Learning Optimal Conformal Classifiers

David Stutz [1 2] **Krishnamurthy (Dj) Dvijotham** [1] **Ali Taylan Cemgil** [1] **Arnaud Doucet** [1]

Abstract

Modern deep learning based classifiers show very high accuracy on test data but this does *not* provide sufficient guarantees for safe deployment, especially in high-stake AI applications. Usually, predictions are obtained without a reliable uncertainty estimate or a formal guarantee. *Conformal prediction (CP)* addresses these issues by using the classifier's predictions to form *confidence sets* containing the true class with a user-specified probability. However, using CP as a separate processing step after training prevents the underlying model from adapting to the prediction of confidence sets. We propose **conformal training (ConfTr)**, a training procedure that "simulates" conformalization on each mini-batch using differentiable CP. Compared to standard training, ConfTr reduces the average confidence set size (*inefficiency*) of state-of-the-art CP methods applied after training. Moreover, it can can influence how inefficiency is distributed across classes, while retaining the coverage guarantee. We plan to make our code publicly available.

1. Introduction

In classification tasks, for input x, we approximate the posterior distribution over classes $y \in [K] := \{1, \ldots, K\}$, denoted $\pi_y(x) \approx P(Y = y|X = x)$. By following Bayes' decision rule and predicting the *single* class with highest posterior probability, deep networks $\pi_{\theta,y}(x)$ with parameters θ achieve impressive accuracy on held-out test sets. However, this does not *guarantee* safe deployment. *Conformal prediction (CP)* (Vovk et al., 2005) uses a post-training calibration step to *guarantee* a user-specified *coverage*: by allowing to predict confidence sets $C(X) \subseteq [K]$, CP guarantees the true class Y to be included with confidence level α, *i.e.* $P(Y \in C(X)) \geq 1 - \alpha$. This guarantee only requires exchangeability w.r.t. the calibration examples (X_i, Y_i), $i \in I_{\text{cal}}$ and has to be understood marginally across examples (X, Y) and calibration sets I_{cal}.

CP also outputs intuitive uncertainty estimates: larger confidence sets $|C(X)|$ generally convey higher uncertainty. Although CP is agnostic to details of the underlying model $\pi_\theta(x)$, the obtained uncertainty estimates depend strongly on the model's performance. If the underlying classifier is poor, CP results in too large and thus uninformative confidence sets. "Uneven" uncertainty estimates are also a common issue, where uncertainty increases for difficult classes, often alongside lower coverage. Recent work tackles such problems by explicitly minimizing inefficiency (Sadinle et al., 2019; Angelopoulos et al., 2021) or working towards (approximate) conditional coverage (Romano et al., 2020; Cauchois et al., 2020). These various objectives are typically achieved by changing the so-called *conformity scores* on which the construction of the confidence set $C(X)$ relies. In all cases, CP is used as a post-training calibration step. In contrast, our work does *not* focus on advancing CP itself, *e.g.*, through new conformity scores, but develops a novel training procedure for the classifier π_θ.

Indeed, while the flexibility of CP regarding the underlying model appears attractive, it is also a severe limitation: Learning the model parameters θ is *not* informed about the post-hoc "conformalization", *i.e.*, they are are not tuned towards any specific objective such as reducing inefficiency. During training, the model will typically be trained to minimize cross-entropy loss. At test time, in contrast, it is used to obtain a set predictor $C(X)$ with specific properties such as low inefficiency. Concurrent work (Bellotti, 2021) addresses this issue by learning a set predictor $C(X)$ through (soft) thresholding of logits. However, this approach ignores the crucial calibration step of CP during training and does *not* allow to optimize losses beyond marginal coverage or inefficiency. Our work subsumes (Bellotti, 2021), but additionally considers the calibration step during training, which is crucial for further decreasing inefficiency and allowing fine-grained control over class-conditional inefficiency.

Our **contributions** can be summarized as follows:

1. We propose **conformal training (ConfTr)**, a procedure allowing to train model and conformal wrapper *end-to-end*. This is achieved by developing smooth

[*]Equal contribution [1]DeepMind [2]Max Planck Institute for Informatics, Saarland Informatics Campus. Correspondence to: David Stutz <dstutz@deepmind.com>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

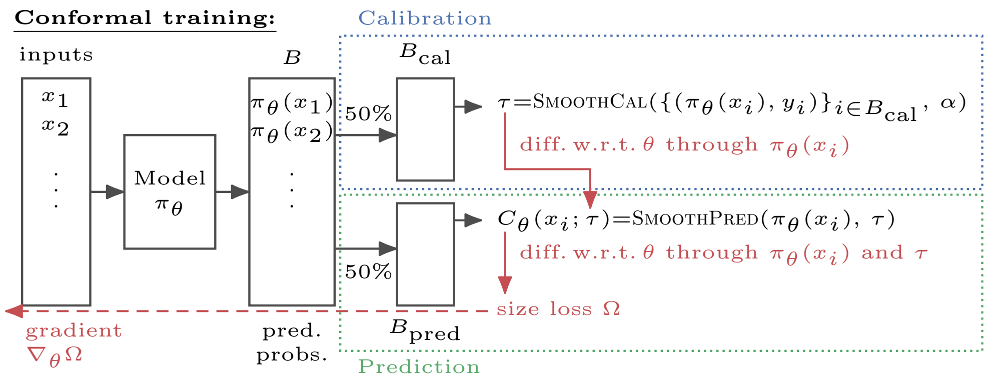
Conformal training:
inputs
x_1
x_2
Model π_θ
B
$\pi_\theta(x_1)$
$\pi_\theta(x_2)$
pred. probs.
Calibration
B_{cal}
50%
τ=SmoothCal($\{(\pi_\theta(x_i), y_i)\}_{i\in B_{\text{cal}}}, \alpha$)
diff. w.r.t. θ through $\pi_\theta(x_i)$
$C_\theta(x_i;\tau)$=SmoothPred($\pi_\theta(x_i), \tau$)
diff. w.r.t. θ through $\pi_\theta(x_i)$ and τ
50%
size loss Ω
B_{pred}
gradient $\nabla_\theta\Omega$
Prediction

Figure 1: **Illustration of *conformal training (ConfTr)*:** We develop differentiable prediction and calibration steps for *conformal prediction (CP)*, SmoothCal and SmoothPred. During training, this allows ConfTr to "simulate" CP on each mini-batch B by calibrating on the first half B_{cal} and predicting on the other half B_{pred}. While ConfTr can optimize arbitrary losses on the predicted confidence sets, we concentrate on reducing (class-conditional) average confidence set size (*inefficiency*) using a size loss Ω. *After* training using, *any* existing CP method can be used to obtain a coverage guarantee.

implementations of recent CP methods for use during training. On each mini-batch, ConfTr "simulates" conformalization, using half of the batch for calibration, and the other half for prediction and loss computation, *c.f.* Fig. 1. After training, *any* existing CP method can provide a coverage guarantee.

2. In experiments, using ConfTr for training consistently reduces the inefficiency of conformal predictors such as *threshold CP (*Thr*)* (Sadinle et al., 2019) or APS (Romano et al., 2020) applied *after* training. We further improve over (Bellotti, 2021), illustrating the importance of the calibration step during training, and allow to reduce *class-conditional* inefficiency.

Because ConfTr is agnostic to the CP method used at test time, our work is complementary to most related work, *i.e.*, *any* advancement in terms of CP is directly applicable to ConfTr. Most importantly, ConfTr preserves the coverage guarantee obtained through CP.

This is a short version of our ICLR 2022 paper (Stutz et al., 2021). While this paper is intended to be self-contained, we refer to (Stutz et al., 2021) for additional details and complementary experiments on influencing the composition of confidence sets using ConfTr. We plan to make our code and experiments publicly available.

2. Differentiable Conformal Prediction

We are interested in training the model π_θ end-to-end with the conformal wrapper. In the following, we briefly summarize the conformal predictor of (Sadinle et al., 2019) before developing a smooth variant in Sec. 2.1 for use during training in Sec. 2.2.

The **threshold conformal predictor (Thr)** (Sadinle et al., 2019) constructs the confidence sets by thresholding probabilities: $C_\theta(x;\tau) := \{k : \pi_{\theta,k}(x) =: E_\theta(x,k) \geq \tau\}$. Here, the subscript C_θ makes the dependence on the model π_θ and its parameters θ explicit. During calibration, τ is computed as the $\alpha(1 + {}^1/_{|I_{\text{cal}}|})$-quantile of the so-called conformity scores $E_\theta(x_i, y_i) = \pi_{\theta,y_i}(x_i)$. The conformity scores indicate, for each example, the threshold that ensures coverage. Marginal coverage of $(1-\alpha)$ is guaranteed on a test example (X, Y). In practice, Thr can also be applied on logits (ThrL) or log-probabilities (ThrLP) instead of probabilities. Other conformal predictors, *e.g.*, (Romano et al., 2020; Angelopoulos et al., 2021) follow the same procedure, but using different conformity scores E.

Performance of CP is then measured using two metrics: (empirical and marginal) **coverage (Cover)**, computed as the fraction of true labels y included in $C(x)$, as well as **inefficiency (Ineff)**, the average size of $C(x)$. Due to the marginal coverage guarantee provided by CP, the empirical coverage, when averaged across several calibration/test splits, is Cover $\approx 1-\alpha$. Thus, we concentrate on inefficiency as the main metric to compare across CP methods and models. All CP methods have in common that they are used as a "wrapper" around π_θ. Then, "better" CP methods generally result in lower inefficiency for a *fixed* model π_θ. However, fine-grained control over inefficiency, *e.g.*, conditioned on the class or the composition of the $C(X)$ is generally difficult to obtain. Integrating CP into the training procedure promises a higher degree of control, however, requires differentiable CP implementations.

2.1. Differentiable Prediction and Calibration Steps

Differentiating through CP involves differentiable prediction and calibration steps:

Prediction involves thresholding the conformity scores $E_\theta(x, k)$, which can be smoothed using the sigmoid

```
1: function ConformalTraining($\alpha$, $\lambda$=1)
2:     for mini-batch $B = B_{\text{cal}} \uplus B_{\text{pred}}$ do
3:         $\tau = $ SmoothCal($\{(\pi_\theta(x_i), y_i)\}_{i \in B_{\text{cal}}}, \alpha$)
4:         $C_\theta(x_i; \tau) = $ SmoothPred($\pi_\theta(x_i), \tau$)
5:         $\Omega_B = \sum_{i \in B_{\text{pred}}} \Omega(C_\theta(x_i; \tau))$
6:         update parameters $\theta$ using $\Delta = \nabla_\theta {}^1/_{|B_{\text{pred}}|} \Omega_B$
```

```
1: function SmoothPred($\pi_\theta(x)$, $\tau$, $T$=1)
2:     compute $E_\theta(x,k)$, $k \in [K]$ {e.g., $E_\theta(x,k) = \pi_{\theta,k}(x)$}
3:     return $C_{\theta,k}(x;\tau) = \sigma(\frac{(E_\theta(x,k)-\tau)}{T})$, $k \in [K]$
```

```
1: function SmoothCal($\{(\pi_\theta(x_i), y_i\}_{i=1}^n$, $\alpha$)
2:     return SmoothQuant($\{E_\theta(x_i, y_i)\}$, $\alpha(1+\frac{1}{n})$)
```

Algorithm 1: **Conformal Training (ConfTr) and Smooth CP:** ConfTr (left) calibrates on a part of each mini-batch, B_{cal}. Thereby, we obtain empirical coverage close to $(1-\alpha)$ on the other part, B_{pred}. Then, the inefficiency on B_{pred} is minimized to update the model parameters θ. Smooth implementations of prediction and calibration are used (right, see text for details).

function $\sigma(z) = {}^1/_{1+\exp(-z)}$ and a temperature hyper-parameter T: $C_{\theta,k}(x;\tau) := \sigma\left({}^{(E_\theta(x,k)-\tau)}/_T\right)$. Essentially, $C_{\theta,k}(x;\tau) \in [0,1]$ represents a *soft* assignment of class k to the confidence set, *i.e.*, can be interpreted as the probability of k being included. For $T \to 0$, the "hard" confidence set will be recovered, *i.e.*, $C_{\theta,k}(x;\tau) = 1$ for $k \in C_\theta(x;\tau)$ and 0 otherwise. For Thr, the conformity scores are already differentiable as $E(x,k) = \pi_{\theta,k}(x)$.

Calibration additionally involves a differentiable quantile computation. This can be accomplished using any smooth sorting approach (Blondel et al., 2020; Cuturi et al., 2019; Williamson, 2020), often involving a "dispersion" hyper-parameter ϵ such that smooth sorting approximates "hard" sorting for $\epsilon \to 0$. Overall, this results in the threshold τ being differentiable w.r.t. the predictions of the calibration examples $\{(\pi_\theta(x_i), y_i)\}_{i \in I_{\text{cal}}}$ and the model's parameters θ.

Overall, this allows us to differentiate through both calibration and prediction w.r.t. the *model parameters* θ. Unfortunately, as this approximation is using smooth operations, the coverage guarantee seems lost. However, in the limit of $T, \epsilon \to 0$ we recover the original non-smooth computations and the corresponding coverage guarantee. Thus, it is reasonable to assume that, in practice, we *empirically* obtain coverage close to $(1-\alpha)$. We found that this is sufficient because these smooth variants are *only* used during training. At test time, we use the original (non-smooth) implementations.

2.2. ConfTr by Optimizing Inefficiency

The key idea of **conformal training (ConfTr)** is to "simulate" CP during training, *i.e.*, performing *both* calibration and prediction steps on each mini-batch. Thus, ConfTr can be viewed as a generalization of (Bellotti, 2021) that just differentiates through the prediction step with a fixed threshold. Specifically, during stochastic gradient descent training, ConfTr performs (differentiable) CP by splitting each mini-batch B in half: On the first half B_{cal}, we calibrate τ by computing the $\alpha(1 + {}^1/_{|B_{\text{cal}}|})$-quantile of the conformity scores in a differentiable manner. On the second half, we use τ to predict $C_\theta(x_i;\tau)$, $i \in B_{\text{pred}}$. Assuming empirical coverage to be close to $(1-\alpha)$ in practice, we only need to minimize inefficiency during training:

$$\Omega(C_\theta(x;\tau)) = \max\left(0, \sum_{k=1}^{K} C_{\theta,k}(x;\tau) - \kappa\right). \quad (1)$$

We emphasize that ConfTr optimizes the model parameters θ on which the confidence sets C_θ depend. Here, Ω is a "smooth" *size loss* intended to minimize the expected inefficiency, *i.e.*, $\mathbb{E}[|C_\theta(X;\tau)|]$. Remember that $C_{\pi,k}(x;\tau)$ can be understood as a soft assignment of class k to the confidence set $C_\theta(x;\tau)$. By default, we use $\kappa = 1$. Through additional weights $\omega \cdot \Omega(C(X;\tau))$ with $\omega := \omega(Y)$ depending on the ground truth Y, we can further control class-conditional inefficiency. In (Stutz et al., 2021), we also demonstrate how to integrate an additional classification loss $\mathcal{L}_{\text{class}}$ in Eq. (1) to control the composition of confidence sets. After training, any CP method can be applied to re-calibrate τ on a held-out calibration set I_{cal} as usual, *i.e.*, the thresholds τ obtained during training are *not* kept. This ensures that we obtain a coverage guarantee of CP.

3. Experiments

Our experiments demonstrate that ConfTr can reduce (class-conditional) inefficiency of (non-differentiable) Thr as well as APS (Romano et al., 2020) compared to CP applied to a baseline model trained using cross-entropy loss. Thereby, we outperform concurrent work of (Bellotti, 2021), denoted Bel. We consider several benchmark datasets (LeCun et al., 1998; Xiao et al., 2017; Cohen et al., 2017; Krizhevsky, 2009) as well as architectures and report inefficiency averaged across 10 random calibration/test splits for 10 trained models for each method. We set $\alpha = 0.01$ and use the same α during training using ConfTr. Hyper-parameters are optimized for Thr or APS individually and ConfTr is trained using ThrLP, *i.e.*, Thr applied on log-probabilities for better gradient flow.

Main Results: In Tab. 1, we summarize the inefficiency reductions possible through ConfTr (trained with ThrLP) in comparison to Bel (trained with ThrL) and the baseline. Bel does *not* consistently improve inefficiency on all datasets. Specifically, on MNIST, EMNIST or CIFAR100, inefficiency actually *worsens*. Our ConfTr, in contrast, reduces inefficiency consistently, not only for Thr but also

Table 1: **Inefficiency Reduction**, comparing Bel (trained w/ THRL) and ConfTr (trained w/ THRLP) using THR or APS at test time (α=0.01). We also report improvements relative to the baseline, *i.e.*, standard cross-entropy training, in percentage in parentheses. ConfTr results in a consistent improvement of inefficiency for both THR and APS. Training with an additional classification loss, $\mathcal{L}_{\text{class}}(C_\theta(x;\tau), y) = (1 - C_{\theta,y}(x;\tau))$ in addition to the size loss in Eq. (1) can be helpful. * On CIFAR, the inefficiency reduction is smaller as ConfTr trains a linear model on pre-trained ResNet features.

Inefficiency $\downarrow$, ConfTr (trained w/ THRLP), $\alpha = 0.01$								Split ($\cdot 10^3$)	K	Model
	THR				APS			(Split as train/cal/test)		
Dataset	Basel.	Bel	ConfTr	$+\mathcal{L}_{\text{class}}$	Basel.	ConfTr	$+\mathcal{L}_{\text{class}}$			
MNIST	2.23	2.70	2.18	**2.11** (-5.4%)	2.50	2.16	**2.14** (-14.4%)	55/5/10	10	linear
F-MNIST	2.05	1.90	1.69	**1.67** (-18.5%)	2.36	1.82	**1.72** (-27.1%)	55/5/10	10	2-layer MLP
EMNIST	2.66	3.48	2.66	**2.49** (-6.4%)	4.23	**2.86**	2.87 (-32.2%)	98.8/5.2/18.8	52	2-layer MLP
CIFAR10	2.93	2.93	2.88	**2.84** (-3.1%)	3.30	3.05	**2.93** (-11.2%)	45/5/10	10	ResNet-34*
CIFAR100	10.63	10.91	10.78	**10.44** (-1.8%)	16.62	12.99	**12.73** (-23.4%)	45/5/10	100	ResNet-50*

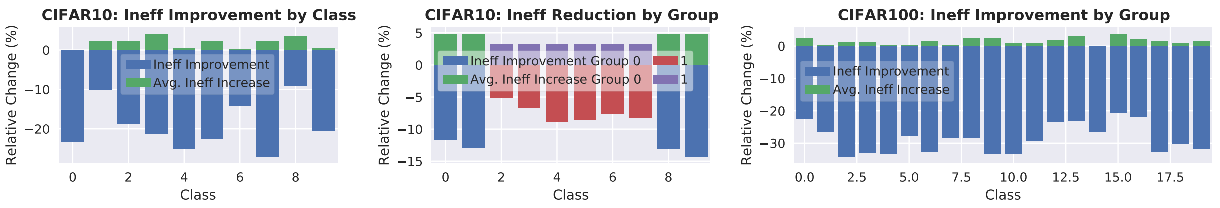

Figure 2: **Shaping Class-Conditional Inefficiency on CIFAR:** Possible inefficiency reductions, in percentage change, per class (blue) and the impact on the overall, average inefficiency across classes (green). *Left:* Significant inefficiency reductions are possible for all classes on CIFAR10. *Middle:* The same strategy applies to groups of classes, *e.g.*, "vehicles" vs "animals", as well. *Right:* Similarly, on CIFAR100, we group classes by their coarse class (20 groups à 5 classes), see (Krizhevsky, 2009), allowing inefficiency improvements of more than 30% per individual group.

for APS. Here, improvements on CIFAR for THR are generally less pronounced. This is likely because we train linear models on top of a pre-trained ResNet (He et al., 2016) where features are not taking into account conformalization at test time. For APS, in contrast, improvements are still significant. Across all datasets, training with an additional classification loss $\mathcal{L}_{\text{class}}$ generally performs slightly better, especially for datasets with many classes such as EMNIST (K=52) or CIFAR100 (K=100). Overall, ConfTr yields significant inefficiency reductions, independent of the CP method used at test time.

Shaping Conditional Inefficiency: We use ConfTr to reduce class-conditional inefficiency for specific classes or a group of classes. This is motivated by observing that inefficiency (*i.e.*, uncertainty) varies widely across classes, with more difficult classes (*e.g.*, "cat" on CIFAR10) obtaining higher inefficiency than easier ones (*e.g.*, "automobile"). Thus, in Fig. 2, we increase the weight $\omega = 10$ of the size loss for specific classes or groups of classes. Then, we report the *relative* change in percentage, showing that inefficiency reductions of 20% or more are possible for many classes, including "cat" on CIFAR10 (left, blue). This is also possible for two groups of classes, "vehicles" vs. "animals" (middle). However, these reductions usually come at the cost of a slight increase in average inefficiency across all classes (green). On CIFAR100, we consider 20 coarse classes, each containing 5 of the 100 classes (right). Again, significant inefficiency reductions per coarse class are possible. We found that these observations generalize to all other considered datasets and different class groups.

4. Conclusion and Outlook

We introduced **conformal training (ConfTr)**, a novel method to train conformal predictors *end-to-end* with the underlying model. This addresses a major limitation of conformal prediction (CP) in practice: The model is fixed, leaving CP little to no control over the predicted confidence sets. In thorough experiments, we demonstrated that ConfTr can improve overall inefficiency as well as class-conditional inefficiency of state-of-the-art CP methods such as THR or APS. Importantly, ConfTr does *not* lose the (marginal) coverage guarantee provided by CP.

In (Stutz et al., 2021), we show that ConfTr also allows to manipulate the composition of the predicted confidence sets. For example, ConfTr can penalize coverage confusion, *i.e.*, cases where two dissimilar classes are included in the same confidence sets. This notion can also be extended to avoiding overlap between groups of classes. We also note that ConfTr is agnostic to the CP method used at test time, making it directly applicable to, *e.g.*, future conformity scores. Moreover, ConfTr can easily be adapted for regression tasks or considering conditional or application-specific guarantees as in (Sadinle et al., 2016; Bates et al., 2021).

References

Angelopoulos, A. N., Bates, S., Jordan, M., and Malik, J. Uncertainty sets for image classifiers using conformal prediction. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2021.

Bates, S., Angelopoulos, A., Lei, L., Malik, J., and Jordan, M. I. Distribution-free, risk-controlling prediction sets. *arXiv.org*, abs/2101.02703, 2021.

Bellotti, A. Optimized conformal classification using gradient descent approximation. *arXiv.org*, abs/2105.11255, 2021.

Blondel, M., Teboul, O., Berthet, Q., and Djolonga, J. Fast differentiable sorting and ranking. In *Proc. of the International Conference on Machine Learning (ICML)*, 2020.

Cauchois, M., Gupta, S., and Duchi, J. Knowing what you know: valid and validated confidence sets in multiclass and multilabel prediction. *arXiv.org*, abs/2004.10181, 2020.

Cohen, G., Afshar, S., Tapson, J., and van Schaik, A. EMNIST: an extension of MNIST to handwritten letters. *arXiv.org*, abs/1702.05373, 2017.

Cuturi, M., Teboul, O., and Vert, J. Differentiable ranking and sorting using optimal transport. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, 86(11):2278–2324, 1998.

Romano, Y., Sesia, M., and Candès, E. J. Classification with valid and adaptive coverage. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Sadinle, M., Lei, J., and Wasserman, L. A. Least ambiguous set-valued classifiers with bounded error levels. *arXiv.org*, abs/1609.00451, 2016.

Sadinle, M., Lei, J., and Wasserman, L. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association (JASA)*, 114(525): 223–234, 2019.

Stutz, D., Dvijotham, K., Cemgil, A. T., and Doucet, A. Learning optimal conformal classifiers. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2021.

Vovk, V., Gammerman, A., and Shafer, G. *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg, 2005.

Williamson, J. H. Differentiable parallel approximate sorting networks. `https://johnhw.github.io/differentiable_sorting/index.md.html`, 2020.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv.org*, abs/1708.07747, 2017.