Curriculum Learning of Multiple Tasks

Anastasia Pentina, Viktoriia Sharmanska, Christoph H. Lampert
IST Austria, Am Campus 1, 3400 Klosterneuburg, Austria
{apentina,vsharman,chl}@ist.ac.at

Abstract

Sharing information between multiple tasks enables algorithms to achieve good generalization performance even from small amounts of training data. However, in a realistic scenario of multi-task learning not all tasks are equally related to each other, hence it could be advantageous to transfer information only between the most related tasks.

In this work we propose an approach that processes multiple tasks in a sequence with sharing between subsequent tasks instead of solving all tasks jointly. Subsequently, we address the question of curriculum learning of tasks, i.e. finding the best order of tasks to be learned. Our approach is based on a generalization bound criterion for choosing the task order that optimizes the average expected classification performance over all tasks. Our experimental results show that learning multiple related tasks sequentially can be more effective than learning them jointly, the order in which tasks are being solved affects the overall performance, and that our model is able to automatically discover a favourable order of tasks.

1. Introduction

Multi-task learning [6] studies the problem of solving several prediction tasks. While traditional machine learning algorithms can be applied to solve each task independently, they usually need significant amounts of labelled data to achieve generalization of reasonable quality. However, in many cases it is expensive and time consuming to annotate large amounts of data, especially in computer vision applications such as object categorization. An alternative approach is to share information between several related learning tasks and this has been shown experimentally to allow better generalization from fewer training points per task [26].

In this work we focus on the parameter transfer approach to multi-task learning that rests on the idea that models corresponding to related tasks are similar to each other in terms of their parameter representations. We concentrate on the case of linear predictors and assume that similarity between the models is measured by the Euclidean distance between the corresponding weight vectors [35]. In a multi-task setting this idea was introduced by Evgeniou and Pontil in [9]. There the authors propose an SVM-based algorithm that enforces the weight vectors corresponding to different tasks to lie close to some common prototype, and they show its effectiveness on several datasets. However, this algorithm treats all the tasks symmetrically, which might not be optimal in more realistic scenarios. There might be some outlier tasks or groups of tasks such that there is no similarity between the tasks from different groups. Hence, more flexible models are needed that are able to exploit the structure underlying tasks relations and avoid negative consequences of transferring information between unrelated tasks.

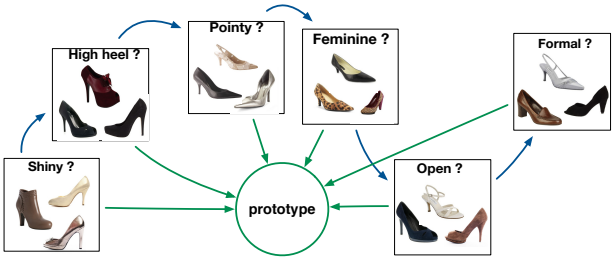

Figure 1. Schematic illustration of the proposed multi-task learning approach. If each task is related to some other task but not equally much to all others, learning tasks in a sequence (blue arrows) can be beneficial to classical multi-task learning based on sharing information from a single prototype (green arrows).

The idea of regularizing by Euclidean distance between the weight vectors of different tasks is also commonly used in domain adaptation scenario where the learner has access to two or more prediction tasks but is interested in performing well only on one of them. All other tasks serve only as sources of additional information. This setup has been shown to lead to effective algorithms in various computer vision applications: object detection [2], personalized image search [17], hand prosthetics [25] and image categorization [33, 34]. Its modification that transfers correlation patterns between the features was used for object detection [11] and recognition [13]. Though the domain adaptation sce-

nario is noticeably different from the multi-task one, as it concentrates on solving only one task instead of all of them, the two research areas are closely related in terms of their methodology and therefore can benefit from each other. In particular, the learning algorithm we propose can be seen as a way to decompose a multi-task problem into a set of domain adaptation problems.

Our approach is motivated by the human educational process. If we consider students at school, they, similarly to a multi-task learner, are supposed to learn many concepts. However, they learn them not all simultaneously, but in a sequence. By processing tasks in a meaningful order, students are able to gradually increase their knowledge and reuse previously accumulated information to learn new concepts more effectively. Inspired by this example we propose to solve tasks in a sequential manner by transferring information from a previously learned task to the next one instead of solving all of them simultaneously. This approach makes learning more flexible in terms of variability between the tasks and memory efficient as it does not require processing all training data at the same time.

As for students at school, the order in which tasks are solved may crucially affect the overall performance of the learner. We study this question by using PAC-Bayesian theory [24] to prove a generalization bound that depends on the data representation and algorithm used to solve the tasks. The bound quantifies the effectiveness of the order in which tasks are solved and therefore can be used to find a beneficial order. Based on the bound we propose a theoretically justified algorithm that automatically chooses a favourable sequence for learning. Our experimental results show that learning tasks in a sequence can be superior to independent learning as well as to the standard multi-task approach of solving them jointly, and that our algorithm is able to reliably discover an advantageous order.

2. Related Work

While our work is based on the idea of transferring information through weight vectors, other approaches to multi-task learning have been proposed as well. A popular idea in the machine learning literature is that parameters of related tasks can be represented as linear combinations of a small number of common latent basis vectors. Argyriou *et al*. proposed a method to learn such representations using sparsity regularization in [1]. This method was later extended to allow partial overlap between groups of tasks in [19]. It was also adapted to the lifelong setting in [31], where Ruvolo and Eaton proposed a way to sequentially update the model as new tasks arrive, and discussed in [27], where a generalization bound for lifelong learning was first presented. In [30], the model was extended to the case when the learner is allowed to choose which task to solve next and several heuristics were proposed for making this choice. Experimentally, subspace-based methods have shown good performance in situations where many tasks are available and the underlying feature representations are low-dimensional. When the feature dimensionality gets larger, however, their computational cost grows quickly, and this makes them not applicable for the type of computer vision problems we are interested in.[1] An exception is [15], where Jayaraman *et al*. apply subspace-based method to jointly learn multiple attribute predictors. However, even there, dimensionality reduction procedure was required.

Methods based on the sharing of weight vector have also been generalized since their original introduction in [9], in particular to relax the assumption that all tasks have to be related. In [8], Evgeniou *et al*. achieved this by introducing a graph regularization. Alternatively, Chen *et al*. [7] proposed to penalize deviations in weight vectors for highly correlated tasks. However, these methods require prior knowledge about the amount of similarities between tasks. In contrast, the algorithm we present in this work does not assume all tasks to be related, yet does not need a priori information regarding their similarities, either.

The question how to order a sequence of learning steps to achieve best performance has previously been studied mainly in the context of single task learning, where the question is in which order one should process the training examples. In [4] Bengio *et al*. showed experimentally that choosing training examples in an order of gradually increasing difficulty can lead to faster training and higher prediction quality. Similarly, Kumar *et al*. [20] introduced the self-paced learning algorithm, which automatically chooses the order in which training examples are processed for solving a non-convex learning problem. In the context of learning multiple tasks, the question in which order to learn them was introduced in [21], where Lad *et al*. proposed an algorithm for optimizing the task order based on pairwise preferences. However, they considered only the setting in which tasks are performed in a sequence through user interaction and therefore their approach is not applicable in the standard multi-task scenario. In the setting of multi-label classification, the idea of decomposing a multi-target problem into a sequence of single-target ones was proposed by Read *et al*. in [29]. However, there the sharing of information occurs through augmentations of the feature vectors, not through a regularization term.

3. Method

In the multi-task scenario a learning system observes multiple supervised learning tasks, for example, recognizing objects or predicting attributes. Its goal is to solve all these tasks by sharing information between them. For-

[1] In preliminary experiments we tried to use ELLA [31], as one of the fastest exiting methods, but found the experiments intractable to do at full size. A simplified setup produced results clearly below that of other baselines.

mally we assume that the learner observes n tasks, denoted by $t_1, \ldots, t_n$, which share the same input and output spaces, $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} = \{-1, +1\}$, respectively. Each task t_i is defined by the corresponding set $S_i = \{(x_1^i, y_1^i), \ldots, (x_{m_i}^i, y_{m_i}^i)\}$ of m_i training points sampled i.i.d. according to some unknown probability distribution D_i over $\mathcal{X} \times \mathcal{Y}$. We also assume that for solving each task the learner uses a linear predictor $f(x) = \text{sign}\langle w, x\rangle$, where $w \in \mathbb{R}^d$ is a weight vector, and we measure the classification performance by the $0/1$ loss, $l(y_1, y_2) = [\![y_1 \neq y_2]\!]$. The goal of the learner is to find n weight vectors $w_1, \ldots, w_n$ such that the average expected error on tasks $t_1, \ldots, t_n$ (given that the predictions are made by the corresponding linear predictors) is minimized:

$$\text{er}(w_1, \ldots, w_n) = \frac{1}{n} \sum_{i=1}^{n} \mathop{\mathbf{E}}_{(x,y)\sim D_i} [\![y \neq \text{sign}\langle w_i, x\rangle]\!]. \quad (1)$$

3.1. Learning in a fixed order

We propose to decompose a multi-task problem of solving n tasks into n domain adaptation problems. Specifically, we assume that the tasks $t_1, \ldots, t_n$ are processed sequentially in some order $\pi \in \mathfrak{S}_n$, where $\mathfrak{S}_n$ is the symmetric group of all permutations over n elements, and information is transferred between subsequent tasks: from $t_{\pi(i-1)}$ to $t_{\pi(i)}$ for all $i = 2, \ldots, n$. In this procedure the previously solved task serves as a source of additional information for the next task and any of the existing domain adaptation methods can be used. In this paper we use an Adaptive SVM [16, 36] to train classifiers for every task due to its proved effectiveness in computer vision applications. For a given weight vector $\tilde{w}$ and training data for a task, the Adaptive SVM performs the following optimization:

$$\min_{w} \quad \|w - \tilde{w}\|^2 + \frac{C}{m} \sum_{j=1}^{m} \xi_j \quad (2)$$

$$\text{sb.t.} \quad y_j \langle w, x_j\rangle \geq 1 - \xi_j, \quad \xi_j \geq 0 \quad \text{for all} \quad 1 \leq j \leq m.$$

Specifically, to learn a predictor for the task $t_{\pi(i)}$ we solve (2) using the weight vector obtained for the previous task, $w_{\pi(i-1)}$, as $\tilde{w}$. For the very first task, $t_{\pi(1)}$, we use the standard linear SVM, i.e. $\tilde{w} = \mathbf{0}$. To simplify the notation, we define $\pi(0)$ to be 0 and w_0 to be the zero vector.

Note that this approach does not rely on the assumption that all the tasks $t_1, \ldots, t_n$ are equally related. However its performance will depend on the order π as it needs subsequent tasks to be related. In the next section we study this question using statistical learning theory and introduce an algorithm for automatically defining a beneficial data-dependent order.

3.2. Learning a data-dependent order

Here we examine the role of the order π in terms of the average expected error (1) of the resulting solutions. However, we do not limit our theoretical analysis to the case of using Adaptive SVMs as described earlier. Specifically, we only assume that the learning algorithm used for solving each individual task $t_{\pi(i)}$ is the same for all tasks and deterministic. This algorithm, $\mathcal{A}(w_{\pi(i-1)}, S_{\pi(i)})$, returns $w_{\pi(i)}$ based on the solution $w_{\pi(i-1)}$ obtained for a previously solved task and training data $S_{\pi(i)}$. The following theorem provides an upper-bound on the average expected error (1) of the obtained predictors (the proof can be found in the supplementary material).

Theorem 1. *For any deterministic learning algorithm $\mathcal{A}$ and any $\delta > 0$, the following inequality holds with probability at least $1-\delta$ (over sampling the training sets $S_1, \ldots, S_n$) uniformly for any order $\pi \in \mathfrak{S}_n$:*

$$\frac{1}{2n} \sum_{i=1}^{n} \mathop{\mathbf{E}}_{(x,y)\sim D_i} [\![y \neq \text{sign}\langle w_i, x\rangle]\!] \leq$$
$$\frac{1}{n} \sum_{i=1}^{n} \left[\frac{1}{m_{\pi(i)}} \sum_{j=1}^{m_{\pi(i)}} \bar{\Phi}\left(\frac{y_j^{\pi(i)} \langle w_{\pi(i)}, x_j^{\pi(i)}\rangle}{||x_j^{\pi(i)}||} \right) + \frac{||w_{\pi(i)} - w_{\pi(i-1)}||^2}{2\sqrt{\bar{m}}} \right] + \frac{1}{8\sqrt{\bar{m}}} - \frac{\log \delta}{n\sqrt{\bar{m}}} + \frac{\log n}{\sqrt{\bar{m}}}, \quad (3)$$

where $\bar{m}$ is the harmonic mean of the sample sizes $m_1, \ldots, m_n$, $\bar{\Phi}(z) = \frac{1}{2}\left(1 - \text{erf}\left(\frac{z}{\sqrt{2}}\right)\right)$, $\text{erf}(z)$ is the Gauss error function [12, 23], $\pi(0) = 0$, $w_0 = \mathbf{0}$ and $w_{\pi(i)} = \mathcal{A}(w_{\pi(i-1)}, S_{\pi(i)})$.

The left hand side of the inequality (3) is one half of the average expected error on tasks $t_1, \ldots, t_n$. This is the quantity of interest that the learner would like to minimize. However, since the underlying data distributions $D_1, \ldots, D_n$ are unknown, it is not computable. In contrast, its upper bound given by the right hand side of (3) contains only computable quantities. It is an average of n terms (up to constants which do not depend on π), where each term corresponds to one task. If we consider the term corresponding to the task $t_{\pi(i)}$, its first part is an analogue of the training error. Each term $\bar{\Phi}\left(y_j^{\pi(i)} \langle w_{\pi(i)}, x_j^{\pi(i)}\rangle ||x_j^{\pi(i)}||^{-1}\right)$ has a value between 0 and 1 and is a monotonically decreasing function of the distance between the training point $x_j^{\pi(i)}$ and the hyperplane defined by $w_{\pi(i)}$. Specifically, it is close to 0 when $x_j^{\pi(i)}$ is correctly classified and has large distance from the separating hyperplane, it is close to 1 when the point is in the wrong halfspace far from the hyperplane and is 0.5 when $x_j^{\pi(i)}$ lies on the hyperplane. Therefore it captures the confidence of the predictions on the training set. The second part of the term corresponding to the task $t_{\pi(i)}$ is a complexity term. It measures the similarity between subsequent tasks $t_{\pi(i-1)}$ and $t_{\pi(i)}$ by the L_2-distance between the obtained weight vectors. As a result the value of the right-hand side of (3)

depends on π and captures the influence that the task $t_{\pi(i)}$ may have on the subsequent tasks $t_{\pi(i+1)}, \ldots, t_{\pi(n)}$. Therefore it can be seen as a quality measure of order π: a low value of the right hand side of (3) ensures a low expected error (1). It leads to an algorithm for obtaining an order π that is adjusted to the tasks $t_1, \ldots, t_n$ by minimizing the right hand side of (3) based on the data $S_1, \ldots, S_n$. Because (3) holds uniformly in π, its guarantees also hold for the learned order[2].

Minimizing the right hand side of (3) is an expensive combinatorial problem, because it requires searching over all possible permutations $\pi \in \mathfrak{S}_n$. We propose an incremental procedure for performing this search approximately. We successively determine $\pi(i)$ by minimizing the corresponding term of the upper bound (3) with respect to yet unsolved tasks. Specifically, at the i-th step, when $\pi(1), \ldots, \pi(i-1)$ are already defined, we search for a task t_k that minimizes the following objective function and is not included in the order π yet:

$$\frac{1}{m_k} \sum_{j=1}^{m_k} \bar{\Phi}\left(\frac{y_j^k \langle w_k, x_j^k \rangle}{||x_j^k||} \right) + \frac{||w_k - w_{\pi(i-1)}||^2}{2\sqrt{\bar{m}}}, \qquad (4)$$

where $w_k = \mathcal{A}(w_{\pi(i-1)}, S_k)$. We let $\pi(i)$ be the index of the task that minimizes (4). Suchwise at each step we choose the task that is easy (has low empirical error) and similar to the previous one (the corresponding weight vectors are close in terms of L_2 norm). Therefore this optimization process well fits the intuitive concept of starting with the simplest task and proceeding with most similar ones. The resulting procedure in the case of using Adaptive SVM (2) for solving each task is summarized in Algorithm 1 and we refer to it as SeqMT. Note that the computational complexity of SeqMT is quadratic in the number of tasks n, because on each step it trains an ASVM for every yet unsolved task (steps 5-7 in Algorithm 1). However, it can be paralellized, since every such ASVM is trained independently from the others.

3.3. Learning with multiple subsequences

The proposed algorithm, SeqMT, relies on the idea that all tasks can be ordered in a sequence, where each task is related to the previous one. In practice, this is not always the case, since we can have outlier tasks that are not related to any other tasks, or we can have several groups of tasks, in which case it is beneficial to form subsequences within the groups, but it is disadvantageous to join them into one single sequence.

Therefore, we propose an extension of the SeqMT model, that allows tasks to form subsequences, where the information is transferred only between the tasks within the subsequence. Our multiple subsequences version, MultiSeqMT, also chooses tasks iteratively, but at any stage it allows the learner to choose whether to continue one of the existing subsequences or to start a new one. In order to decide which task to solve next and which subsequence to continue with it, the learner performs a two-stage optimization. First, for each of the exiting subsequences s (including empty one that corresponds to the no transfer case) the learner finds the task t_s that is the most promising to continue with. This is done in the same way as how the next task is chosen in the SeqMT algorithm. Afterwards, the learner compares the values of criterion (4) for every pair (s, t_s) and chooses the subsequence s^* with the minimal value and continues it with the task t_{s^*}. Please refer to the extended technical report [28] for the exact formulation.

Algorithm 1 Sequential Learning of Multiple Tasks

1: **Input** $S_1, \ldots, S_n$ {training sets}
2: $\pi(0) \leftarrow 0$, $w_0 \leftarrow \mathbf{0}$
3: $T \leftarrow \{1, 2, \ldots, n\}$ {indices of yet unused tasks}
4: **for** $i = 1$ to n **do**
5: **for all** $k \in T$ **do**
6: $w_k \leftarrow$ solution of (2) using S_k, $w_{\pi(i-1)}$
7: **end for**
8: $\pi(i) \leftarrow$ minimizer of (4) w.r.t. k
9: $w_{\pi(i)} \leftarrow w_k$ where $k = \pi(i)$
10: $T \leftarrow T \setminus \{\pi(i)\}$
11: **end for**
12: **Return** $w_1, \ldots, w_n$ and $\pi(1), \ldots, \pi(n)$

4. Experiments

In this section we verify our two main claims: 1) learning multiple tasks in a sequential manner can be more effective than learning them jointly; 2) we can find *automatically* a favourable order in terms of average classification accuracy. We use two publicly available datasets: *Animals with Attributes (AwA)*[3] [22] and *Shoes*[4] [5] augmented with attributes[5] [18]. In the first experiment, we study the case when each task has a certain level of difficulty for learning the object class, which is defined by human annotation in a range from easiest to hardest. We show the advantage of a curriculum learning model over learning multiple tasks jointly and learning each task independently. We also study the automatically determined orders in more details, comparing them with the orders when learning goes from easiest to hardest tasks in the spirit of human learning. In the second experiment, we study the scenario of learning visual

[2] Note that, in contrast, the algorithm $\mathcal{A}$ is assumed to be fixed in advance. Therefore the order π is the only parameter that can be adjusted by minimizing (3) with preservation of the performance guarantees given by Theorem 1.

[3] http://attributes.kyb.tuebingen.mpg.de/
[4] http://tamaraberg.com/attributesDataset/index.html
[5] http://vision.cs.utexas.edu/whittlesearch/

attributes that characterize shoes across different shoe models. In this setting, some tasks are clearly related such as *high heel* and *shiny*, and some tasks are not, such as *high heel* and *sporty*. Therefore, we also apply the variant of our algorithm that allows multiple subsequences, showing that it better captures the task structure and is therefore the favourable learning strategy.

4.1. Learning the order of easy and hard tasks

We focus on eight classes from the *AwA* dataset: *chimpanzee, giant panda, leopard, persian cat, hippopotamus, raccoon, rat, seal*, for which human annotation is available, whether an object is *easy* or *hard* to recognize in an image [32]. For each class the annotation specifies ranking scores of its images from easiest to hardest. To create easy-hard tasks, we split the data in each class into five equal parts with respect to their easy-hard ranking and use these parts to create five tasks per class. Each part has on average 120 samples except the class *rat*, for which AwA contains few images, so there are only approximately 60 samples per part. Each task is a binary classification of one of the parts against the remaining seven classes. For each task we balance 21 vs 21 training images and 77 vs 77 test images (35 vs 35 in case of class *rat*) with equal amount of samples from each of the classes acting as negative examples. The data between different tasks does not overlap. As our feature representation, we use 2000 dimensional bag-of-words histograms obtained from SURF descriptors [3] provided with the dataset. We L_2-normalize the features and augment them with a unit element to act as a bias term.

Evaluation metric. To evaluate the performance we use the classification error rate. We repeat each of the experiments 20 times with different random data splits and measure the average error rate across the tasks. We report mean and standard error of the mean of this value over all repeats.

Baselines. We compare our sequential learning model (SeqMT) with the multi-task algorithm from [9], [10] that treats all tasks symmetrically (MT). Specifically, MT regularizes the weight vectors for all tasks to be similar to a prototype w_0 that is learned jointly with the task weight vectors by solving the following optimization problem:

$$\min_{w_0, w_i, \xi_j^i} \|w_0\|^2 + \frac{1}{n}\sum_{i=1}^{n}\|w_i - w_0\|^2 + \frac{C}{n}\sum_{i=1}^{n}\frac{1}{m_i}\sum_{j=1}^{m_i}\xi_j^i$$
$$\text{subject to } \; y_j^i\langle w_i, x_j^i\rangle \geq 1 - \xi_j^i, \quad \xi_j^i \geq 0 \quad \text{for all } i, j. \quad (5)$$

In order to study how relevant the knowledge transfer actually is, we compare SeqMT with a linear SVM baseline that solves each task independently (IndSVM). As a reference, we also trained a linear SVM on data that is merged from all tasks and outputs one linear predictor for all tasks (MergedSVM).

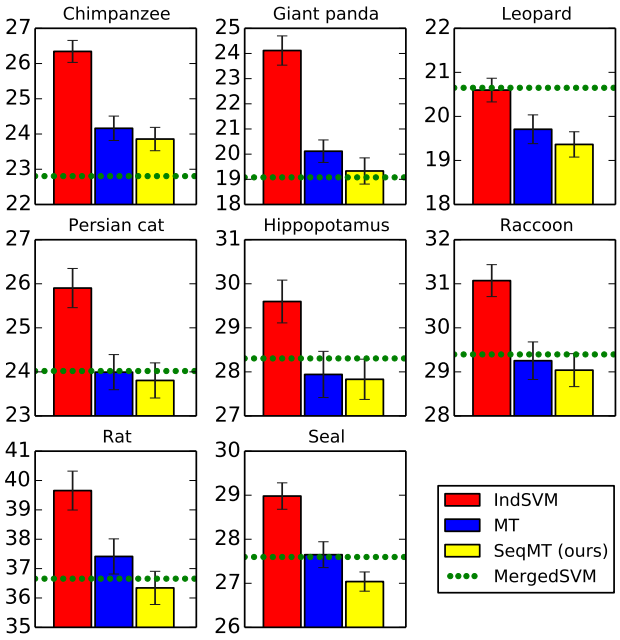

Figure 2. Learning the order of easy and hard tasks on AwA dataset: comparison of the proposed SeqMT method with the multi-task (MT) and the single-task (IndSVM) baselines. The height of the bar corresponds to the average error rate performance over 5 tasks across 20 repeats (the lower the better). As a reference, we also provide the MergedSVM baseline, trained on data that is merged from all tasks. For a complete table with all the results, please refer to the technical report [28].

To understand the impact that the task order has on the classification accuracy we compare the performance of SeqMT with baselines that learn tasks in random order (Random), and in order from easiest to hardest (Semantic) according to the human annotation as if it was given to us. Another baseline we found related is inspired by the diversity heuristic from [30]. It defines the next task to be solved by maximizing (4) instead of minimizing it. We refer to it as Diversity. All the baselines were re-implemented and use the same features across experiments.

Model selection. We perform a cross validation model selection approach for choosing the regularization trade-off parameter C for each of the methods. In all our experiments, we select C over 8 parameter values $\{10^{-2}, 10^{-1} \ldots, 10^{5}\}$ using 5×5 fold cross-validation.

Results. We present the results of this experiment in Figure 2 and Figure 3. As we can see from Figure 2, the proposed SeqMT method outperforms MT and IndSVM algorithms in all 8 cases. This shows that knowledge transfer between the tasks is clearly advantageous in this scenario, and it supports our claim that learning tasks sequentially is more effective than learning them jointly if not all tasks are equally related. As expected, the reference baseline

MergedSVM improves over single-task baseline IndSVM in all but one case, as training with more data has better generalization ability. In some cases, the MergedSVM performs on par or even better than SeqMT and MT methods, as for example, in cases of *chimpanzee* and *giant panda*. We expect that this happens when tasks are so similar that a single hyperplane can explain most of them. In this case, MergedSVM benefits from the amount of data that is available to find this hyperplane. When the tasks are different enough, MergedSVM is unable to explain all of them with one shared hyperplane and loses to SeqMT and MT models, that learn one hyperplane per task. This can be seen, e.g. in the cases of *hippopotamus* and *seal*, and particularly much in the case of *leopard*, where the MergedSVM does not improve even over independent training.

Next we examine the importance of the order in which the tasks are being solved, reporting our findings in Figure 3. All methods in this study use Adaptive SVM as a learning algorithm for solving the next task and differ only by how the order of tasks is defined. In all 8 cases the proposed SeqMT algorithm outperforms the Random order baseline, which learns the tasks in a random order[6]. The Diversity algorithm is much worse than other baselines, presumably because the max heuristic of choosing the next task is not effective in this setting. As a reference, we also check the Semantic baseline when the tasks are being solved from easiest to hardest (as if we had prior information about the easy-hard order of the tasks[7]). In 6 out of 8 classes, the order learned by our SeqMT model (yellow rhombus) is better or on par with the Semantic (green square), except for classes *chimpanzee* and *giant panda*, where we did not manage to learn the best order. Interestingly, for some classes following the strategy of semantic order is worse or on par with learning them in a random order (cases with *seal* and *hippopotamus*). We credit this to the fact that human perception of easiness and hardness does not always coincide with what is easy and hard to learn for a machine learning algorithm. In fact, in cases of *seal* and *hippopotamus*, the human and machine understanding are rather opposite: the hardest task for human is the easiest from machine learning perspective, and the easiest task for human is hardest or medium hard for the learning algorithm. Hence, learning these classes in random order leads to better results than learning in a fixed unfavourable order. We check this by computing the error rates of single SVMs trained per each task: easiest, easy, medium, hard and hardest as defined by human studies and visualize the results in Figure 4.

Finally, for each class we compute the performance of all possible orders to learn 5 tasks, which result in 120 baselines[8]. We visualize the performance of all orders as a violin plot [14], where each horizontal slice of the shaded area reflects how many different orders achieve this error rate (performance stated on the vertical axis). Overall, SeqMT is highly competitive with best possible fixed orders, clearly outperforming them in two cases of *rat* and *seal* (rhombus is lower than the yellow area), and loosing in *chimpanzee*, which we have observed before. Learning the adaptive order of tasks based on the training sets is advantageous to solving them in a fixed order.

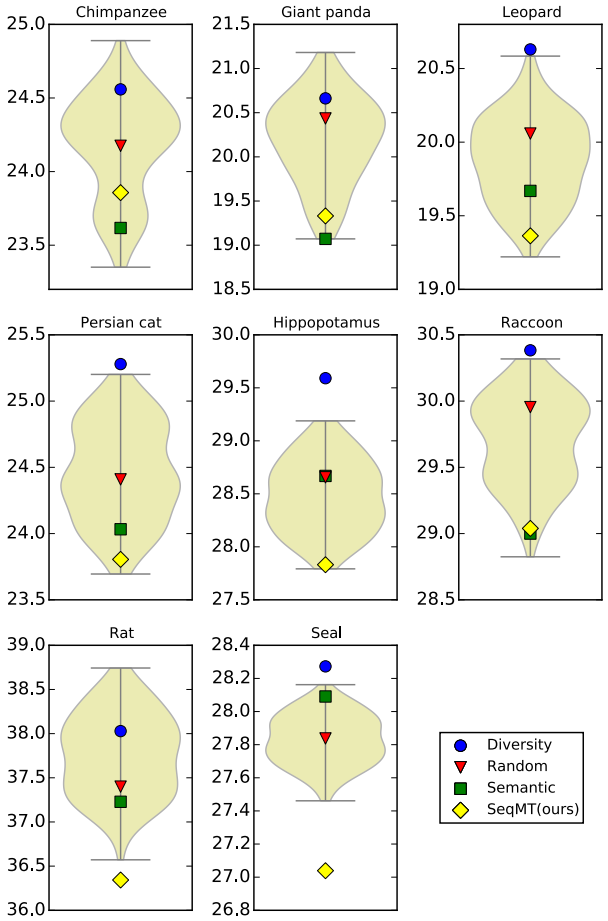

Figure 3. Study of different task order strategies in the experiment with *AwA* dataset. Four main baselines SeqMT, Semantic, Random, Diversity have a distinctive marker and color, and their vertical location captures averaged error rate performance (shown on the vertical axis). The performance of all possible orders is visualized as a background violin plot, where one horizontal slice of the shaded area reflects how many different orders achieve this error rate performance. Note, symmetry of this area is used only for aesthetic purposes. For a complete table with all the results, please refer to the supplementary material. Best viewed in colors.

In order to better understand practical benefits of Theorem 1, we evaluate the dependencies between the performance of a particular task order and its rank according

[6] A different random order is taken for each class for each of the 20 repeats.

[7] This order is fixed for each class for each of the 20 repeats.

[8] One baseline defines one fixed order across all 20 repeats. In SeqMT, we learn an adaptive order that can differ across the repeats.

	Chimpanzee	Giant panda	Leopard	Persian cat	Hippopotamus	Raccoon	Rat	Seal
Error+Compl	$\mathbf{23.86 \pm 0.33}$	$\mathbf{19.33 \pm 0.52}$	$\mathbf{19.36 \pm 0.29}$	$\mathbf{23.81 \pm 0.40}$	$\mathbf{27.83 \pm 0.46}$	$\mathbf{29.04 \pm 0.37}$	$\mathbf{36.34 \pm 0.57}$	$\mathbf{27.04 \pm 0.22}$
Error	24.47 ± 0.42	20.02 ± 0.58	19.97 ± 0.27	24.84 ± 0.46	29.07 ± 0.55	29.75 ± 0.31	38.00 ± 0.54	28.27 ± 0.38
Compl	23.94 ± 0.32	19.44 ± 0.50	$\mathbf{19.36 \pm 0.29}$	$\mathbf{23.81 \pm 0.40}$	$\mathbf{27.83 \pm 0.46}$	$\mathbf{29.04 \pm 0.37}$	$\mathbf{36.34 \pm 0.57}$	$\mathbf{27.04 \pm 0.22}$

Table 1. Trade-off between complexity and error terms in the proposed SeqMT strategy of choosing next task (4) on the AwA dataset. The numbers are average error rate performance over 5 tasks across 20 repeats.

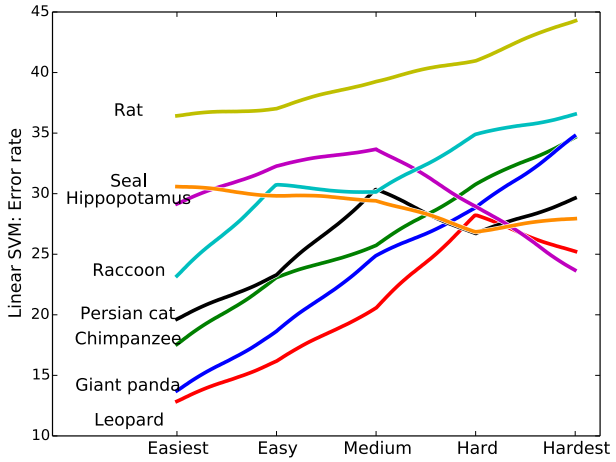

Figure 4. Visualization of machine learning performance (linear SVM) w.r.t. human annotation of easy-hard tasks for AwA dataset experiment. Ideally, when human and machine understanding coincide, it would be a bottom-up diagonal line. Therefore, in cases of *seal* and *hippopotamus* we would not expect that learning in semantic order would lead us to the best performing strategy, as in other classes. Best viewed in colors.

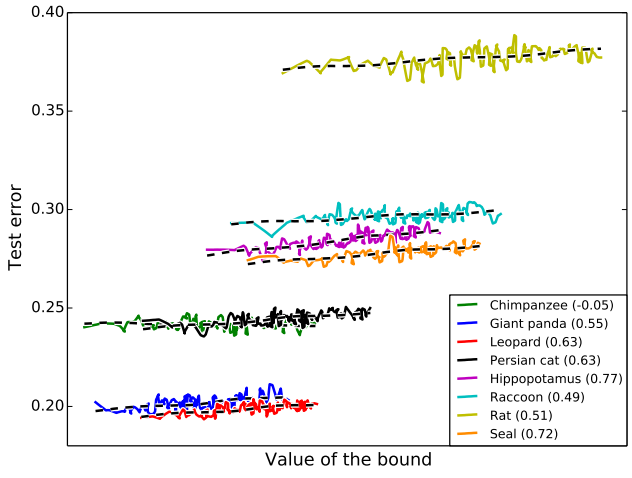

Figure 5. Visualization of the dependence between the test error and the value of the bound in AwA experiment. On every data split all possible task orders were sorted according to the corresponding values of the objective function (3) and their test errors were averaged over 20 repeats. Numbers in the brackets in the legend are correlation coefficients between the test error and the value of the bound. Black dashed lines illustrate the best linear fit. Best viewed in colors.

to (3). For this, for every data split we sort all possible task orders according to the corresponding values of (3) and compute their test error (averaged over 20 repeats) as well as the correlation coefficients between the error rate and the value of the bound. The results are shown in Figure 5. In all cases except for *chimpanzee* there is a positive correlation between ranking of the task order based on the theoretical analysis and its test performance, hence, providing the evidence of usefulness of the Theorem 1.

We also study the importance of the two terms in the objective function (4) for choosing the next task. For this, we compare our algorithm to two simplifications: choosing the next task based on the training error only (Error) and choosing the next task based on the complexity term only (Compl). The results in Table 1 suggest that the complexity term, i.e. the similarity between tasks, is the more important component, but that its combination with the error term achieves never worse and sometimes even better results.

To conclude, our proposed algorithm orders the tasks into a learning sequence to achieve the best performance results, and is beneficial to all other strategies including the order annotated for human learning.

4.2. Learning subsequences of related attributes

We focus on 10 attributes that describe shoe models [18]: *pointy at the front, open, bright in color, covered with ornaments, shiny, high at the heel, long on the leg, formal, sporty, feminine* and 10 classes from the *Shoes* dataset: *athletic, boots, clogs, flats, heels, pumps, rain boots, sneakers, stiletto, wedding shoes*. Attribute description comes in form of class ranking from 1 to 10, with 10 denoting class that "has it the most" and 1 denoting class that "has it the least". We form 10 binary classification tasks, one for each attribute, using samples from top-2 classes as positive (classes with 10 and 9 ranks) and samples from bottom-2 classes as negative (classes with 1 and 2 ranks). For more clarifications on attribute-class description, see the Supplementary material. For each task we balance 50 vs 50 training images and 300 vs 300 test images, randomly sampled from each class in equal amount. The data between different tasks does not overlap. As feature representation, we use 960 dimensional GIST descriptor concatenated with L_1-normalized 30 dimensional color descriptor, augmented with a unit element as bias term.

Baselines. In addition to all baselines described in the previous section, we add the MultiSeqMT method that allows to learn multiple subsequences of attributes (with

Methods	Average error
IndSVM	10.34 ± 0.13
MergedSVM	29.67 ± 0.10
MT	10.37 ± 0.13
SeqMT (ours)	10.96 ± 0.12
MultiSeqMT (ours)	$\mathbf{9.95 \pm 0.12}$
Diversity	12.66 ± 0.17
Random	12.14 ± 0.20
RandomMultiSeq	10.89 ± 0.14

Table 2. Learning subsequences of related attributes on Shoes dataset. We compare the proposed MultiSeqMT and SeqMT methods with the multi-task (MT) and the single-task (IndSVM) baselines, and report the MergedSVM result as a reference baseline. We examine the importance of subsequences in which the tasks are being solved and compare our methods with Diversity, Random and RandomMultiSeq baselines. The numbers correspond to average error rate performance over 10 tasks across 20 repeats (the lower the better). The best result is highlighted in **boldface**.

the information transfer within a subsequence). Additionally we include a baseline RandomMultiSeq that learns attributes in random order with an option to randomly start a new subsequence.

Results. We present the results of this experiment in Table 2. As we can see from it, the proposed MultiSeqMT method outperforms all other baselines and is a favourable strategy in this scenario. It is better than the SeqMT model which confirms that learning multiple subsequences is advantageous, when not all given tasks are equally related. The single-task learning baseline IndSVM is rather strong and performs on par with the multi-task learning MT baseline, possibly because multi-task learning is negatively affected by it forcing transfer between unrelated tasks. As expected, MergedSVM is unable to explain all tasks with one hyperplane and performs very poorly in this case.

Similarly to the previous experiment, we examine the importance of sequences and subsequences in which the tasks are being solved. First, we compare the performance of the MultiSeqMT and SeqMT methods with the baselines that learn tasks in certain order (last three rows in the Table 2), and then we will share our findings about the learned subsequences of attributes.

As we can see from Table 2, MultiSeqMT is able to order the tasks into subsequences in the most effective way. Learning multiple random subsequences as RandomMultiSeq does is better than learning a single sequence of all tasks, as SeqMT, Random and Diversity baselines do. However since SeqMT performs on par with RandomMultiSeq and clearly better than Random baseline, we conclude, that even with one sequence we are able to learn a good order of tasks that is discretely affected by transfer between unrelated tasks. The Diversity baseline is worse than other baselines also in this setting.

Finally, we analyze the subsequences that MultiSeqMT has learned. On average, there are 4.6 sequences, typically, the longest is 5-6 elements, then there are several pairs, and a few singletons. In particular, there are six attributes, *shiny, high at the heel, pointy at front, feminine, open* and *formal*, that can benefit from each other and often form a subsequence of related tasks. Inside the group, the attributes *shiny* and *high at the heel* frequently start the subsequence and transfer happens between both of them interchangeably. The next attributes that often follow the previous two are *pointy at front* and *feminine*; they are also closely related and interchangeable in order. The attribute *open* is not always in the subsequence, but once it is included, it transfers to *formal*, which often ends the subsequence.

The remaining four attributes, *bright in color, covered with ornaments, long on the leg* and *sporty*, either form smaller subsequences, sometimes of two tasks only, or they appear as separate tasks. Occasionally there is transfer from *long on the leg* attribute to *covered with ornaments*, which we credit to the fact the shoe class *boots* shares a high rank for both of those attributes. In half of the cases, the attributes *sporty* and *bright in color* are not related to the other tasks and form their own subsequences.

5. Conclusion

In this work, we proposed to solve multiple tasks in a sequential manner and studied the question if and how the order in which a learner solves a set of tasks influences its overall performance. First, we provided a theoretical result: a generalization bound that can be used to access the quality of the learning order. Secondly, we proposed a principled algorithm for choosing an advantageous order based on the theoretical result. Finally, we tested our algorithm on two datasets and showed that: 1) learning multiple tasks sequentially can be more effective than learning them jointly; 2) the order in which tasks are solved effects the overall classification performance; 3) our method is able to automatically discover a beneficial order.

A limitation of our model is that currently it allows to transfer only from the previous task to solve the current one, hence it outputs a sequence of related tasks or multiple task subsequences. In future work, we plan to extend our model by relaxing this condition and allowing the tasks to be organized in a tree, or a more general graph structure.

Acknowledgments We thank Novi Quadrianto for helpful discussions. This work was in parts funded by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement no 308036.

References

[1] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 2008. 2

[2] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *International Conference on Computer Vision (ICCV)*, 2011. 1

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding (CVIU)*, 2008. 5

[4] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *International Conference on Machine Learing (ICML)*, 2009. 2

[5] T. L. Berg, A. C. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *European Conference on Computer Vision (ECCV)*, 2010. 4

[6] R. Caruana. Multitask learning. *Machine Learning*, 1997. 1

[7] X. Chen, S. Kim, Q. Lin, J. G. Carbonell, and E. P. Xing. Graph-structured multi-task regression and an efficient optimization method for general fused lasso. arXiv:1005.3579 [stat.ML], 2010. 2

[8] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research (JMLR)*, 6, 2005. 2

[9] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2004. 1, 2, 5

[10] J. R. Finkel and C. D. Manning. Hierarchical Bayesian domain adaptation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2009. 5

[11] T. Gao, M. Stark, and D. Koller. What makes a good detector? – structured priors for learning from few examples. In *European Conference on Computer Vision (ECCV)*, 2012. 1

[12] P. Germain, A. Lacasse, F. Laviolette, and M. Marchand. PAC-Bayesian learning of linear classifiers. In *International Conference on Machine Learing (ICML)*, 2009. 3

[13] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *European Conference on Computer Vision (ECCV)*, 2012. 1

[14] J. L. Hintze and R. D. Nelson. Violin plots: A box plot-density trace synergism. *The American Statistician*, 1998. 6

[15] D. Jayaraman, F. Sha, and K. Grauman. Decorrelating Semantic Visual Attributes by Resisting the Urge to Share. In *Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[16] W. Kienzle and K. Chellapilla. Personalized handwriting recognition via biased regularization. In *International Conference on Machine Learing (ICML)*, 2006. 3

[17] A. Kovashka and K. Grauman. Attribute adaptation for personalized image search. In *International Conference on Computer Vision (ICCV)*, 2013. 1

[18] A. Kovashka, D. Parikh, and K. Grauman. Whittlesearch: Image Search with Relative Attribute Feedback. In *Computer Vision and Pattern Recognition (CVPR)*, 2012. 4, 7

[19] A. Kumar and H. Daumé III. Learning task grouping and overlap in multi-task learning. In *International Conference on Machine Learing (ICML)*, 2012. 2

[20] M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *Conference on Neural Information Processing Systems (NIPS)*, 2010. 2

[21] A. Lad, Y. Yang, R. Ghani, and B. Kisiel. Toward optimal ordering of prediction tasks. In *SIAM International Conference on Data Mining (SDM)*, 2009. 2

[22] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2013. 4

[23] J. Langford and J. Shawe-Taylor. PAC-Bayes and margins. In *Conference on Neural Information Processing Systems (NIPS)*, 2002. 3

[24] D. A. McAllester. Some PAC-Bayesian theorems. *Machine Learning*, 1999. 2

[25] F. Orabona, C. Castellini, B. Caputo, A. E. Fiorilla, and G. Sandini. Model adaptation with least-squares SVM for adaptive hand prosthetics. In *International Conference on Robotics and Automation (ICRA)*, 2009. 1

[26] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 2010. 1

[27] A. Pentina and C. H. Lampert. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learing (ICML)*, 2014. 2

[28] A. Pentina, V. Sharmanska, and C. H. Lampert. Curriculum learning of multiple tasks. arXiv:1412.1353 [stat.ML], 2014. 4, 5

[29] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, 2009. 2

[30] P. Ruvolo and E. Eaton. Active Task Selection for Lifelong Machine Learning. In *Conference on Artificial Intelligence (AAAI)*, 2013. 2, 5

[31] P. Ruvolo and E. Eaton. ELLA: An efficient lifelong learning algorithm. In *International Conference on Machine Learing (ICML)*, 2013. 2

[32] V. Sharmanska, N. Quadrianto, and C. Lampert. Learning to transfer privileged information. arXiv:1410.0389 [cs.CV], 2014. 5

[33] T. Tommasi and B. Caputo. The more you know, the less you learn: from knowledge transfer to one-shot learning of object categories. In *British Machine Vision Conference (BMVC)*, 2009. 1

[34] T. Tommasi, F. Orabona, and B. Caputo. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *Computer Vision and Pattern Recognition (CVPR)*, 2010. 1

[35] J. Yang, R. Yan, and A. G. Hauptmann. Cross-domain video concept detection using adaptive SVMs. In *International Conference on Multimedia (ICM)*, 2007. 1

[36] J. Yang, R. Yan, and A. G. Hauptmann. Cross-domain video concept detection using adaptive svms. In *International Conference on Multimedia (MM)*, 2007. 3