Adaptive Curriculum Learning

Yajing Kong[1], Liu Liu[1], Jun Wang[2], Dacheng Tao[3,1],
[1]The University of Sydney, Australia, [2]City University of Hong Kong, China, [3]JD Explore Academy, China
ykon9947@uni.sydney.edu.au, liuliu1@sydney.edu.au, jwang.cs@cityu.edu.hk, dacheng.tao@gmail.com

Abstract

Inspired by the human learning principle that learning easier concepts first and then gradually paying more attention to harder ones, curriculum learning uses the non-uniform sampling of mini-batches according to the order of examples' difficulty. Just as a teacher adjusts the curriculum according to the learning progress of each student, a proper curriculum should be adapted to the current state of the model. Therefore, in contrast to recent works using a fixed curriculum, we devise a new curriculum learning method, Adaptive Curriculum Learning (Adaptive CL), adapting the difficulty of examples to the current state of the model. Specifically, we make use of the loss of the current model to adjust the difficulty score while retaining previous useful learned knowledge by KL divergence. Moreover, under a non-linear model and binary classification, we theoretically prove that the expected convergence rate of curriculum learning monotonically decreases with respect to the loss of a point regarding the optimal hypothesis, and monotonically increases with respect to the loss of a point regarding the current hypothesis. The analyses indicate that Adaptive CL could improve the convergence properties during the early stages of learning. Extensive experimental results demonstrate the superiority of the proposed approach over existing competitive curriculum learning methods.

1. Introduction

In human education, a teacher arranges the learning materials in the order of increasing difficulty, such that students can learn more complex concepts faster after gaining sufficient basic and easy knowledge [8,24]. Curriculum learning (CL), inspired by the teaching strategy, learns by starting with easy examples and then gradually putting more weight on harder ones [2].

Learning with a proper curriculum can benefit the generalizability and convergence [2,46]. However, a good curriculum is not always easy to develop, as it must address the key question of how to measure the difficulty of each example. Various methods have tried to address the question. For example, self-paced learning (SPL) [19] favored training examples with smaller current losses and selected the easy examples by solving a biconvex optimization problem. Different from SPL, [11,27,46] obtained the difficulty score, i.e., the measurement of difficulty, by transfer learning or bootstrapping before training, and used the score to determine a fixed curriculum. However, using a fixed curriculum ignores the fact that a proper curriculum should be adjusted according to the learning progress of every example, just as students learn better if the teacher can adjust the curriculum according to the progress of every student.

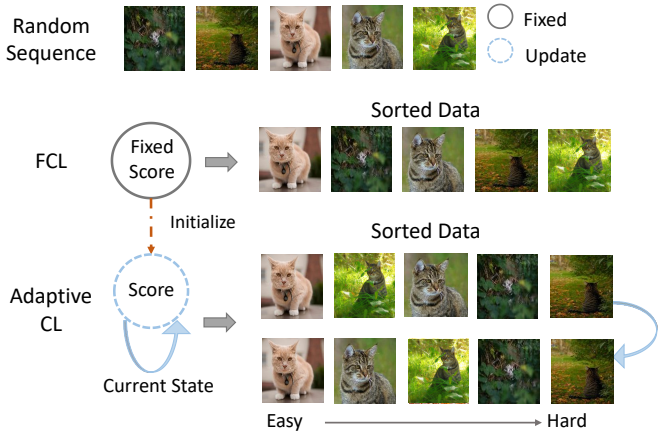


Figure 1. Comparison of fixed curriculum learning (FCL) [11] and our method (Adaptive CL). Different from FCL that uses a fixed difficulty score to determine the order of data, our method uses an adaptive difficulty score that is adjusted by the current state of the model during training. The bottom line shows sorted data that are gradually updated based on the Adaptive CL algorithm.

Our work is inspired by the excellent work of [11], but does not depend on a fixed curriculum. We propose a simple but effective curriculum learning approach that takes the feedback of the current state of the model into account. As shown in Figure 1, the proposed method can gradually adapt the order of examples from easy to hard. In particular, we use the difficulty score to measure the difficulty of examples and regard the examples with lower scores as easier examples. We first obtain the initial (pseudo-ideal) difficulty

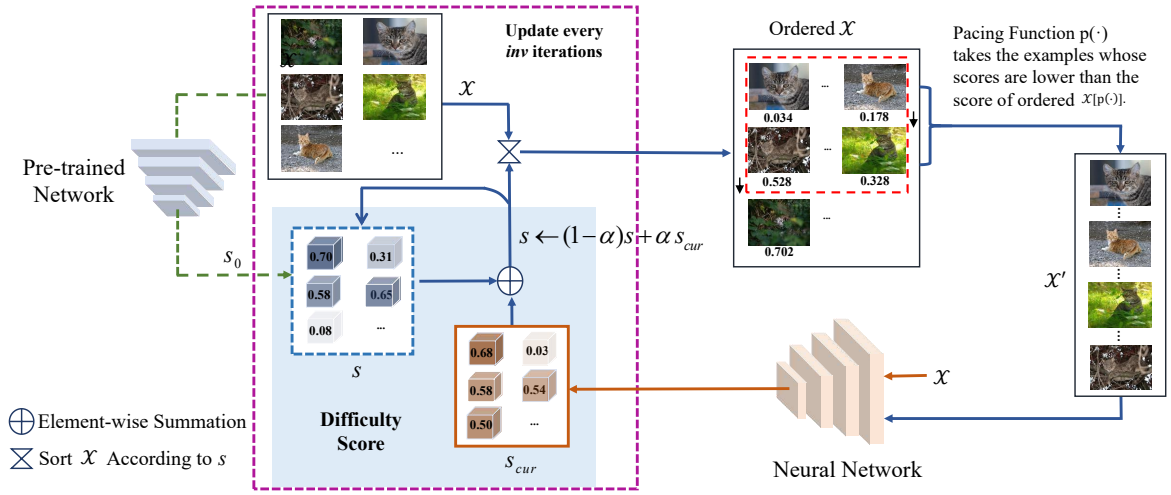


Figure 2. The pipeline of our method. First, we obtain the initial difficulty score s_0 from a pre-trained network. During training, we sort the dataset $\mathcal{X}$ in ascending order according to the difficulty score s, which is adjusted by the current difficulty score s_{cur} using α every inv iterations. With the ordered dataset and pacing function $p(\cdot)$, we can obtain the sample pool $\mathcal{X}'$. The network then trains on the mini-batch sampled randomly from the pool and generates a new current difficulty score. We iterate the procedure until the model converges.

score through a pre-trained network and then gradually adjust the score using the current losses of the examples. During this procedure, to prevent the model from placing too much attention on the current stage of the network or not fully utilizing the knowledge learned from easy examples, we introduce a parameter α to adjust the adaptation. Moreover, considering that people generally learn concepts more quickly if they can make use of knowledge learned from former similar tasks, we use the Kullback-Leibler (KL) divergence between the output of the pre-trained model and the current model to prevent the network from completely forgetting the previous useful learned knowledge. Furthermore, it is also important to control the pace of presenting materials from easy to hard in human education proceeding, as too fast would make the students confused while too slow would make learning boring. Therefore, we use a pacing function to control the growth speed of the average score of mini-batch. The full procedure of the proposed method is shown in Figure 2.

In curriculum learning, theoretical analysis is only discussed for convex problems [45, 46], e.g., linear regression. For non-convex problems, the challenge of CL is not well understood, although curriculum learning has been applied to non-linear models for decades. We, therefore, provide a theoretical analysis of curriculum learning under a non-linear model in the task of binary classification. For a data point, we refer to the loss regarding the optimal hypothesis as the ideal difficulty score and the loss regarding the current hypothesis as the current difficulty score. Under mild assumptions, we prove that during the early stages of learning, the expected convergence rate of curriculum learning monotonically decreases with respect to the ideal difficulty score, and given the ideal difficulty score the expected convergence rate monotonically increases with respect to the current difficulty score. In view that our adaptive difficulty score is a weighted sum of positive pseudo-ideal difficulty score and negative current difficulty score, the theoretical results indicate that Adaptive CL could improve the convergence rate in the early stages of learning, since we train the examples with lower scores in the early stages of learning. To summarize, our contributions are threefold:

- We propose a new curriculum learning approach, Adaptive CL, that adapts the difficulty score to the current state of the network while remembering the useful knowledge learned from the pre-trained model.
- We theoretically analyze the relationship between the expected convergence rate and the difficulty scores in curriculum learning under a simple non-linear model for binary classification. The theoretical analyses indicate that the proposed method could improve the convergence rate in the early stages of learning.
- We compare the proposed algorithm with existing competitive methods on several benchmarks and networks. Extensive empirical results demonstrate the superiority of the proposed method.

2. Related Work

In human education or animal training, curricula are commonly used to facilitate learning [18, 28, 32, 38]. Inspired by the learning principles, [2] first proposed curriculum learning in the context of machine learning. The work of [2] has sparked considerable interest in applying

curriculum learning to different research fields including computer vision [1, 9, 23, 30, 33, 36]. For example, [36] proposed a transferable curriculum learning approach that could inform the network which of the source examples were noiseless and transferable for domain adaptation. [7] proposed Curriculum DeepSDF, a "shape curriculum" for learning continuous signed distance function on shapes. [42] proposed a curriculum for imbalanced data classification and learned examples not only from imbalanced to balanced but also from easy to hard. [27] proposed a curriculum for conversation response ranking and explored the effect of different difficulty scores. Moreover, curriculum learning is also prevalent in speech recognition [3, 21, 31], natural language [29, 33, 47], and reinforcement learning [10, 13, 22, 25, 26, 39, 40].

Following, we introduce the literary works that are mostly related to our work. [11, 27] used transfer learning or bootstrapping to obtain the difficulty score before training and used the score to determine a fixed curriculum. [19] proposed self-paced learning, which favors examples with small losses during training. Self-paced curriculum learning (SPCL) [15] combined predetermined curriculum learning and self-paced learning. Our approach is similar to SPCL while our method uses a different scheme to adjust the curriculum. Focal loss [20] proposed a new loss function that the weight of examples are reflected on the loss function. [34] introduced a new "data parameter" to the softmax function to learn a curriculum. Similarly, [14] proposed CurricularFace, a dynamic curriculum learning for deep face recognition by adding a modulation coefficient function to the softmax-based loss function. [14, 20, 34] realized adaptive curriculum learning by reshaping the loss function. Instead of changing the loss function, we use a specifically-defined adaptive difficulty score that could adapt the curriculum to the current learning progress. MentorNet [16] is a data-driven curriculum learning approach on corrupted labels that also takes the feedback of the current network. However, the method requires training a MentorNet to adjust the curriculum during training, which would increase the cost of computing.

There are few theoretical analyses in curriculum learning. [11] proved that under mild conditions, curriculum learning would not change the corresponding global minimum of the objective function while modifying the optimization landscape. [44, 46] showed that the expected convergence rate monotonically decreased with the loss of a point regarding the optimal hypothesis and monotonically increased with the loss of a point regarding the current hypothesis in convex problems, e.g., linear regression with the least-squares loss. However, the analysis of curriculum learning in non-convex problems is absent. Therefore, from the viewpoint of non-linear model and binary classification, we prove that curriculum learning also has an effect on non-convex problems.

3. Adaptive curriculum learning

In this section, we introduce the proposed method, Adaptive Curriculum Learning (Adaptive CL). We first introduce the adaptive difficulty score that takes the feedback of the current state of the network into account. Then we leverage KL divergence to ensure that the model learns from the useful knowledge learned from the pre-trained model. Moreover, to control the pace of presenting data from easy to hard, we use the pacing function to control the growth speed of the average difficulty score of mini-batch.

3.1. Notations and definitions

Let $\mathcal{X} = \{(x_i, y_i)\}_{i=1}^{n}$ be the training dataset, where x_i is the i-th data point, y_i is its corresponding label, and n is the total number of data. We use $i \in [n]$ to denote that i is generated from $[n] = \{1, 2, ..., n\}$. Let $\mathcal{N}(\theta)$ be the target model with parameter θ. We apply stochastic gradient descent (SGD) to optimize the model. SGD usually trains data in a sequence of mini-batches $\mathcal{B} = [\mathcal{B}_1, ..., \mathcal{B}_M]$, where M is the total batch number in training. Those mini-batches are randomly sampled from the full dataset. However, in curriculum learning, the mini-batches will be arranged in the order of increasing average difficulty, and we denote the new sequential mini-batches as $\mathcal{B}' = [\mathcal{B}'_1, ..., \mathcal{B}'_M]$. The order is determined by the difficulty score s. The pace of presenting examples from easy to hard is controlled by the pacing function $p(\cdot)$.

3.2. The proposed algorithm

In this subsection, we introduce the proposed method, Adaptive CL. The pipeline is shown in Figure 2. Adaptive CL trains the target model with sequential min-batches $\mathcal{B}' = [\mathcal{B}'_1, ..., \mathcal{B}'_M]$, which are arranged in the order of increasing average difficulty score. We use the pre-trained model to obtain the initial difficulty score s_0 and then gradually adjust the difficulty score using the current losses of examples. With the difficulty score, we sort the full dataset $\mathcal{X}$ in ascending order, and obtain a sample pool $\mathcal{X}'$ which contains the first $p(\cdot)$ easier examples. Then the next mini-batch will be randomly selected from the sample pool $\mathcal{X}'$. In our method, we use pacing functions $p(\cdot)$ that increase with iterations, so the average difficulty score of mini-batch also tends to increase with iterations, meaning that increasing numbers of "hard" examples are added to the sample pool $\mathcal{X}'$. Moreover, sampling data from the sample pool $\mathcal{X}'$ may cause bias, since the pool may contain a different number of examples in each class. To avoid such bias, like [11] we keep the samples balanced by forcing the sample pool $\mathcal{X}'$ to contain the same number of examples from each class as in the full dataset. During training, in order to retain the knowledge learned from the pre-trained model, we add KL

Algorithm 1 Adaptive Curriculum Learning

Input: Pacing function p, Initial difficulty score s_0, Data $\mathcal{X}$, Target network $\mathcal{N}(\theta)$, Interval *inv*, Parameters λ and α, Pre-trained model $\mathcal{M}(\theta')$.
Output: Target network $\mathcal{N}$

```
Initialize s to s_0
sort X according to s in ascending order
for m = 1, ..., M do
    size ← p(m)
    X_m' ← X[1, ..., size]
    uniformly sample B'_m from X'_m
    Train N on B'_m over the objective function (2)
    if mod(m, inv) = 0 then
        s_cur ← N(θ)
        s ← (1 − α(m))s + α(m)s_cur
        sort X according to s in ascending order
    end if
end for
return N
```

divergence to the objective function. The pseudo-code of the proposed algorithm is given in Algorithm 1. In the following subsections, we will introduce the difficulty score, objective function, and pacing function in detail.

3.3. Difficulty score

The difficulty score plays a key role in curriculum learning as it measures the difficulty of each example and determines the order of training examples presented to the model. [46] used the minimal loss regarding the optimal hypothesis as an ideal difficulty score. However, when a teacher does not know the optimal curriculum in advance, he or she will first arrange the curriculum according to his or her previous teaching experience and then gradually adjust the curriculum according to the learning progress of every student. Inspired by this teaching strategy, we first regard the difficulty score obtained from the pre-trained model as an initial (pseudo-ideal) difficulty score and then adapt the score to the current state of the model.

We refer to the loss obtained from the current stage of the network as the current difficulty score s_{cur} and update the difficulty score in a certain interval, use inv for short. Then, the k+1-th difficulty score can be represented as:

$$s^{k+1} \leftarrow (1-\alpha)s^k + \alpha s^k_{cur}, \tag{1}$$

where $k = \lfloor m/inv \rfloor$, m indicates the m-th mini-batch, α is an adjustment parameter and s is initialized to s_0, the difficulty score obtained from a pre-trained model.

It should be emphasized that α should be negative. The examples with lower current difficulty scores fit the current model well and provide less information. So by decreasing a small portion of the current difficulty score, the model will pay less attention to those examples that contain less information. In section 4, we provide a theoretical analysis under a non-linear model to validate that such a selection is reasonable. In section 5, we also compare the performance of using positive α and negative α, respectively, to validate that a negative α is better.

3.4. Objective function

KL divergence is frequently used to maintain useful knowledge learned from a pre-trained model [4, 5, 17, 43]. Therefore, for an objective function, in addition to cross-entropy $\mathcal{L}_{XE}$, we also use the KL divergence $\mathcal{L}_{\mathcal{KL}}$ between the output of the pre-trained model $\mathcal{M}$ and the current model $\mathcal{N}$ to retain useful learned knowledge:

$$\mathcal{L} = \mathcal{L}_{XE} + \lambda \mathcal{L}_{KL}, \tag{2}$$

where $\mathcal{L}_{XE} = -\frac{1}{|\mathcal{B}'|}\sum_{j=1}^{|\mathcal{B}'|} y_j log\mathcal{N}(x_j,\theta)$, $\mathcal{L}_{KL} = -\frac{1}{|\mathcal{B}'|}\sum_{j=1}^{|\mathcal{B}'|} \mathcal{M}(x_j,\theta')log\mathcal{N}(x_j,\theta)$[1], $\{x_j, y_j\}$ is a data point of the current mini-batch $\mathcal{B}'$, $\mathcal{M}(x_j,\theta')$ is the pre-trained model with parameter θ', λ is a balancing parameter that controls how much the knowledge learned from the pre-trained model should be retained.

3.5. Pacing function

Simply obtaining the difficulty score of each example is insufficient to establish a proper curriculum, since it is also important to control the pace of curriculum learning, i.e., the growth speed of the average difficulty score of mini-batch. Like in human education, if a teacher presents materials from easy to hard in a very short period of time, students would become confused and not learn effectively. However, if the teacher presents materials too slowly, students will become bored and lose interest. To control the learning pace, we use the pacing function $p(\cdot) : [M]$[2] $\rightarrow [n]$ to determine the size of the sample pool $\mathcal{X}'$, which is a subdataset containing the first $p(\cdot)$ easier examples [11, 27]. In the training procedure, the m-th mini-batch $\mathcal{B}'_m$ is sampled randomly from $\mathcal{X}'_m$. The average difficulty score of the m-th mini-batch s_m is:

$$s_m = \frac{1}{p(m)}\sum_{j=1}^{p(m)} s_{m,j},$$

where $s_{m,j}$ is the difficulty score of the j-th example in the m-th sample pool $\mathcal{X}'_m$. We use exponential pacing functions, where the datasize of $\mathcal{X}'_i$ increases exponentially in each step. Step denotes all batch number during which $p(\cdot)$ remains constant [11]. More details including the formula, figure illustration, and comparison of different pacing functions can be referred to the appendix.

[1] $KL(\mathcal{M}||\mathcal{N}) = \mathcal{M}(\theta')log\mathcal{M}(\theta') - \mathcal{M}(\theta')log\mathcal{N}(\theta)$. We ignore the first term since θ' is the parameter of the pre-trained model and would not have any impact on the optimization.

[2] For an integer, [M] denotes {1, 2, ..., M}

4. Theoretical Analysis

In this section, we first analyze the relationship between the expected convergence rate and the ideal difficulty score, and then we analyze the relationship between the expected convergence rate and the current difficulty score. Unlike previous works that analyzed curriculum learning for convex problems on linear regression and binary classification with hinge loss [45, 46], we analyze curriculum learning with a non-linear model for binary classification with the least squares loss. Non-linear models bring challenges in analyzing the convergence of curriculum learning, including 1) several critical properties, e.g., symmetry, are not available anymore, and 2) we need to consider the effect of the non-convex activation function. Under mild assumptions, we prove that the expected convergence rate monotonically decreases with the ideal difficulty score, and given the ideal difficulty score, the expected convergence rate monotonically increases with the current difficulty score during the early stages of learning. The analytical results indicate that Adaptive CL could improve the convergence rate in the early stages of learning.

4.1. Definitions and notations

Let $\mathcal{X} = \{(x_i, y_i)\}_{i=1}^n$ be the training data, where $x_i \in \mathbb{R}^d$ is the i-th data point and y_i is its corresponding label. The ideal difficulty of a point x is measured by its minimal loss regarding the optimal hypothesis. The current difficulty of a point x is measured by its loss regarding the current hypothesis.

We focus on the differential effect of a data point's difficulty score on convergence towards the global solution under a simple non-linear model in the task of binary classification. Let ψ be the ideal difficulty score and we define the ideal difficulty score as $\psi = |g(\bar{\mathbf{w}}^\top \boldsymbol{x}_i) - y_i|$, where $g(\cdot)$ is the activation functions, $\boldsymbol{x}_i = \begin{bmatrix} x_i^\top & 1 \end{bmatrix}^\top$ [3], and $\bar{\mathbf{w}}$ is the global solution of the empirical loss. In an analogous way, let γ be the current difficulty score and we define the current difficulty score at time t as $\gamma = |g\left(\mathbf{w}_t^\top \mathbf{x}_i\right) - y_i|$, where $\mathbf{w}_t$ is the model parameter at time t.

4.2. Convergence rate with an ideal difficulty score

In this subsection, we prove that under some mild assumptions, the expected convergence rate monotonically decreases with respect to the ideal difficulty score in the early stages of learning.

Curriculum Learning trains the model on a sequence of training points $X_t = \{\boldsymbol{x}_{i_t}, y_{i_t}\}, t \in [T]$, sampled from the training dataset. As we analyze a non-linear model with the least squares loss, the loss function can be represented as $L(X_t, \mathbf{w}) = (g(\mathbf{w}^\top \boldsymbol{x}_{i_t}) - y_{i_t})^2$. Following the stochastic gradient descent rule, the update step at time t:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \frac{\partial L(X_t, \mathbf{w})}{\partial \mathbf{w}}|_{\mathbf{w}=\mathbf{w}_t}, \tag{3}$$

where η is the learning rate.

Let $\Delta(\psi)$ be the expected convergence rate at time t. Given difficulty score ψ, $\Delta(\psi)$ could be defined as

$$\Delta(\psi) = \mathbb{E}[\|\mathbf{w}_t - \bar{\mathbf{w}}\|^2 - \|\mathbf{w}_{t+1} - \bar{\mathbf{w}}\|^2 | \psi].$$

Theorem 1. *Let $h(\boldsymbol{w}) = g\left(\boldsymbol{w}^\top \boldsymbol{x}\right)$ be a non-linear model whose label belongs to $\{-1, 1\}$, $\{(\boldsymbol{x}_{i_t}, y_{i_t})\}$ is the sampled data point at time t, where $i_t \in [n]$ and $t \in [T]$. $g(\cdot)$ is the tanh activation function: $g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$. Let $\gamma = |g(\boldsymbol{w}_t^\top \boldsymbol{x}_{i_t}) - y_{i_t}|$ be the current difficulty score and $\psi = |g(\bar{\boldsymbol{w}}^\top \boldsymbol{x}_{i_t}) - y_{i_t}|$ be the ideal difficulty score. $\boldsymbol{w}_t$ is the model parameter at time t, and $\bar{\boldsymbol{w}}$ is the optimal model parameter. In the early stages of learning, the expected convergence rate at time t monotonically decreases with respect to the ideal difficulty score ψ of $\boldsymbol{x}_{i_t}$, i.e., $\frac{\partial \Delta(\psi)}{\partial \psi} \leq 0$, under the assumptions that*

1) the current example, i.e., the "easy" example, could be correctly labeled;

2) when $y_{i_t} = -1$, $g(\boldsymbol{w}_t^\top \boldsymbol{x}_{i_t}) - g(\bar{\boldsymbol{w}}^\top \boldsymbol{x}_{i_t}) \geq 0.2$;

3) when $y_{i_t} = 1$, $g(\boldsymbol{w}_t^\top \boldsymbol{x}_{i_t}) - g(\bar{\boldsymbol{w}}^\top \boldsymbol{x}_{i_t}) \leq -0.2$.

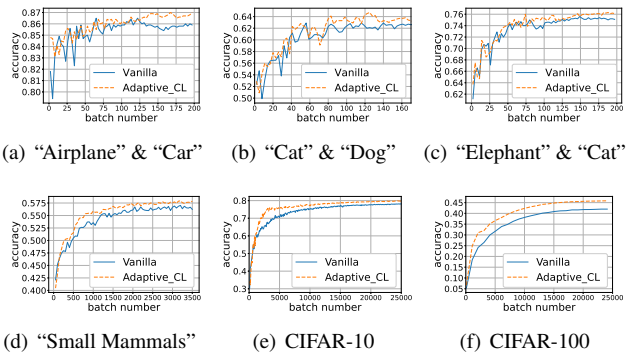
We leave the proof in the appendix. The assumptions of **Theorem 1** are mild during the early stages of learning. For the first assumption, curriculum learning begins the training on the "*easy*" examples, indicating that the current examples could be correctly classified by the model in the early stages. The second and third assumptions are also easy to realize. For example, when $y_{i_t} = -1$, it is reasonable to assume that $g(\bar{\mathbf{w}}^\top \boldsymbol{x}_{\boldsymbol{i_t}})$ is near to -1. During the early stages of training, the prediction of data may be far away from the true label, i.e., $g(\mathbf{w}_t^\top \boldsymbol{x}_{i_t}) - g(\bar{\mathbf{w}}^\top \boldsymbol{x}_{i_t}) \geq 0.2$. Therefore, the second assumption that when $y_{i_t} = -1$, $g(\mathbf{w}_t^\top \boldsymbol{x}_{i_t}) - g(\bar{\mathbf{w}}^\top \boldsymbol{x}_{i_t}) \geq 0.2$ could be satisfied. The situation of $y_{i_t} = 1$ is similar to the case of $y_{i_t} = -1$. Therefore, under mild assumptions, we could conclude that the expected convergence rate monotonically decreases with respect to the ideal difficulty score during the early stages of learning.

4.3. Convergence rate with a current difficult score

In this subsection, we prove that if the gradient step η is small enough and given the ideal difficulty score, the expected convergence rate monotonically increases with respect to the current difficulty score.

Given the difficulty scores ψ and γ, we define $\Delta(\psi, \gamma) = \mathbb{E}[\|\mathbf{w}_t - \bar{\mathbf{w}}\|^2 - \|\mathbf{w}_{t+1} - \bar{\mathbf{w}}\|^2 | \psi, \gamma]$ as the expected convergence rate at time t.

[3] $\boldsymbol{w}^\top \boldsymbol{x}_i = \begin{bmatrix} w^\top & b \end{bmatrix}^\top$, where $\boldsymbol{w} = \begin{bmatrix} w^\top & b \end{bmatrix}^\top$, Therefore, $\boldsymbol{x}_i = \begin{bmatrix} x_i^\top & 1 \end{bmatrix}^\top$.



(a) "Airplane" & "Car" (b) "Cat" & "Dog" (c) "Elephant" & "Cat"

(d) "Small Mammals" (e) CIFAR-10 (f) CIFAR-100

Figure 3. Comparison of Adaptive CL (without KL divergence) and Vanilla. Top: Binary Classification & MLP. Bottom: Multi-class Classification & HNN. The results show that our method gets higher accuracy faster, agreeing with the theoretical analysis that Adaptive CL could have a faster convergence rate in the early stages of learning.

Theorem 2. *Let* $h(\boldsymbol{w}) = g\left(\boldsymbol{w}^\top \boldsymbol{x}\right)$ *be a non-linear model whose label belongs to* $\{-1, 1\}$, $\{(\boldsymbol{x}_{i_t}, y_{i_t})\}$ *is the sampled data point at time* t*, where* $i_t \in [n]$ *and* $t \in [T]$. $g(\cdot)$ *is a tanh activation function:* $g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$. *Let* $\gamma = |g(\boldsymbol{w}_t^\top \boldsymbol{x}_{i_t}) - y_{i_t}|$ *be the current difficulty score and* $\psi = |g(\bar{\boldsymbol{w}}^\top \boldsymbol{x}_{i_t}) - y_{i_t}|$ *be the ideal difficulty score.* $\boldsymbol{w}_t$ *is the model parameter at time* t*, and* $\bar{\boldsymbol{w}}$ *is the optimal model parameter. Assume that the gradient step* η *is small enough. Given the ideal difficulty score* ψ*, the expected convergence rate at time* t *monotonically increases with respect to the current difficulty score of* $\boldsymbol{x}_{i_t}$*, i.e.,* $\frac{\partial \Delta(\psi,\gamma)}{\partial \gamma} \geq 0$.

We leave the proof in the Appendix. It is interesting that fixed the ideal difficulty score, the expected convergence rate monotonically increases with respect to the current difficulty score, which is contrary to the ideal difficulty score. However, we claim that the results coincide with intuitions. Following a curriculum that is based on extrinsic (ideal) difficulty, a learner should not "wasting time" on examples that are easy for the current model (hypothesis). Specifically, during training, easier examples will be trained at first, and the current difficulty score of those examples will be lower more. However, those examples that the model fits well contain less information for the model. Hence, the learner should not "wasting time" on those examples and should pay a little more attention to examples with higher current difficulty scores while not violate the ideal difficulty score significantly.

According to **Theorems 1** and **2**, the expected convergence rate monotonically decreases with respect to the ideal difficulty score and monotonically increases with respect to the current difficulty score in the early stages of learning. The results indicate that the expected convergence rate monotonically decreases with respect to our adaptive difficulty score in the early stages of learning, since the adaptive score uses a weighted sum of a positive pseudo-ideal difficulty score and negative current difficulty score. Therefore, Adaptive CL could converge faster in the early stages of learning because it regards the examples with lower difficulty scores as "easy" examples and trains "easy" examples first.

5. Experiments

In this section, we first demonstrate the effectiveness of Adaptive CL. Then we compare the proposed method to existing competitive methods on different benchmarks. We also explore the effect of three components in Adaptive CL including adaptation, interval, and the initial difficulty scores.

5.1. Datasets, architectures, and difficulty score

Datasets We use five kinds of datasets: two-class datasets [6], small ImageNet [6], CIFAR-10 [18], CIFAR-100 [18], and the superclasses of CIFAR-100. The two-class datasets are small datasets containing two classes, each of which has 250 training examples and 50 test examples. The small ImageNet is a subset of five randomly selected classes from ImageNet datasets (see Appendix). CIFAR-10 and CIFAR-100, arranged in 10 classes and 100 classes respectively, contain 50,000 training images and 10,000 validation images, and consist of 32×32 color images. A superclass of CIFAR-100 is a subset of 3000 images from CIFAR-100 and contains five fine classes. In this paper, we use the following superclasses: "Aquatic Mammals (AM)", "Small Mammals (SM)", "Flowers", "Medium-sized (MS) mammals", and "Household Electrical (HE) devices".

Architectures For the architectures, we use five networks: (a) MLP network with a hidden layer whose hidden size is 20. (b) hand-crafted neural network (HNN) containing 8 convolutional layers as [11]; (c) VGG-16 network [37]; (d) ResNet-v1-14 [12]; (e) ResNet-18 [12].

Difficulty score We illustrate how we obtain the difficulty score. For the initial difficulty score, we extract the image features with 2048-dimensions from the penultimate layer's activation states of an inception network [41], which is pre-trained on the ImageNet. With those features, we then train a classifier and use the corresponding confidence score to determine the initial difficulty score of each example. In the experiment, we use the Radial Basis Kernel SVM [35] as the classifier. During training, we use the confidence score of the current network to determine examples' current difficulty scores and then adjust the adaptive difficulty score according to Eq.(1).

5.2. The effectiveness of Adaptive CL

In Figure 3, we plot the average validation accuracy curve of Vanilla (sampling mini-batches randomly from the

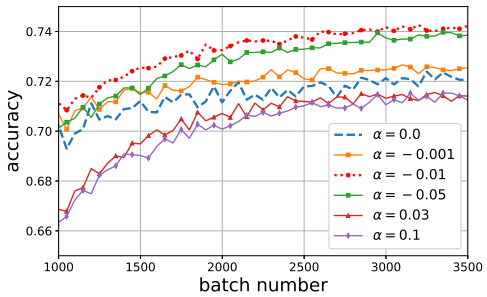


Figure 4. The average validation accuracy curve of Adaptive CL (without KL divergence) with different values of α.

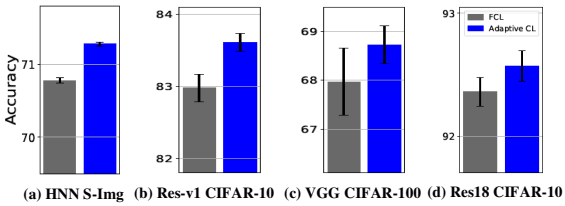


Figure 5. Comparison of FCL and Adaptive CL on different datasets and networks. The results are average validation accuracy and STE (STandard Error). S-Img denotes the small ImageNet, HNN denotes the hand-crafted network, Res-v1 denotes ResNet-v1-14 and Res18 denotes ResNet18.

full dataset) and the proposed method on various settings: Binary classification and MLP network (10 runs) and multi-class classification and HNN (6 runs). To eliminate the influence of knowledge distillation, we set $\lambda = 0$. The results show that Adaptive CL reaches higher accuracy faster than Vanilla at the beginning of learning, agreeing with the analysis that Adaptive CL could have a faster convergence during the early stages of learning.

Figure 4 shows the results of experiments trained on the hand-craft network and HE devices dataset with different values of α (25 runs). The results show that adaptive CL (without KL divergence) with negative α can improve the performance while the positive α impairs performance, which is corresponding to our analysis that α needs to be negative. In addition, when α is negative, $\alpha = -0.01$ reaches best performance. α with too large or too small magnitude will degenerate the performance. If the magnitude of α is too large, the model can not fully utilize the knowledge learned from easy examples and the pre-trained model. However, if the magnitude of α is too small, the model will adapt the difficulty score to the current state of the current model too slowly. Moreover, in Figure 4, negative α outperforms the case of $\alpha = 0.0$ by about 2%, demonstrating that adaptation CL can improve the performance than the curriculum without any adaptation, i.e., fixed curriculum learning (FCL).

Figure 5 shows more comparison results between the proposed method and FCL on different datasets and networks: (a) HNN & small ImageNet; (b) ResNet-v1-14 & CIFAR-10; (c) VGG & CIFAR-100; (d) ResNet18 & CIFAR-10. The repetition of experiment is 25 in (a), 5 in (b), and 3 in (c) and (d). The results in Figure 5 show that Adaptive CL outperforms FCL on different datasets and networks, further validating the effectiveness of Adaptive CL.

5.3. Comparison of Adaptive CL and baselines

Adaptive CL is compared to the following methods: 1) Vanilla samples the mini-batches randomly from the full dataset. 2) Fixed curriculum learning (FCL) [11] trains examples from easy to hard according to the difficulty score obtained by transfer learning, i.e., $\alpha = 0$. 3) Self-paced learning (SPL) [19] favors training samples with smaller current losses. 4) Self-paced curriculum learning (SPCL) [15] combines predetermined curriculum learning and self-paced learning. 5) Focal loss [20], a well-known curriculum that reshapes the standard cross-entropy loss so that the model will automatically down-weight the contribution of easy examples during training. Unless specified otherwise, for a fair comparison, we tune the method-specific hyper-parameters only, such as α, λ in our method, μ in SPL, and γ in Focal loss. Other hyperparameters such as learning rate, batch size, pacing function, dropout rate are kept unchanged. The details of the hyperparameters settings can be found in the appendix.

The results in Table 1 show that both adaptation (Ours w/o KL) and KL loss (Ours w/o adapt) have better performance than FCL, and combining the two modules can further improve the performance, validating the effect of score adaptation and KL loss in Adaptive CL. Moreover, Adaptive CL also has a considerable improvement than other methods. Compared to FCL and Vanilla, the proposed method is an adaptive curriculum approach and can take the feedback of the current state of the network into account. Therefore, Adaptive CL outperforms Vanilla and FCL and especially has a nearly 4% improvement than Vanilla in CIFAR-100.

It is interesting to find that the performances of SPL and SPCL are not good sometimes. The reason is derived from the fact that SPL prefers examples with lower current losses. During training, the easy examples would be trained at first and the losses of examples in which the model fits well would be lower more. Therefore, those examples would be seen as easy examples all the time and would be trained more likely during the early stages of learning. However, the examples with lower current losses provide less information, so there would be an influence on the performance. Moreover, with a predetermined curriculum, SPCL performed better than SPL most of the time. However, SPCL also prefers examples with low current losses, while our theoretical analysis indicates that with a pseudo difficulty score, the convergence rate is faster if the model pays a lit-

Table 1. Comparison of the proposed method and existing methods. The results are average validation accuracy and STE. The network is the hand-craft network. w/o is short for without.

Method	HE devices	Aquatic Mammals	Small Mammals	Cifar10	Cifar100
Vanilla	72.11 (± 0.270)	56.37 (± 0.173)	56.46 (± 0.223)	78.09 (± 0.083)	42.16 (± 0.169)
SPL [19]	70.88 (± 0.227)	54.95 (± 0.181)	53.66 (± 0.194)	77.95 (± 0.078)	39.70 (± 0.286)
SPCL [15]	71.59 (± 0.171)	57.01 (± 0.215)	57.04 (± 0.144)	76.48 (± 0.182)	45.00 (± 0.230)
Focal loss [20]	73.01 (± 0.255)	56.78 (± 0.196)	56.47 (± 0.189)	79.65 (± 0.060)	42.34 (± 0.136)
FCL [11]	72.10 (± 0.178)	57.10 (± 0.159)	57.45 (± 0.183)	78.51 (± 0.106)	45.25 (± 0.206)
Ours(w/o adapt)	72.25 (± 0.193)	57.16 (± 0.180)	57.65 (± 0.173)	78.56 (± 0.104)	45.49 (± 0.269)
Ours (w/o KL)	73.64 (± 0.149)	57.60 (± 0.144)	57.71 (± 0.194)	79.04 (± 0.095)	45.88 (± 0.338)
Adaptive CL	**74.06** (± 0.210)	**57.71** (± 0.144)	**57.93** (± 0.160)	**79.74** (± 0.074)	**46.08** (± 0.216)

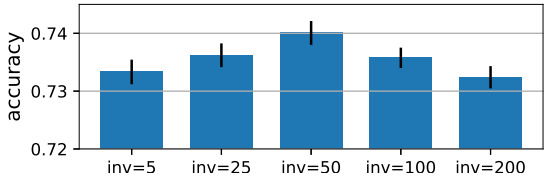


Figure 6. The effect of different inv. We train the hand-crafted network on the HE devices dataset 25 times. Bars indicate the average final validation accuracy over the last few iterations. Error bars indicate the STE.

tle more attention to the examples whose current losses are higher.

Different from SPL and SPCL, our method not only uses prior knowledge (initial curriculum) but also adjusts the curriculum that agrees with theoretical analysis, making our method perform better than SPL and SPCL. Especially, our method performs more than 4% higher than SPL on Small Mammals and more than 3% higher than SPCL on CIFAR-10. As for Focal loss, it adjusts the difficulty of examples dynamically by reshaping loss during training. However, Focal loss may perform badly, especially when the method makes a wrong evaluation of the difficulties of the examples. Such a mistake would affect the subsequent performance such that leading to worse performance. Compared to the Focal loss, we make use of the prior knowledge to determine an initial curriculum, so the proposed method is less affected by such a mistake.

5.4. Interval and initial difficulty score

In this subsection, we exploit the influence of interval and initial difficulty score.

Firstly, we explore the effect of different intervals, i.e., the update frequency of the difficulty score. We set inv = 5, 25, 50, 100, and 200, respectively. Figure 6 shows that $inv = 50$ reaches the best performance. Using too small or too large values of inv will degenerate the performance, because using too small inv can not fully utilize the knowledge of easy examples and using too large inv will adapt the score to the current task too slowly.

Secondly, we also want to know whether the proposed method is still effective if we use an initial difficulty score obtained from another task. Therefore, we obtain an initial difficulty score from a pre-trained VGG-16 network instead of the inception network. We use the new initial difficulty score and compare the proposed method with FCL on three datasets. The results in Table 2 indicate that Adaptive CL still outperforms FCL even with another source of the initial difficulty score. More results including the change of the difficulty score, comparison of running time, the validation accuracy curve of the adaptive CL and baselines, comparison of different pacing functions can be found in the appendix.

Table 2. Comparison of FCL and Adaptive CL with VGG-based initial difficulty score. The results are the average performance and STE. The network is the hand-crafted network and the repetition is 25 times.

Datasets	FCL	Adaptive CL
HE devices	73.32 (± 0.141)	**74.17** (± 0.234)
MS mammals	75.34 (± 0.197)	**75.84** (± 0.150)
Flowers	68.90 (± 0.100)	**69.45** (± 0.159)

6. Conclusion

In this paper, we propose a curriculum learning approach that adapts the difficulty score to the current state of the model, while remembering the knowledge learned from the previous model. Empirical results demonstrate that the proposed algorithm has better performance than existing competitive methods. We also analyze the convergence properties of curriculum learning with a simple non-linear model in the task of binary classification, and the results indicate that the proposed method could improve the convergence rate in the early stages of learning. From the theoretical perspective, a future research direction include analyzing the contribution of curriculum learning on the convergence rate with multi-layer neural networks for multi-class classification.

Acknowledgements. The work of Dr Liu Liu is supported by Australian Research Council project DP-180103424.

References

[1] Srikar Appalaraju and Vineet Chaoji. Image similarity using deep cnn and curriculum learning. *arXiv preprint arXiv:1709.08761*, 2017. 3

[2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009. 1, 2

[3] Antoine Caubrière, Natalia Tomashenko, Antoine Laurent, Emmanuel Morin, Nathalie Camelin, and Yannick Estève. Curriculum-based transfer learning for an effective end-to-end spoken language understanding and domain portability. *arXiv preprint arXiv:1906.07601*, 2019. 3

[4] Wuyang Chen, Zhiding Yu, Zhangyang Wang, and Anima Anandkumar. Automated synthetic-to-real generalization. In *Proceedings of Machine Learning and Systems 2020*, pages 8272–8282, 2020. 4

[5] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4794–4802, 2019. 4

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 248–255. Ieee, 2009. 6

[7] Yueqi Duan, Haidong Zhu, He Wang, Li Yi, Ram Nevatia, and Leonidas J Guibas. Curriculum deepsdf. *arXiv preprint arXiv:2003.08593*, 2020. 3

[8] Jeffrey L Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993. 1

[9] Madan Ravi Ganesh and Jason J Corso. Rethinking curriculum learning with incremental labels and adaptive compensation. *arXiv preprint arXiv:2001.04529*, 2020. 3

[10] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. *arXiv preprint arXiv:1704.03003*, 2017. 3

[11] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning*, pages 2535–2544. PMLR, 2019. 1, 3, 4, 6, 7, 8

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[13] Shiyi He, Chang Xu, Tianyu Guo, Chao Xu, and Dacheng Tao. Reinforced multi-label image classification by exploring curriculum. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. 3

[14] Yuge Huang, Yuhan Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shaoxin Li, Jilin Li, and Feiyue Huang. Curricularface: adaptive curriculum learning loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5901–5910, 2020. 3

[15] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 2, page 6, 2015. 3, 7, 8

[16] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313, 2018. 3

[17] Yongcheng Jing, Yiding Yang, Xinchao Wang, Mingli Song, and Dacheng Tao. Amalgamating knowledge from heterogeneous graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15709–15718, 2021. 4

[18] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2, 6

[19] M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *Advances in neural information processing systems*, pages 1189–1197, 2010. 1, 3, 7, 8

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 3, 7, 8

[21] Reza Lotfian and Carlos Busso. Curriculum learning for speech emotion recognition from crowdsourced labels. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(4):815–826, 2019. 3

[22] Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher-student curriculum learning. *IEEE transactions on neural networks and learning systems*, 2019. 3

[23] Pietro Morerio, Jacopo Cavazza, Riccardo Volpi, René Vidal, and Vittorio Murino. Curriculum dropout. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3544–3552, 2017. 3

[24] Robin R Murphy. Search and rescue robotics. *Springer handbook of robotics*, pages 1151–1173, 2007. 1

[25] Sanmit Narvekar. Curriculum learning in reinforcement learning. In *International Joint Conference on Artificial Intelligence*, pages 5195–5196, 2017. 3

[26] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020. 3

[27] Gustavo Penha and Claudia Hauff. Curriculum learning strategies for ir. In *European Conference on Information Retrieval*, pages 699–713. Springer, 2020. 1, 3, 4

[28] Gail B Peterson. A day of great illumination: Bf skinner's discovery of shaping. *Journal of the experimental analysis of behavior*, 82(3):317–328, 2004. 2

[29] Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom M Mitchell. Competence-based curriculum learning for neural machine translation. *arXiv preprint arXiv:1903.09848*, 2019. 3

[30] Meng Qu, Jian Tang, and Jiawei Han. Curriculum learning for heterogeneous star network embedding via deep reinforcement learning. In *Proceedings of the Eleventh ACM*

International Conference on Web Search and Data Mining, pages 468–476, 2018. 3

[31] Shivesh Ranjan and John HL Hansen. Curriculum learning based approaches for noise robust speaker recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(1):197–210, 2017. 3

[32] Conditioned Reflexes. an investigation of the physiological activity of the cerebral cortex. *Trans, by GV Anrep.) London: Oxford Univ. Press*, 1927. 2

[33] Dana Ruiter, Cristina España-Bonet, and Josef van Genabith. Self-induced curriculum learning in neural machine translation. *arXiv preprint arXiv:2004.03151*, 2020. 3

[34] Shreyas Saxena, Oncel Tuzel, and Dennis DeCoste. Data parameters: A new family of parameters for learning a differentiable curriculum. In *Advances in Neural Information Processing Systems*, pages 11095–11105, 2019. 3

[35] Bernhard Scholkopf, Kah-Kay Sung, Christopher JC Burges, Federico Girosi, Partha Niyogi, Tomaso Poggio, and Vladimir Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE transactions on Signal Processing*, 45(11):2758–2765, 1997. 6

[36] Yang Shu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Transferable curriculum for weakly-supervised domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4951–4958, 2019. 3

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[38] Burrhus F Skinner. Reinforcement today. *American Psychologist*, 13(3):94, 1958. 2

[39] Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407*, 2017. 3

[40] Maxwell Svetlik, Matteo Leonetti, Jivko Sinapov, Rishi Shah, Nick Walker, and Peter Stone. Automatic curriculum graph generation for reinforcement learning agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2590–2596, 2017. 3

[41] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 6

[42] Yiru Wang, Weihao Gan, Jie Yang, Wei Wu, and Junjie Yan. Dynamic curriculum learning for imbalanced data classification. In *Proceedings of the IEEE international conference on computer vision*, pages 5017–5026, 2019. 3

[43] Zhen Wang, Liu Liu, and Dacheng Tao. Deep streaming label learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*, volume 119, pages 9963–9972, 2020. 4

[44] Daphna Weinshall and Dan Amir. Theory of curriculum learning, with convex loss functions. *arXiv preprint arXiv:1812.03472*, 2018. 3

[45] Daphna Weinshall and Dan Amir. Theory of curriculum learning, with convex loss functions. *Journal of Machine Learning Research*, 21(222):1–19, 2020. 2, 5

[46] Daphna Weinshall, Gad Cohen, and Dan Amir. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *International Conference on Machine Learning*, pages 5238–5246. PMLR, 2018. 1, 2, 3, 4, 5

[47] Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, 2020. 3