma 1 ((Nguyen and McLachlan 2017))**.** *If $v \neq 0$, then the function $[u]_+$ is majorized at v by*

$$\mathcal{M}(u;v) = \frac{1}{4|v|}(u+|v|)^2. \quad (17)$$

Besides, every function can be majorized by itself, including the first term $\frac{1}{2}\|\boldsymbol{\theta}\|_2^2$ in Eq.(13). Following Lemma 1 and letting $u = \varepsilon + z_i^y\phi_{\mathbf{x}_i}^T\boldsymbol{\theta} - z_i^y\mathcal{S}_i^y$, $v = \varepsilon + z_i^y\phi_{\mathbf{x}_i}^T\boldsymbol{\theta}^{(k)} - z_i^y\mathcal{S}_i^{y}$[1], then the majorizer is as

$$\mathcal{M}(\boldsymbol{\theta};\boldsymbol{\theta}^{(k)}) = \frac{C}{n}\sum_{i=1}^{n}\frac{(\varepsilon+z_i^y\phi_{\mathbf{x}_i}^T\boldsymbol{\theta}-z_i^y\mathcal{S}_i^y+|\vartheta_i^{(k)}|)^2}{4d_i|\vartheta_i^{(k)}|}+\frac{1}{2}\|\boldsymbol{\theta}\|_2^2 \quad (18)$$

where $\vartheta_i^{(k)} = \varepsilon + z_i^y\phi_{\mathbf{x}_i}^T\boldsymbol{\theta}^{(k)} - z_i^y\mathcal{S}_i^y$. Let $\boldsymbol{\vartheta}^{(k)} \in \mathbb{R}^n = [\vartheta_1^{(k)}, ..., \vartheta_n^{(k)}]^T$, $\mathbf{z}^y = [z_1^y, ..., z_n^y]^T$, $Z^y = \mathrm{diag}(\mathbf{z}^y)$, $\Phi = [\phi(\mathbf{x}_1), ..., \phi(\mathbf{x}_n)] \in \mathbb{R}^{m\times n}$, $\boldsymbol{d} = [d_1, ..., d_n]^T$, $D = \mathrm{diag}(\boldsymbol{d})$, $\Upsilon^{(k)} = \mathrm{diag}(\boldsymbol{\vartheta}^{(k)})$ and $\boldsymbol{\mu}^y \in \mathbb{R}^n$ with each element $\mu_i^y = z_i^y\mathcal{S}_i^y$. Note that we usually add a small positive number (*e.g.* 10^{-6}) to the $|\vartheta_i^{(k)}|$ in the denominator in Eq.(18) for the numerical stability. Then Eq.(18) can be rewritten as

$$\mathcal{M}(\boldsymbol{\theta};\boldsymbol{\theta}^{(k)}) = \frac{1}{2}\|\boldsymbol{\theta}\|_2^2 + \frac{C}{4n}(\boldsymbol{\eta}^{(k)})^T D^{-1}|\Upsilon^{(k)}|^{-1}\boldsymbol{\eta}^{(k)} \quad (19)$$

with

$$\boldsymbol{\eta}^{(k)}(\boldsymbol{\theta}) = \boldsymbol{\varepsilon} + Z^y\Phi^T\boldsymbol{\theta} - \boldsymbol{\mu}^y + |\boldsymbol{\vartheta}^{(k)}| \quad (20)$$

where $|\cdot|$ is element-wise absolute and $\boldsymbol{\varepsilon}$ is a vector with elements being all ε. Eq.(19) is a positive quadratic form. To minimize the $\mathcal{M}(\boldsymbol{\theta};\boldsymbol{\theta}^{(k)})$, we zero the derivative of $\boldsymbol{\theta}$,

$$\begin{aligned}\frac{\partial\mathcal{M}}{\partial\boldsymbol{\theta}} &= \boldsymbol{\theta} + \frac{C}{2n}\Phi Z^y D^{-1}|\Upsilon^{(k)}|^{-1}(\boldsymbol{\varepsilon} + Z^y\Phi^T\boldsymbol{\theta} - \boldsymbol{\mu}^y + |\boldsymbol{\vartheta}^{(k)}|)\\ &= A^{(k)}\boldsymbol{\theta} - \boldsymbol{b}^{(k)} = \mathbf{0}\end{aligned} \quad (21)$$

where

$$A^{(k)} = \mathbf{I} + \frac{C}{2n}\Phi Z^y D^{-1}|\Upsilon^{(k)}|^{-1}Z^y\Phi^T \quad (22)$$

$$= \mathbf{I} + \frac{C}{2n}\Phi D^{-1}|\Upsilon^{(k)}|^{-1}\Phi^T \quad (23)$$

$$\boldsymbol{b}^{(k)} = \frac{C}{2n}\Phi Z^y D^{-1}|\Upsilon^{(k)}|^{-1}(\boldsymbol{\mu}^y - |\boldsymbol{\vartheta}^{(k)}| - \boldsymbol{\varepsilon}). \quad (24)$$

[1]We use $\phi_{\mathbf{x}_i}$ to take place of $\phi(\mathbf{x}_i)$ for briefness.

Table 2: Data statistics. #train is the number of training examples while #test is that of test ones. #features and #classes are the number of features and classes, respectively.

Dataset	#train	#test	#features	#classes
letter	15000	5000	16	26
protein	17766	6621	357	3
satimage	4435	2000	36	6
shuttle	43500	14500	9	7
vowel	528	462	10	11
yeast	1185	299	8	10

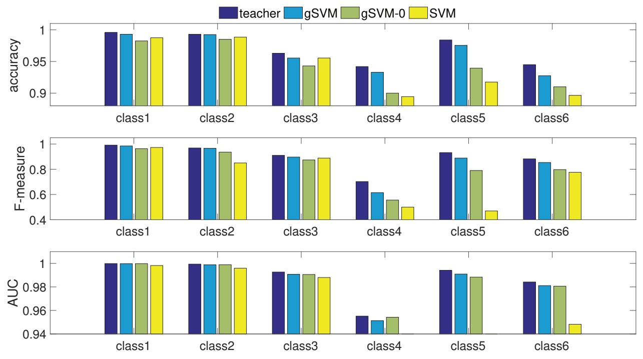

Figure 2: Performance of binary classifiers with respect to each class on satimage dataset. Note that for the visual clarity, some results of SVM's AUC are not shown since they are smaller than 0.94.

Due to $A^{(k)}$'s being positive definite, the next iterate $\boldsymbol{\theta}^{(k+1)}$ can be easily obtained by

$$\boldsymbol{\theta}^{(k+1)} = (A^{(k)})^{-1}\boldsymbol{b}^{(k)} \tag{25}$$

The algorithm outline is presented in Algorithm 2. In addition, the majorizer Eq.(19) is also a least-squares with varying weight, and Algorithm 2 can also be regarded as an iteratively-reweighted least-squares (IRLS) algorithm (Navia-Vazquez et al. 2001). (Nguyen and McLachlan 2017) demonstrated it is globally convergent to the minimum.

Experimental Results

In this part, we experimentally investigate the effectiveness and efficiency of the proposed gated SVM in dealing with the STMS problem.

Configuration

We first illustrate our experimental configuration, including some preliminaries and the specific setting.

Datasets. We used 6 benchmark multi-class datasets in our experiment, including letter (Hsu and Lin 2002), protein (Wang 2002), satimage (Hsu and Lin 2002), shuttle (Hsu and Lin 2002), vowel and yeast.[2] The various statistics of all datasets are presented in Tabel 2.

[2]Datasets can be downloaded in https://www.csie.ntu.edu.tw/cjlin/libsvmtools/datasets/multiclass.html

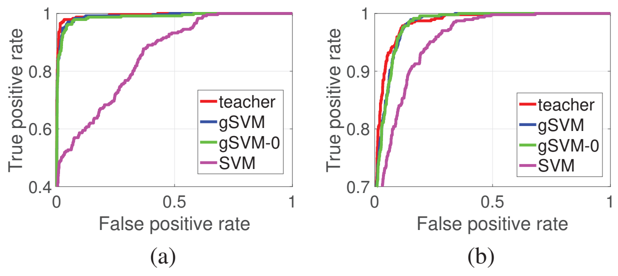

Figure 3: ROC curves for two classes on satimage dataset.

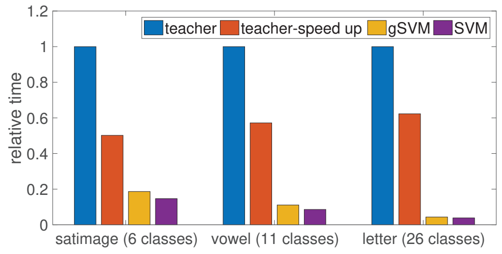

Figure 4: Relative inference time to the teacher model, *i.e.* the specific time devided by the time of the teacher model. We do not show the time of gSVM-0 since it is the same with gSVM in inference stage.

Comparison methods. Since we first deal with the STMS problem, there are few comparison methods, but the following can serve as solutions.
1) Just using the teacher model (denoted as "teacher"). For the target class, we can directly use the teacher's prediction results; if other classes are predicted by the teacher, then we think the examples do not belong to the target class.
2) Directly training a binary classifier for the target class. In our experiment, we use the widely-used SVM method (denoted as "SVM").
3)To verify the effectiveness of the proposed difficulty coding from the teacher, we chose gSVM-0 (Eq.(8)) as a natural comparison method.

Evaluation metrics. We select four evaluation metrics to comprehensively assess the classification performance of the learned binary classifiers, including accuracy, F-measure, receiver operating characteristic curve (ROC curve) and area under the curve (AUC).

Teacher model. SSVM is used for obtaining the teacher models. In particular, we use the joint feature map $\psi(\mathbf{x}, y)$ to be $(\mathcal{I}(y=1)\phi(x), ..., \mathcal{I}(y=K)\phi(x))^T$, where $\mathcal{I}(\cdot)$ is an indicator function, and the distance measure is Hamming distance, *i.e.* $\Delta(y, y_i) = \mathcal{I}(y \neq y_i)$.[3]

Experimental setting. For each dataset, with the trained teacher model (*i.e.* a multiclass classifier), we train a binary classifier for each of its class. The parameters C, ε and ϵ are tuned by 10-fold cross validation in

[3]Actually, this formulation also coheres with the form of Crammer-Singer multiclass SVM (Crammer and Singer 2001).

Table 3: The mean and median of classification performance of all binary classifiers in term of all classes for each dataset.

		mean			median		
dataset	method	accuracy	F-measure	AUC	accuracy	F-measure	AUC
letter	teacher	99.72%	96.35%	99.95%	99.73%	96.43%	99.97%
	SVM	97.35%	43.47%	90.89%	96.90%	44.37%	93.40%
	gSVM-0	98.41%	81.75%	98.64%	98.47%	80.19%	98.70%
	gSVM	98.88%	82.26%	99.72%	99.16%	87.50%	99.81%
protein	teacher	80.50%	67.55%	84.91%	81.03%	71.86%	84.34%
	SVM	78.31%	60.86%	81.41%	77.57%	64.26%	82.34%
	gSVM-0	78.67%	63.23%	82.54%	78.15%	63.79%	82.88%
	gSVM	81.01%	69.86%	85.03%	81.06%	68.47%	85.23%
satimage	teacher	97.05%	89.80%	98.76%	97.35%	92.13%	99.34%
	SVM	94.00%	74.28%	91.46%	93.65%	81.31%	96.81%
	gSVM-0	94.33%	81.95%	98.54%	94.12%	83.56%	98.95%
	gSVM	96.28%	86.76%	98.55%	96.55%	89.29%	99.08%
shuttle	teacher	99.95%	82.20%	98.81%	99.96%	85.71%	98.99%
	SVM	97.47%	42.83%	82.61%	98.91%	43.33%	88.01%
	gSVM-0	98.13%	77.54%	94.86%	98.34%	73.66%	95.02%
	gSVM	99.57%	80.42%	99.28%	99.61%	83.17%	99.85%
vowel	teacher	91.93%	53.69%	92.12%	93.29%	55.38%	93.10%
	SVM	89.95%	22.01%	80.00%	89.91%	24.55%	85.11%
	gSVM-0	86.19%	43.03%	91.05%	85.50%	42.00%	90.04%
	gSVM	90.93%	52.49%	91.96%	92.21%	51.72%	91.92%
yeast	teacher	92.04%	63.59%	85.52%	96.49%	64.12%	83.28%
	SVM	90.47%	35.37%	81.86%	95.82%	36.67%	81.37%
	gSVM-0	91.44%	49.71%	83.00%	95.99%	53.08%	84.36%
	gSVM	91.71%	51.67%	83.10%	96.55%	55.14%	85.40%

the sets $\{10^{-3}, ..., 10^{0}, ..., 10^{3}\}$, $\{0, 0.1, 0.2, ..., 1.5\}$ and $\{0.1, 0.2, ..., 0.5\}$, respectively.

Results and Analysis

In Figures 2 and 3, we show the performance of accuracy, F-measure, AUC and ROC curve on dataset satimage for example. Since we trained binary classifiers for all classes in terms of each dataset, to comprehensively investigate the performance gap among all competing methods, we calculate the mean and median of all classifiers on each dataset and evaluation metric (Van Hasselt, Guez, and Silver 2016). All obtained results are summarized in Tabel 3.

First, for verifying the effectiveness of the proposed difficulty coding, we can see that gSVM is better than gSVM-0 in most classes in Figure 2, and for the cases in Table 3, gSVM wins gSVM-0 in all cases. This implies that the proposed difficulty coding does play a positive role in enhancing the performance of classifiers. Second, it can be seen that the proposed gSVM is comparable to the teacher model in all classes in Figure 2 and achieves near results with the teacher in Tabel 3. Besides, their ROC curves are very close in Figure 3. The slight superiority of teacher model to a binary SVM in a binary inference may result from the advantages of a well-designed multiclass classifier over the simple One-vs-All strategy. However, since the teacher model has the much larger model size than gSVM, and its inference stage needs to involve all classes, its inference time would be much more than that of gSVM. We report the relative inference time to the teacher on three datasets in Figure 4. We can see that gSVM and SVM have the considerably less time than that of teacher model (even with the speed-up technique). As the number of classes increases, the gap could be much larger. That implies that teacher model is not appropriate for the binary prediction when the number of classes is large. Third, gSVM and SVM have the competing inference time according to Figure 4, however, SVM's classification performance is inferior to that of gSVM in Figures 2, Figure 3 and Table 3. Thus gSVM enjoys the superiority to SVM in classification performance.

To sum up, from empirical results gSVM has the comparable classification performance with the teacher model, but much faster inference speed than the teacher model. As for SVM, it has the worse classification performance than gSVM, though it has similar efficiency with gSVM.

Conclusion

In this paper, we illustrate a new problem that uses a pre-trained teacher model to learn a series of student models. We define this problem as "Single-Teacher Multi-Student" (STMS) problem. By taking the multi-class and binary classification as an example, we investigate how to use a well-trained multiclass classifier to learn a series of binary classifiers with respect to each of its class. To solve this problem, we propose a gated SVM as the model for learning binary classifiers. In gated SVM, the students' prediction is determined by both the teacher and themselves. Moreover, we use the teacher's score output to construct different difficulty degrees for various training examples, so the students' training is guided adaptively by the teacher. Experimental results show the effectiveness of our method and validate our theory.

Acknowledgments

We thank the constructive comments of all reviewers and the support from the NSFC 61375026 and 2015BAF15B00, and ARC Projects FL-170100117, DE-180101438, DP-180103424, DP-140102164 and LP-150100671.

References

Aly, M. 2005. Survey on multiclass classification methods. *Neural Netw* 19.

Bishop, C. M. 2006. *Pattern recognition and machine learning*. springer.

Crammer, K., and Singer, Y. 2001. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research* 2(Dec):265–292.

Du, Y.; Xu, C.; and Tao, D. 2017. Privileged matrix factorization for collaborative filtering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, 1610–1616.

Grant, M.; Boyd, S.; and Ye, Y. 2008. Cvx: Matlab software for disciplined convex programming.

Guo, Y. 2017. Problems in large-scale image classification. In *AAAI*, 5038–5039.

He, R.; Wu, X.; Sun, Z.; and Tan, T. 2017. Learning invariant deep representation for nir-vis face recognition. In *AAAI*, 2000–2006.

He, H.; Eisner, J.; and Daume, H. 2012. Imitation learning by coaching. In *Advances in Neural Information Processing Systems*, 3149–3157.

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

Hsu, C.-W., and Lin, C.-J. 2002. A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks* 13(2):415–425.

Hunter, D. R., and Lange, K. 2004. A tutorial on mm algorithms. *The American Statistician* 58(1):30–37.

Joachims, T.; Finley, T.; and Yu, C.-N. J. 2009. Cutting-plane training of structural svms. *Machine Learning* 77(1):27–59.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 133–142. ACM.

Lange, K. 2016. *MM Optimization Algorithms*. SIAM.

Lapin, M.; Hein, M.; and Schiele, B. 2014. Learning using privileged information: Svm+ and weighted svm. *Neural Networks* 53:95–108.

Mairal, J. 2015. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization* 25(2):829–855.

Martinez, A. M. 1998. The ar face database. *CVC technical report*.

Mosek, A. 2010. The mosek optimization software. *Online at http://www. mosek. com* 54.

Navia-Vazquez, A.; Perez-Cruz, F.; Artes-Rodriguez, A.; and Figueiras-Vidal, A. R. 2001. Weighted least squares training of support vector classifiers leading to compact and adaptive schemes. *IEEE Transactions on Neural Networks* 12(5): 1047 – 1059.

Nguyen, H. D., and McLachlan, G. J. 2017. Iteratively-reweighted least-squares fitting of support vector machines: A majorization–minimization algorithm approach. *arXiv preprint arXiv:1705.04651*.

Nie, F.; Wang, X.; and Huang, H. 2017. Multiclass capped lp-norm svm for robust classifications.

Tsochantaridis, I.; Hofmann, T.; Joachims, T.; and Altun, Y. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, 104. ACM.

Van Hasselt, H.; Guez, A.; and Silver, D. 2016. Deep reinforcement learning with double q-learning. In *AAAI*, 2094–2100.

Vapnik, V. 2013. *The nature of statistical learning theory*. Springer science & business media.

Wang, J.-Y. 2002. Application of support vector machines in bioinformatics. *Taipei: Department of Computer Science and Information Engineering, National Taiwan University*.

Xu, W.; Sun, H.; Deng, C.; and Tan, Y. 2017. Variational autoencoder for semi-supervised text classification. In *AAAI*, 3358–3364.

Xu, C.; Tao, D.; and Xu, C. 2013. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*.

Yang, E.; Deng, C.; Liu, W.; Liu, X.; Tao, D.; and Gao, X. 2017. Pairwise relationship guided deep hashing for cross-modal retrieval. In *AAAI*, 1618–1625.

You, S.; Xu, C.; Wang, Y.; Xu, C.; and Tao, D. 2017a. Privileged multi-label learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence,* 3336–3342. IJCAI 2017, Melbourne, Australia, August 19-25, 2017

You, S.; Xu, C.; Xu, C.; and Tao, D. 2017b. Learning from multiple teacher networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1285–1294. ACM.

Yu, C.-N. J., and Joachims, T. 2009. Learning structural svms with latent variables. In *Proceedings of the 26th annual international conference on machine learning*, 1169–1176. ACM.

Zhu, X.; Liu, J.; and Lopes, M. 2017. No learner left behind: On the complexity of teaching multiple learners simultaneously. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, 3588–3594.