FreeAugment: Data Augmentation Search Across All Degrees of Freedom

Tom Bekor*⊙, Niv Nayman*⊙, and Lihi Zelnik-Manor⊙

Technion - Israel Institute of Technology, Haifa, Israel
Corresponding author: `tom.bekor@campus.technion.ac.il`

Abstract. Data augmentation has become an integral part of deep learning, as it is known to improve the generalization capabilities of neural networks. Since the most effective set of image transformations differs between tasks and domains, automatic data augmentation search aims to alleviate the extreme burden of manually finding the optimal image transformations. However, current methods are not able to jointly optimize all degrees of freedom: (1) the number of transformations to be applied, their (2) types, (3) order, and (4) magnitudes. Many existing methods risk picking the same transformation more than once, limit the search to two transformations only, or search for the number of transformations exhaustively or iteratively in a myopic manner. Our approach, FreeAugment, is the first to achieve global optimization of all four degrees of freedom simultaneously, using a fully differentiable method. It efficiently learns the number of transformations and a probability distribution over their permutations, inherently refraining from redundant repetition while sampling. Our experiments demonstrate that this joint learning of all degrees of freedom significantly improves performance, achieving state-of-the-art results on various natural image benchmarks and beyond across other domains.[1]

Keywords: Data Augmentation · AutoML · Differentiable Optimization

1 Introduction

Data augmentations expand the training set by generating virtual samples through random transformations to the original ones. This mitigates the overfitting [39] problem when training large deep neural networks. However, designing a data augmentation pipeline for a given task requires domain knowledge, as different domains benefit from different types of transformations of different strengths. This requirement has been alleviated by Data Augmentation Search (DAS), which enables the automatic customization of data augmentation for different tasks and domains. DAS methods sample from an optimized data augmentation policy, that consists of the probability distribution of choosing transformations

---

[1] Project page: https://tombekor.github.io/FreeAugment-web
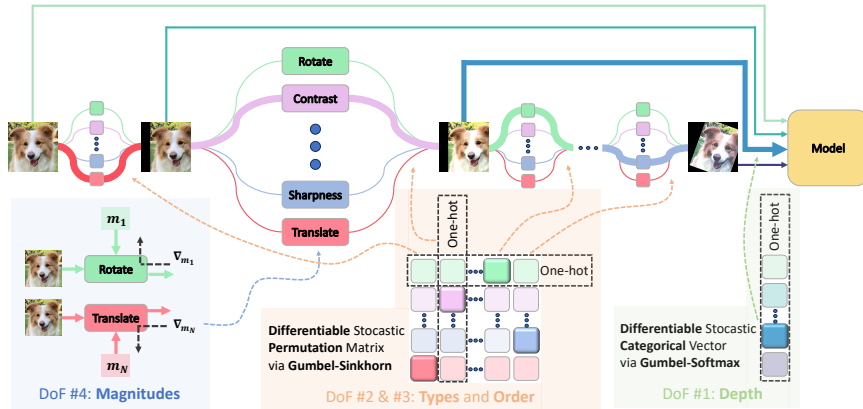
* Equal contribution.

Fig. 1: FreeAugment differentiable search space over four degrees of freedom (DoF). The search space over policy depth, transformations type, order, and magnitudes is constructed in a fully differentiable manner and these are learned end-to-end.

from a predefined set together with their magnitudes (intensities). This combinatorial search space of policies makes the optimization problem challenging. Differentiable DAS methods [21, 25, 29, 44, 52] have taken a gradient-based approach to solve a bilevel optimization problem that targets the best augmentation policy, such that a neural network trained with this policy on the training set (inner problem) generalizes well to a validation set (outer problem). This approach has shown good results; however, previous methods either limit the number of sequentially applied transformations (policy depth) to two [5, 6, 8, 12, 21, 22, 25], exhaustively enumerate through several policy depth values [27, 52], or myopically increase the depth at every search iteration [56]. In addition, seeking the best order of transformations, some methods sample predefined sub-policies, including all possible permutations of transformations [5, 8, 21, 22]. As the number of those grows exponentially with the policy depth, these methods limit the depth to at most two. Other methods sequentially sample transformations from independent categorical distributions and risk sampling the same transformation more than once [25, 27, 52, 56].

In this paper, we aim to globally optimize the augmentation policy across all four degrees of freedom (DoF): (1) the number of sequentially applied transformations (policy depth), their (2) type, (3) order, and (4) magnitudes. To this end, we introduce FreeAugment, the first fully differentiable DAS method to jointly optimize all DoF in an end-to-end manner using gradient descent, without turning to complementary heuristics or greedy strategies. As illustrated in Fig. 1, FreeAugment achieves all of that by (1) inducing a learnable probability distribution over the policy depth and thus can sample a varying depth. (2) Optimizing the distributions over types and order of transformations is unified as learning the probability distribution over permutations. This formalism inherently guarantees that images do not undergo any transformation more than once. In practice, this distribution is closely approximated by the Gumbel-Sinkhorn

method [28], which allows sampling from all possible permutations of a pre-defined set of transformations and learning the underlying probability distribution in a fully differentiable manner. Lastly, (3) stochastic magnitudes for each augmentation are sampled from a parameterized probability distribution, whose parameters are learned by backpropagation through the differentiable implementation of the candidate transformations during the search. This entire composition is optimized on a validation set in an end-to-end manner by bilevel optimization.

Our experiments show that the proposed framework is general and robust to a variety of domains. It finds policies that lead to comparable or better results than other methods on the natural images benchmarks CIFAR10/100 [20] and ImageNet-100 [45], as well as on DomainNet [36], which includes sketches, paintings, drawings, cliparts and infographs.

To summarize, our key contributions are three-fold:

- We propose a novel way to jointly learn the number of transformations in an end-to-end manner, without reverting to heuristics or multi-stage solutions.
- To avoid redundant repetition of transformations, we formalize the search over their types and order as learning the probability distribution over permutations of transformations, and utilize efficient tools for doing so in a fully differentiable manner.
- Through extensive experimentation, we showcase the synergistic impact of jointly learning all degrees of freedom of data augmentation policies. This leads to state-of-the-art performance on benchmarks of natural images and other domains compared to previous DAS methods.

2   Related Work

To alleviate the domain-knowledge and manual labor required for choosing the right data augmentation policy for any given domain, AutoAugment (AA) [5] was the pioneering work to automate the search of data augmentations. This was done by modeling the policy search problem as a sequence prediction problem, and using a recurrent neural network controller in a Reinforcement Learning (RL) framework to predict the type and magnitude of at most two sequential transformations. Since RL tends to suffer from high variance, this method is sample inefficient and hence requires a large amount of data and compute. A following evolutionary approach, PBA [12], is based on sampling population of policies and hence still expensive. A Bayesian optimization [43] based method, FastAA [22] does not leverage gradient information, and thus in practice those all are not learned in an end to end manner and do not perform better than random search, e.g., RandAugment [6] and TrivialAugment [30]. Inspired by differentiable neural architecture search [26, 32–34], differentiable DAS formulate the search space of augmentations so that it can be optimized using gradient decent. FasterAA [8] minimizes the distance between the original and augmented image distributions, AdversarialAA [55] and TeachAugment [42] maximize the training loss with respect to the augmentation policy under some regularizations, and

MADAO [9] minimizes the validation loss directly with Neumann series approximation of the gradients. MedPipe [11], DAAS [50] and DHA [57] jointly search for augmentation policies and network architectures. OHLA [23], DADA [21], DDAS [25], DABO [29], DRA [52] and SLACK [27] use meta-learning with one-step gradient update and differentiable optimization. Most of the aforementioned [5,6,9,12,21–23,25,30,55] limit the number of sequential transformations to two, and some of those [8,52] further enumerate several possible values for this number (up to four transformations) but eventually fixate on a couple of transformations as well. DeepAA [56] was the first to search beyond two transformations, but it does so by myopically stacking augmentation layers one at a time until convergence to an identity layer. FreeAugment is the first to perform global end-to-end optimization over the distribution of the number of transformations in a fully differentiable manner. Additionally, all of the listed methods, but DeepAA and SLACK, determine the order of applied transformations by sampling from predefined sub-policies of all possible permutations. The number of those grows exponentially with the policy depth, which is hence limited to be at most two. FreeAugment is the first to learn the distribution over all permutations of different lengths.

3   Method

Data augmentation can be viewed as a sequential application of image transformations over an input image X, where an image transformation τ with an application *magnitude m* transforms the input image into another *augmented* image $\tau(X; m)$ of the same size. This magnitude defines how strongly to transform the input image, e.g., the rotation angle for a rotation transformation. Image transformations are selected from a predefined set $\{\tau_1, \ldots, \tau_N\}$ of N candidate *elementary transformations*.

Following previous work [5, 8, 21, 22, 27], we define the augmentation policy as a probabilistic model $\mathcal{P}_\phi$ that generates a sequence of elementary transformations with corresponding magnitudes, governed by parameters ϕ as outlined in Algorithm 1. Those parameters are divided into three groups, δ, Π, and μ, associated with the different degrees of freedom.

Algorithm 1: FreeAugment Policy Sampling at Search

Data: Input image X_0, policy parameters $\phi = (\delta, \Pi, \mu)$
Result: Augmented image X'
1 $\mathbf{d} \sim \text{Gumbel-Softmax}(\delta)$;   // Sample depth one-hot vector (Sec. 3.1)
2 $\mathbf{P} \sim \text{Gumbel-Sinkhorn}(\Pi)$;   // Sample a permutation matrix (Sec. 3.2)
3 $\mathbf{M} \sim \text{Uniform}(\mu)$;   // Sample a magnitude matrix (Sec. 3.3)
4 **for** $k \in \{1, \ldots, K\}$ **do**   // Iterate over augmentation layers
5   $\quad \Big| \quad X_k \leftarrow \sum_{i=1}^{N} \mathbf{P}_{ik} \cdot \tau_i(X_{k-1}, \mathbf{M}_{ik})$
6 **end**
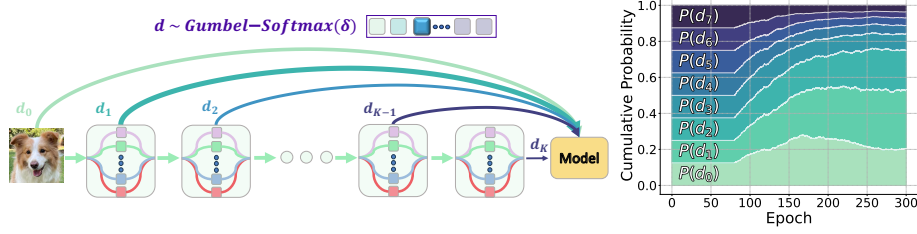7 $X' \leftarrow \sum_{k=0}^{K} \mathbf{d}_k \cdot X_k$

Fig. 2: Illustration of FreeAugment's policy depth search mechanism (Left) and learned depth distribution along the search (Right), where the last epoch represents the resulting probability distribution over policy depth.

3.1   1st Degree of Freedom: Depth

The policy depth refers to the number of sequential transformations applied to a single image. To dynamically determine this depth, inspired by [13,31], we induce a learnable Gumbel-Softmax distribution [16] over categorical depths. It is parameterized by a vector of logits δ, where each element δ_k for $k \in \{0, \ldots, K\}$ represents the unnormalized log-probability of selecting an augmentation policy with depth k, as illustrated in Fig. 2 (Left). This results in a probability distribution over the depths that is smooth and differentiable:

$$\mathbf{d} \sim \text{Gumbel-Softmax}(\delta; t), \quad \text{such that } \mathbb{P}(d_k = 1 \mid t) = \frac{e^{(\delta_k + g_k)/t}}{\sum_{i=1}^{K} e^{(\delta_i + g_i)/t}} \quad (1)$$

where g_i are i.i.d samples from a standard Gumbel distribution, and t is the temperature parameter. Effectively, the straight-through (ST) gradient estimator [2] is utilized to to efficiently learn the logits δ by backpropagating through the discrete choice of augmentation depth of maximal value in $\mathbf{d}$, as demonstrated in Fig. 2 (Right).

3.2   2nd and 3rd Degrees of Freedom: Types & Order

Learning the distribution over types and order of transformations, requires sampling a transformation for every augmentation layer. Doing so for each layer independently using the Gumbel-Softmax distribution, as in [52], permits the repetition of the same transformations in several layers.

To mitigate that, we view the search for the types and order of transformations as a matching problem, where we seek to match between N elementary transformations to K augmentation layers. Thus, we couple the different layers and learn a probability distribution for sampling an entire permutation matrix $\mathbf{P}$ of size $N \times K$, where not only its columns are normalized but also its rows. Its kth column $\mathbf{P}_{1:N,k}$ represents the distribution over transformations at augmentation layer k, as illustrated the middle of Fig. 1 and the top middle of Fig. 3.
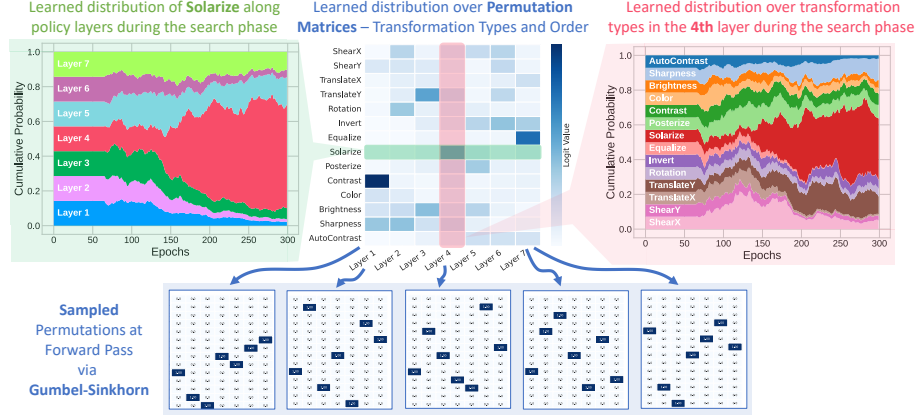
Fig. 3: Learning the transformation types and order as the distribution over permutations of transformations (Top Middle) and sampling from it (Bottom). The learning dynamics of a column (Top Right) and a row (Top Left) of the matrix along the search are shown, with the final values embedded in the matrix.

To this end, we utilize the Gumbel-Sinkhorn operator [28] for differentiating through permutation sampling as an analogy of the Gumbel-Softmax for differentiating through categorical sampling.

We first state an analog of the normalization implemented by the softmax; a double stochastic matrix (DSM) is a square matrix of non-negative real numbers with the property that every row and every column sums to one. Sinkhorn [40,41] showed that any non-negative square matrix $\exp(\bar{\Pi}) \in \mathbb{R}^{N \times N}$ (with full support [19]) can be converted to a DSM by alternating between rescaling its rows and columns to one using the Sinkhorn operator [1,14]:

$$S(\bar{\Pi}) = \lim_{l \to \infty} S^l(\bar{\Pi}); \quad S^l(\bar{\Pi}) = \mathcal{N}_c\left(\mathcal{N}_r(S^{l-1}(\bar{\Pi}))\right); \quad S^0(\bar{\Pi}) = \exp(\bar{\Pi}) \quad (2)$$

where $\mathcal{N}_r(\bar{\Pi}) = \bar{\Pi} \oslash (\bar{\Pi}\mathbf{1}_N\mathbf{1}_N^\top)$, and $\mathcal{N}_c(\bar{\Pi}) = \bar{\Pi} \oslash (\mathbf{1}_N\mathbf{1}_N^\top\bar{\Pi})$ as the row and column-wise normalization operators of a matrix, with $\oslash$ denoting the element-wise division and $\mathbf{1}_N$ a column vector of ones.

Next we move from the generation of DSMs to sampling from permutation matrices and learn the types and order of transformations. According to the Birkhoff-von Neumann theorem [3,46], the Birkhoff polytope $\mathcal{B}_N$ (which is the set of $N \times N$ DSMs), forms a convex hull for the set of $N \times N$ permutation matrices. Consequently, it is natural to think of DSMs as relaxations of permutation matrices. However, we aim at learning a $N \times K$ matrix of real logits $\Pi \in \mathbb{R}^{N \times K}$ to induce a distribution over permutation matrices of the same size, representing the assignment of N types of transformations to K layers. This matrix is not square and neither non-negative. Inspired by [4,47–49,53], we bypass that by padding Π with additional $N - K$ columns occupied by extremely negative constant values, turning it into a square matrix $\bar{\Pi}$ of size $N \times N$, whose exponent is taken in Eq. (2) to achieve non-negativity.

In practice, sampling from the Gumbel-Sinkhorn distribution of DSMs is done by applying the Sinkhorn operation for L iterations over a perturbed padded matrix and truncating the padded part,

$$\mathbf{P} = \bar{\mathbf{P}}_{1:N,1:K} \quad ; \quad \bar{\mathbf{P}} = S^L((\bar{\Pi} + G)/t) \tag{3}$$

where G is a matrix of standard i.i.d. Gumbel noise of size $N \times N$.

Note that those samples are merely close approximations of permutation matrices. As proven by [28], as the temperature t decreases and the number of Sinkhorn iterations L increases, the resulting matrix $\mathbf{P}$ gets closer to a binary permutation matrix (Fig. 3 top middle). However, this comes at the cost of vanishing gradients due to low temperature and more compute due to the many iterations [28]. To refrain from that during the search phase, the fine corrections required for conversion to binary permutation matrices are done by selecting the index of maximal value at every column of $\mathbf{P}$ (Fig. 3 bottom). Again, the ST estimator [2] is utilized for learning the logits Π by backpropagating through this discrete choice of transformation types and order, as demonstrated on the top right and top left of Fig. 3 respectively.

This effect is also present during the evaluation phase, as a finite choice of L and non-zero temperature might result in deviations from a valid permutation matrix with all rows and columns being one-hot vectors. However, in practice this rarely happens, and the repetitions of the same transformations over different augmentation layers drops significantly, as shown in Sec. 4.3 and Fig. 6.

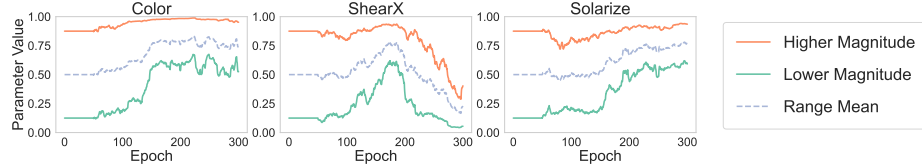## 3.3   4$^{\text{th}}$ Degree of Freedom: Magnitudes

Fig. 4: The dynamics of the learnable magnitude ranges of transformations at the first augmentation layer during the search over the CIFAR10 datasets with WRN-40-2. The lower and upper bounds of the uniform distribution are presented with solid lines and with their mean value with a dashed line.

We follow previous studies [27], which found uniform sampling of transformation magnitudes to work on par with more elaborate sampling strategies, while their range has more impact on the results. Thus, for the elementary transformation τ_i of the kth augmentation layer we set $M_{ik} \sim \text{Uniform}(\mu_{ik})$ with the ranges $\mu_{ik} = (l_{ik}, u_{ik})$ consisting of the learnable lower and upper bounds $l_{ik}, u_{ik} \in \mathbb{R}$ respectively for $i \in \{1, \ldots, N\}$ and $k \in \{1, \ldots, K\}$.

In order to learn those magnitude ranges via backpropagation, as demonstrated in Fig. 4, we utilize a differentiable implementation of the elementary

transformations [38] and the reparameterization trick [18] to backpropagate through the samples of magnitudes,

$$M_{ik} = [\sigma(h_{ik}) - \sigma(l_{ik})] \cdot \epsilon + \sigma(l_{ik}) \tag{4}$$

with $\epsilon \sim \text{Uniform}(0, 1)$ and the sigmoid function $\sigma(\cdot)$ scaling the real learnable variables into the range $(0, 1)$. We then scale M_{ik} to the accepted input range by the corresponding transformation. Some transformations are not differentiable (e.g., posterize and solarize); thus, the ST estimator is used to evaluate the gradients of their magnitude; see the supplementary materials for more details.

3.4   Policy Search as a Bilevel Optimization

Once we outlined a fully differentiable way to sample from the introduced search space of data augmentation policies via Algorithm 1, we follow a large portion of differentiable DAS methods [21, 23, 25, 27, 29, 52] and learn the underlying parameters ϕ by solving a bilevel optimization problem,

$$\min_{\phi} \mathcal{L}_{val}(\theta^*(\phi)) \tag{5}$$

$$\text{s.t.} \quad \theta^*(\phi) = \arg\min_{\theta} \mathcal{L}_{train}(\theta, \phi) \tag{6}$$

where,

$$\mathcal{L}_{val}(\theta) = \mathbb{E}_{(x,y) \sim \mathcal{D}_{val}} \mathcal{L}_{CE}\left(\mathcal{M}_\theta\left(x\right), y\right) \tag{7}$$

$$\mathcal{L}_{train}(\theta, \phi) = \mathbb{E}_{(x,y) \sim \mathcal{D}_{train}} \mathbb{E}_{\mathcal{T} \sim \mathcal{P}_\phi} \mathcal{L}_{CE}\left(\mathcal{M}_\theta(\mathcal{T}(x)), y\right) \tag{8}$$

and D_{train}, D_{val} are the train and validation splits respectively, $\mathcal{L}_{CE}$ is the cross entropy loss, $\mathcal{M}_\theta$ is the model with parameters θ, and $\mathcal{T}$ is an augmentation pipeline sampled from the augmentation policy $\mathcal{P}_\phi$ parameterized by ϕ according to Algorithm 1.

Directly solving the upper problem in Eq. (5) would require solving the lower problem in Eq. (6) for every optimization step over the upper problem. To avoid that, we revert to the single step approximation and optimize θ and ϕ alternately through stochastic gradient descent, with the update terms,

$$\Delta_{\theta_i}(\phi) = \nabla_\theta \mathcal{L}_{train}(\theta_i, \phi) \tag{9}$$

$$\Delta_{\phi_i} = \nabla_\phi \mathcal{L}_{val}(\theta_i - \eta \Delta_{\theta_i}(\phi_i)) = -\eta \nabla^2_{\theta,\phi} \mathcal{L}_{train}(\theta_i, \phi_i) \nabla_\theta \mathcal{L}_{val}(\theta_i) \tag{10}$$

Thus θ_{i+1} and ϕ_{i+1} are updated by an optimizer of choice with $\Delta_{\theta_i}(\phi_i)$ and Δ_{ϕ_i} respectively. In practice, we replace all expectations by estimates on a batch of data samples and sampled augmentations. The variance of those estimates with respect to the sampled augmentations is significantly reduced by sampling a distinct augmentation policy for each image in the batch rather than sampling once for the entire batch. This is enabled by an efficient implementation of Algorithm 1, inspired by [31] as detailed in the supplementary material. Furthermore, the computation of second order derivatives in Eq. (10) is reasonable since the size of ϕ is much smaller than the size of θ, and thus it is performed efficiently using automatic differentiation software [35, 37].

4 Experiments

Next we describe our experimental setup and evaluation process in Sec. 4.1. In Sec. 4.2 our method is compared to previous work over a variety of common natural images benchmarks, as well as a variety of several other domains in Sec. 4.2. Additionally, in Sec. 4.3 we delve into an empirical analysis of insightful aspects of our approach including ablation studies of the main components of our method. This comprehensive evaluation showcases the effectiveness of FreeAugment and provides insights into its properties.

4.1 Experimental Setup

Benchmarks We evaluate the performance of FreeAugment on three standard benchmarks; CIFAR10, CIFAR100 [20], and ImageNet-100 [45], which are all natural images. We also evaluate on the DomainNet [36] dataset, which consists of six distinct domains; real, quickdraw, infograph, sketch, painting, and clipart images, with a total of approximately 0.6 million images of 345 categories. Following [27], we use a reduced train set of 50k images for the two largest domains, real and quickdraw, and leave the rest of the samples for testing, according to the published file names of train and test images for each domain by [27]. In alignment with previous work [5, 22, 25, 27, 30, 56], Wide-ResNet40-2 and Wide-ResNet28-10 [54] architectures are used for CIFAR10/100 datasets and ResNet18 for ImageNet-100 and DomainNet [10].

Baselines Following previous works [5, 12, 56], we compare FreeAugment to a baseline based on standard augmentations: random horizontal flipping and random pad-and-crop for CIFAR-10/100, an Inception-style preprocessing [51] for ImageNet-100, and a set of augmentations used by DomainBed [7] for domain generalization on DomainNet datasets, including random horizontal flipping, color jitter (which is a mix of brightness, contrast, and saturation) and random gray-scale. Additionally, we compare FreeAugment to many other existing methods, including: AA [5], PBA [12], FastAA [22], FasterAA [8], DADA [21], DABO [23], RandAugment (RA) [6], UniformAugment (UA) [24], TrivialAugment (TA) [30], TeachAugment [42] MADAO [9] DeepAA [56], SLACK [27], DRA [52], and DDAS [25].

Search Space FreeAugment's search space is designed to enable the joint optimization of augmentation policies across all 4 DoF depicted in Fig. 1. In all of our experiments we search across augmentation policies with the maximal depth of $K = 7$. We adopt AutoAugment's space of possible transformation types, except for SamplePairing [15] and Identity. These include: ShearX, ShearY, TranslateX, TranslateY, Rotate, Solarize, Posterize, Contrast, Color, Brightness, Sharpness, AutoContrast, Invert and Equalize. All of which but the last three are associated with learnable magnitudes, whose ranges are detailed in the supplementary materials. Following conventional training practices and previous work [5,12,30], cropping, horizontal flipping, and cutout were applied by default.

Policy Search All search phases are conducted on reduced subsets of the data, each containing approximately 10% random samples: 5,000 images for CIFAR10 and CIFAR100, 10,000 images for ImageNet-100, and 6,000 images for each of the six domains in DomainNet. The search phase is conducted on the reduced datasets, with a balanced 50%-50% train-validation split. We search for 300 epochs, with a batch size of 128 and 64 for CIFAR10/100 and ImageNet-100/DomainNet, respectively. The policy distribution is initialized to be uniform over the search space. The magnitude parameters are initialized to $(0.125, 0.875)$ in the normalized range of $(0, 1)$. We use the Adam optimizer [17] for all the policy's learnable parameters, and set different learning rates of 0.02, 0.01, 1.0 for the policy's magnitude, types, and depth learnable parameters. To allow the model to learn before participating in the search, similarly to [27, 56], we use warm-up to start its training before learning the policy. Since we jointly learn 4 DoF of different impact levels, gradual warm-up periods are applied for 50, 65, and 80 epochs, respectively, for magnitude, types, and depth. Throughout the search phase, Gumbel-based distributions are scaled by a temperature parameter that is exponentially annealed from 1.0 to 0.5, and the number of Sinkhorn iterations is set to $L = 20$. These search configurations are kept the same throughout all experiments and ablations, highlighting FreeAugment's robustness to hyperparameters without necessitating additional adjustments for different datasets and architectures.

Policy Evaluation We evaluated the performance of the generated data augmentation policies across the complete CIFAR10, CIFAR100, ImageNet-100, and DomainNet datasets, replicating the exact training configurations in previous work for fair comparison. All hyperparameters are detailed in the supplementary materials to allow full reproducibility. During evaluation, the number of Sinkhorn iterations is set to $L = 20$ and the temperature for all Gumbel-based distributions was consistently set to 0.1, as there is no longer a necessity for differentiation through these. The reported top-1 test accuracy (%) for each experiment is averaged over 5 runs with different random seeds, with the 95% confidence interval denoted by $\pm$.

4.2   Comparison with Previous Approaches

CIFAR10 and CIFAR100 Table 1 provides performance evaluation of augmentation policies generated by FreeAugment on CIFAR10 and CIFAR100 for Wide-ResNet40-2 and Wide-ResNet28-10 compared to the results reported by other state-of-the-art methods. FreeAugment outperforms the sota on CIFAR10 for both architectures and on CIFAR100 for Wide-ResNet40-2, with comparable performance to many other DAS methods. Specifically, FreeAugment improves over the baseline of basic augmentations by $+\mathbf{1.05\%}$, $+\mathbf{1.18\%}$, $+\mathbf{3.3\%}$, $+\mathbf{2.39\%}$ for Wide-ResNet40-2 and Wide-ResNet28-10 on CIFAR10 and CIFAR 100 respectively.

Table 1: Top-1 test accuracy (%) on CIFAR10/100 for WRN-40-2 and WRN-28-10 as reported by the original papers. Our results are averaged over 5 random seeds, with the 95% confidence interval denoted by $\pm$.

	CIFAR10		CIFAR100	
	WRN-40-2	WRN-28-10	WRN-40-2	WRN-28-10
Baseline	95.49 ± .14	96.48 ± .15	76.74 ± .29	81.75 ± .12
AA [5]	96.3	97.4	79.3	82.9
PBA [12]	-	97.42 ± .06	-	83.27 ± .15
FastAA [22]	96.4	97.3	79.4	82.7
DADA [21]	96.4	97.3	79.1	82.5
RA [6]	-	97.3	-	83.3
TeachA [42]	-	97.5	-	83.2
UA [24]	96.25	97.33	79.01	82.82
TA(Wide) [30]	96.32 ± .05	97.46 ± .06	79.86 ± .19	**84.33 ± .17**
FasterAA [8]	96.30	97.40	78.60	82.70
DABO [29]	-	96.44	-	81.90
DeepAA [56]	-	97.56 ± .14	-	84.02 ± .18
DRA [52]	-	97.44 ± .10	-	83.85 ± .16
DDAS [25]	-	97.30 ± .10	-	83.40 ± .2
SLACK [27]	96.29 ± .08	97.46 ± .06	79.87 ± .11	84.08 ± .16
MADAO [9]	93.3	97.3	77.6	-
FreeAugment	**96.54 ± .16**	**97.66 ± .09**	**80.04 ± .23**	84.13 ± .18

ImageNet-100 Table 2 presents a comparison to previous work on ImageNet-100. All methods were reproduced with the exact training configurations of TA for ResNet-18. Since SLACK officially published four generated augmentation policies, we average the performance of all methods over four random seeds. FreeAugment outperforms the Inception-style baseline of manual choice of augmentations by **+1.78%**, and surpasses the leading DAS methods compared to by up to **+0.94%**. Notably, our approach maintained this high level of accuracy despite a reduction in batch size from 128 to 64 images in a batch during the search phase, demonstrating its effectiveness in variance reduction of gradients with respect to the expectation over the augmentation policy, see Sec. 3.4.

Table 2: Top-1 test accuracy (%) on ImageNet-100 for ResNet-18. All results are obtained following the exact same training recipe and averaged across 4 random seeds, with the 95% confidence interval denoted by $\pm$.

	Baseline	TA(Wide)	SLACK	**FreeAugment**
ResNet-18	84.84 ± .43	85.68 ± .54	86.19 ± .81	**86.62 ± .12**

Beyond Natural Images For showcasing the robustness of FreeAugment to different domains beyond natural images, we experiment with the DomainNet

dataset, comprising six distinct domains. We follow SLACK [27] and compare to DomainBed, the ImageNet version of TA(Wide) and SLACK itself. To this end, we evaluate all of them following the same evaluation setting TA's training recipe and average over four random seeds given the four published augmentation policies of SLACK. The results in Tab. 3 show FreeAugment's superior performance compared to existing DAS methods. Notably, all DAS methods outperformed the DomainBed baseline, showcasing the benefits of automatically tailoring designated data augmentation policies for each individual domain. The fact that FreeAugment yields improved results for all domains on average, and individually for each one, highlights its ability to automatically generate optimized augmentation policies across domains.

Table 3: Top-1 test accuracy (%) on DomainNet for ResNet-18. All results are obtained following the exact same training recipe and averaged across 4 random seeds, with the 95% confidence interval denoted by $\pm$.

	DomainBed	TA (Wide)	SLACK	**FreeAugment**
Real	$61.91 \pm .27$	$70.27 \pm .27$	$68.94 \pm .56$	$\mathbf{71.47 \pm .42}$
Quickdraw	$64.55 \pm .16$	$67.01 \pm .27$	$66.39 \pm .35$	$\mathbf{68.62 \pm .16}$
Inforgraph	$25.64 \pm .27$	$33.73 \pm .41$	$30.92 \pm .61$	$\mathbf{34.88 \pm .33}$
Sketch	$58.57 \pm .63$	$65.38 \pm .22$	$63.75 \pm .72$	$\mathbf{66.74 \pm .26}$
Painting	$57.70 \pm .21$	$64.13 \pm .30$	$62.26 \pm .25$	$\mathbf{64.65 \pm .24}$
Clipart	$64.07 \pm .60$	$69.77 \pm .29$	$70.92 \pm .38$	$\mathbf{71.26 \pm .26}$
Average	$55.40 \pm .16$	$61.71 \pm .12$	$60.53 \pm .21$	$\mathbf{62.93 \pm .12}$

4.3   Ablation Study and Empirical Analysis

In this section, we analyze the impact of each component of FreeAugment. We aim to demonstrate the superiority of jointly learning all four degrees of freedom, and compare the effects of employing a fixed-size augmentation policy against a soft distribution over policy depth. Our analysis also includes a comparison of two augmentation types learning methods, Gumbel-Sinkhorn and Gumbel-Softmax, to illustrate their impact on repetitive sampling of transformations, and later on the final model's performance.

Joint Learning of All Degrees of Freedom We evaluate the benefits of jointly optimizing all four DoFs by freezing each one's distribution as a uniform distribution while learning the other three. Table 4 shows the results, indicating that the joint learning of all four degrees yields superior results compared to any variant where one is held frozen.

Fixed vs. Variable Policy Depth FreeAugment samples from a learnable probability distribution over the depth of the data augmentation pipeline. Figure 5 shows the benefit of this approach compared to fixing the depth to each

Table 4: The benefit of jointly learning all DoFs. Top-1 test accuracy (%) on CIFAR100 for WRN-40-2 with different variants of FreeAugment. The results are averaged over 5 random seeds, with the 95% confidence interval denoted by $\pm$.

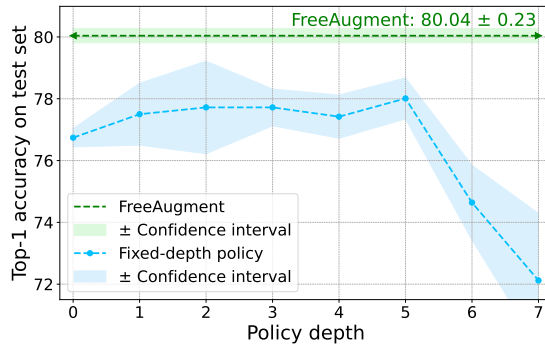| Frozen degrees of freedom	μ	Π	δ	Top-1
Frozen uniform magnitudes	✗	✓	✓	$79.64 \pm .25$
Frozen uniform types & order	✓	✗	✓	$79.54 \pm .45$
Frozen uniform depth	✓	✓	✗	$79.61 \pm .13$
Joint Learning (FreeAugment)	✓	✓	✓	$\mathbf{80.04 \pm .23}$

Fig. 5: The benefits of a variable policy depth. Top-1 test accuracy on CIFAR100 with WRN-40-2 obtained by FreeAugment compared to fixed-depth policies. Each point quantifies the average accuracy over 5 random seeds, with the shaded areas indicating the 95% confidence intervals. The horizontal line marks the performance of FreeAugment of learned variable policy depth.

of the possible depths in the search space, while all other DoFs are learned. The graph shows individual marks indicating the Top-1 Accuracy for each fixed depth on the test set of CIFAR100, and a horizontal line for FreeAugment of learned variable depth. The results are averaged across 5 random seeds, with the 95% confidence intervals in shadow. This result highlights the impact of explicitly learning the probability distribution over the number of transformations.

Mitigating the Repetition of Transformations We formulate the choice of types and order of transformations as learning the distribution over permutations via the Gumbel-Sinkhorn operation (Section 3.2), while other methods [25,27,52,56] learn the frequencies of transformations by optimizing categorical distributions separately for every layer in the data augmentation pipeline. The later approach allows sampling the same transformation at several layers, while the former mitigates this repetition of transformations, as shown in Figure 6. In this figure, we compare FreeAugment to its variant, where the Gumbel-Sinkhorn operation is replaced by a series of Gumbel-Softmax operations for sampling from independent categorical distributions across the layers. We measure the rates at which each variant samples repetitive transformations per image
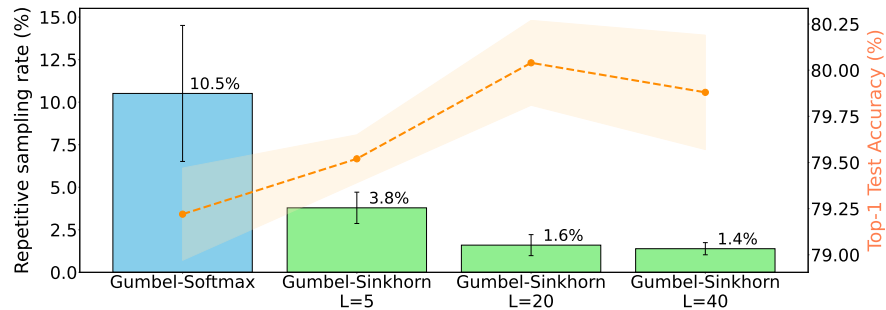
Fig. 6: Repetitive sampled transformations rates (Left Y-axis) and Top-1 test accuracy (%) on CIFAR100 (Right Y-axis) of the Gumbel-Softmax and Gumbel-Sinkhorn operations with an increasing number of Sinkhorn iterations L for the later. Repetitive transformations rates decreases significantly and accuracy increases in correspondence with the utilization of Gumbel-Sinkhorn with increasing length.

along the 10k samples in the evaluation phase of WRN-40-2 on CIFAR10 and the final Top-1 accuracy, also for different amount of Sinkhorn iterations L.

The results clearly demonstrate that incorporating Gumbel-Sinkhorn significantly reduces repetitive augmentation sampling by an order of magnitude compared to the variant utilizing Gumbel-Softmax. Increasing the number of Sinkhorn iterations L refines the approximated samples of permutation matrices and thus further reduces the total repetitions of augmentations and its variance, while increasing the accuracy. Saturated encountered at $L = 20$.

5   Conclusion

In this paper, we propose a novel formulation of the data augmentation search space that allows for the first time to jointly optimize all degrees of freedom of a data augmentation pipeline in an end-to-end manner. This is enabled by utilizing fully differentiable tools for learning the probability distributions over discrete options for the number of transformations in the pipeline, their continuous magnitudes, and their discrete types and order. We jointly learn (1) the categorical distribution over the first by utilizing the Gumbel-Softmax reparameterization trick, (2) the parameterized distribution over magnitudes by applying differentiable implementations of the transformation, and (3) the distribution over their order and types using the Gumbel-Sinkhorn operator. All are learned in an end-to-end manner by alternating gradient decent steps to solve a bilevel optimization problem, whose objective is that the augmentations make a model trained on the train set generalize better to a held-out validation set. Extensive experiments show that this effective way of optimizing the joint probability distribution over all degrees of freedom yields data augmentation policies that achieve state-of-the-art results across various benchmarks.

References

1. Adams, R.P., Zemel, R.S.: Ranking via sinkhorn propagation. arXiv preprint arXiv:1106.1925 (2011)
2. Bengio, Y., Léonard, N., Courville, A.C.: Estimating or propagating gradients through stochastic neurons for conditional computation. CoRR **abs/1308.3432** (2013), `http://arxiv.org/abs/1308.3432`
3. Birkhoff, G.: Three observations on linear algebra. Univ. Nac. Tacuman, Rev. Ser. A **5**, 147–151 (1946)
4. Cho, M., Lee, J., Lee, K.M.: Reweighted random walks for graph matching. In: Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part V 11. pp. 492–505. Springer (2010)
5. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation strategies from data. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 113–123 (2019)
6. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. pp. 702–703 (2020)
7. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. CoRR **abs/2007.01434** (2020), `https://arxiv.org/abs/2007.01434`
8. Hataya, R., Zdenek, J., Yoshizoe, K., Nakayama, H.: Faster autoaugment: Learning augmentation strategies using backpropagation. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16. pp. 1–16. Springer (2020)
9. Hataya, R., Zdenek, J., Yoshizoe, K., Nakayama, H.: Meta approach to data augmentation optimization. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 2574–2583 (2022)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
11. He, X., Chu, X.: Medpipe: End-to-end joint search of data augmentation and neural architecture for 3d medical image classification. In: 2023 IEEE International Conference on Medical Artificial Intelligence (MedAI). pp. 344–354. IEEE (2023)
12. Ho, D., Liang, E., Chen, X., Stoica, I., Abbeel, P.: Population based augmentation: Efficient learning of augmentation policy schedules. In: International conference on machine learning. pp. 2731–2741. PMLR (2019)
13. Hu, Y., Wu, X., He, R.: Tf-nas: Rethinking three search freedoms of latency-constrained differentiable neural architecture search. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16. pp. 123–139. Springer (2020)
14. Indelman, H.C., Hazan, T.: Learning randomly perturbed structured predictors for direct loss minimization. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 4585–4595. PMLR (18–24 Jul 2021), `https://proceedings.mlr.press/v139/indelman21a.html`
15. Inoue, H.: Data augmentation by pairing samples for images classification. arXiv preprint arXiv:1801.02929 (2018)

16. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. In: International Conference on Learning Representations (2016)
17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
18. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
19. Knight, P.A.: The sinkhorn–knopp algorithm: convergence and applications. SIAM Journal on Matrix Analysis and Applications **30**(1), 261–275 (2008)
20. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
21. Li, Y., Hu, G., Wang, Y., Hospedales, T., Robertson, N.M., Yang, Y.: Differentiable automatic data augmentation. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16. pp. 580–595. Springer (2020)
22. Lim, S., Kim, I., Kim, T., Kim, C., Kim, S.: Fast autoaugment. Advances in Neural Information Processing Systems **32** (2019)
23. Lin, C., Guo, M., Li, C., Wu, W., Lin, D., Ouyang, W., Yan, J.: Online hyper-parameter learning for auto-augmentation strategy. CoRR **abs/1905.07373** (2019), `http://arxiv.org/abs/1905.07373`
24. LingChen, T.C., Khonsari, A., Lashkari, A., Nazari, M.R., Sambee, J.S., Nascimento, M.A.: Uniformaugment: A search-free probabilistic data augmentation approach. arXiv preprint arXiv:2003.14348 (2020)
25. Liu, A., Huang, Z., Huang, Z., Wang, N.: Direct differentiable augmentation search. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12219–12228 (2021)
26. Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. arXiv preprint arXiv:1806.09055 (2018)
27. Marrie, J., Arbel, M., Larlus, D., Mairal, J.: Slack: Stable learning of augmentations with cold-start and kl regularization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 24306–24314 (2023)
28. Mena, G., Belanger, D., Linderman, S., Snoek, J.: Learning latent permutations with gumbel-sinkhorn networks. In: International Conference on Learning Representations (2018)
29. Mounsaveng, S., Laradji, I., Ben Ayed, I., Vazquez, D., Pedersoli, M.: Learning data augmentation with online bilevel optimization for image classification. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1691–1700 (2021)
30. Müller, S.G., Hutter, F.: Trivialaugment: Tuning-free yet state-of-the-art data augmentation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 774–782 (2021)
31. Nayman, N., Aflalo, Y., Noy, A., Zelnik, L.: Hardcore-nas: Hard constrained differentiable neural architecture search. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 7979–7990. PMLR (18–24 Jul 2021), `https://proceedings.mlr.press/v139/nayman21a.html`
32. Nayman, N., Aflalo, Y., Noy, A., Zelnik-Manor, L.: Binas: Bilinear interpretable neural architecture search. In: Khan, E., Gonen, M. (eds.) Proceedings of The 14th Asian Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 189, pp. 786–801. PMLR (12–14 Dec 2023), `https://proceedings.mlr.press/v189/nayman23a.html`

33. Nayman, N., Noy, A., Ridnik, T., Friedman, I., Jin, R., Zelnik, L.: Xnas: Neural architecture search with expert advice. Advances in neural information processing systems **32** (2019)
34. Noy, A., Nayman, N., Ridnik, T., Zamir, N., Doveh, S., Friedman, I., Giryes, R., Zelnik, L.: Asap: Architecture search, anneal and prune. In: Chiappa, S., Calandra, R. (eds.) Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 108, pp. 493–503. PMLR (26–28 Aug 2020), `https://proceedings.mlr.press/v108/noy20a.html`
35. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)
36. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1406–1415 (2019)
37. Ren*, J., Feng*, X., Liu*, B., Pan*, X., Fu, Y., Mai, L., Yang, Y.: Torchopt: An efficient library for differentiable optimization. Journal of Machine Learning Research **24**(367), 1–14 (2023), `http://jmlr.org/papers/v24/23-0191.html`
38. Riba, E., Mishkin, D., Ponsa, D., Rublee, E., Bradski, G.: Kornia: an open source differentiable computer vision library for pytorch. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3674–3683 (2020)
39. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. Journal of big data **6**(1), 1–48 (2019)
40. Sinkhorn, R.: A relationship between arbitrary positive matrices and doubly stochastic matrices. The annals of mathematical statistics **35**(2), 876–879 (1964)
41. Sinkhorn, R., Knopp, P.: Concerning nonnegative matrices and doubly stochastic matrices. Pacific Journal of Mathematics **21**(2), 343–348 (1967)
42. Suzuki, T.: Teachaugment: Data augmentation optimization using teacher knowledge. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10904–10914 (June 2022)
43. Tan, J., Nayman, N.: Two-stage kernel Bayesian optimization in high dimensions. In: Evans, R.J., Shpitser, I. (eds.) Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence. Proceedings of Machine Learning Research, vol. 216, pp. 2099–2110. PMLR (31 Jul–04 Aug 2023), `https://proceedings.mlr.press/v216/tan23a.html`
44. Tang, Z., Gao, Y., Karlinsky, L., Sattigeri, P., Feris, R., Metaxas, D.: Onlineaugment: Online data augmentation with less domain knowledge. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16. pp. 313–329. Springer (2020)
45. Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. CoRR **abs/1906.05849** (2019), `http://arxiv.org/abs/1906.05849`
46. Von Neumann, J.: A certain zero-sum two-person game equivalent to the optimal assignment problem. Contributions to the Theory of Games **2**(0), 5–12 (1953)
47. Wang, R., Yan, J., Yang, X.: Combinatorial learning of robust deep graph matching: an embedding based approach. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020)
48. Wang, R., Yan, J., Yang, X.: Graduated assignment for joint multi-graph matching and clustering with application to unsupervised graph matching network learning. Advances in Neural Information Processing Systems **33**, 19908–19919 (2020)

49. Wang, R., Yan, J., Yang, X.: Neural graph matching network: Learning lawler's quadratic assignment problem with extension to hypergraph and multiple-graph matching. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(9), 5261–5279 (2021)
50. Wang, X., Chu, X., Yan, J., Yang, X.: Daas: Differentiable architecture and augmentation policy search. arXiv preprint arXiv:2109.15273 (2021)
51. Wightman, R., Touvron, H., Jégou, H.: Resnet strikes back: An improved training procedure in timm. CoRR **abs/2110.00476** (2021), `https://arxiv.org/abs/2110.00476`
52. Xiao, A., Shen, B., Tian, J., Hu, Z.: Differentiable randaugment: Learning selecting weights and magnitude distributions of image transformations. IEEE Transactions on Image Processing **32**, 2413–2427 (2023). `https://doi.org/10.1109/TIP.2023.3265266`
53. Yew, Z.J., Lee, G.H.: Rpm-net: Robust point matching using learned features. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11824–11833 (2020)
54. Zagoruyko, S., Komodakis, N.: Wide residual networks. arXiv preprint arXiv:1605.07146 (2016)
55. Zhang, X., Wang, Q., Zhang, J., Zhong, Z.: Adversarial autoaugment. arXiv preprint arXiv:1912.11188 (2019)
56. Zheng, Y., Zhang, Z., Yan, S., Zhang, M.: Deep autoaugmentation. Proc. ICLR **1**(3), 6 (2022)
57. Zhou, K., Hong, L., Hu, S., Zhou, F., Ru, B., Feng, J., Li, Z.: Dha: End-to-end joint optimization of data augmentation policy, hyper-parameter and architecture. arXiv preprint arXiv:2109.05765 (2021)