Extrapolating from a Single Image to a Thousand Classes using Distillation

Yuki M. Asano*
Video & Image Sense Lab
University of Amsterdam

Aaqib Saeed*
Dept. of Computer Science and Technology
University of Cambridge

*Equal contribution. y.m.asano@uva.nl, as3227@cl.cam.ac.uk

Abstract

What can neural networks learn about the visual world from a single image? While it obviously cannot contain the multitudes of possible objects, scenes and lighting conditions that exist – within the space of all possible $256^3 \cdot 224 \cdot 224$ 224-sized square images, it might still provide a strong prior for natural images. To analyze this hypothesis, we develop a framework for training neural networks from scratch using a single image by means of knowledge distillation from a supervised pretrained teacher. With this, we find that the answer to the above question is: ‘surprisingly, a lot’. In quantitative terms, we find top-1 accuracies of 94%/74% on CIFAR-10/100, 59% on ImageNet and, by extending this method to audio, 84% on SpeechCommands. In extensive analyses we disentangle the effect of augmentations, choice of source image and network architectures and also discover “panda neurons” in networks that have never seen a panda. This work shows that one image can be used to extrapolate to thousands of object classes and motivates a renewed research agenda on the fundamental interplay of augmentations and images. Webpage:1

1. Introduction

Deep learning has both relied and improved significantly with the increase in dataset sizes. In turn, there are many works that show the benefits of dataset scale in terms of data points and modalities used. Within computer vision, these models trained on ever larger datasets, such as Instagram-1B [40] or JFT-3B [18], have been shown to successfully distinguish between semantic categories at high accuracies. In stark contrast to this, there is little research on understanding neural networks trained on very small datasets.

Why would this be of any interest? While smaller dataset sizes allow for better understanding and control of what the model is being trained with, we are most interested in its ability to provide insights into fundamental aspects of learning: For example, it is an open question as to what exactly is required for arriving at semantic visual representations from random weights, and also of how well neural networks can extrapolate from their training distribution.

While for visual models it has been established that few or no real images are required for arriving at basic features, like edges and color-contrasts [1, 10, 29, 47], we go far beyond these and instead ask what the minimal data requirements are for neural networks to learn semantic categories, such as those of ImageNet. This approach is also motivated by studies that investigate the early visual development in infants, which have shown how little visual diversity babies are exposed to in the first few months whilst developing generalizeable visual systems [3, 49]. In this paper, we study this question in its purest form, by going to one extreme and analyzing whether neural networks can learn to extrapolate from a single datum.

However, addressing this question naively runs into the...
difficulties of i) current deep learning methods, such as SGD or BatchNorm being tailored to large datasets and not working with a single datum and ii) extrapolating to semantic categories requiring information about the space of natural images beyond the single datum. In this paper, we address these issues by developing a simple framework that recombines augmentations and knowledge distillation.

First, augmentations can be used to generate a large number of variations from a single image. This can effectively address issue i) and allow for evaluating the research question on standard architectures and datasets. This use of data augmentations to generate variety is drastically different to their usual use-case in which transformations are generated to implicitly encode desirable invariances during training.

Second, to tackle the difficulty of providing information about the “rest of the world” in the single datum setting, we opt to use the outputs of a supervisedly trained model in a knowledge distillation (KD) fashion. While KD [28] is originally proposed for improving small models’ performance by leveraging what larger models have learned, we re-purpose this as a simple way to provide a supervisory signal about semantic classes into the training process.

We combine the above two ideas and provide both student and teacher only with augmented versions of a single datum, and train the student to match the teacher’s imagined class-predictions of classes – almost all of which are not contained in the single datum, see Fig. 1. With this method, we are not interested in any practical applications. Instead the method merely serves as a means to analyzing the fundamental question of how well neural networks trained from a single datum, such as a single image, or a single audio clip, can extrapolate to semantic classes, for instance those of CIFAR, SpeechCommands or ImageNet.

Despite the fact that the resulting model has only seen a single datum plus augmentations, we find surprisingly high quantitative performances of e.g., 74% on CIFAR-100, 84% on SpeechCommands and 59% top-1, single-crop accuracy on ImageNet-12. We further find that our method benefits from high-capacity student and low-capacity teacher models, and that the source datum’s characteristics matter; random noise or less dense images yield much lower performances than dense pictures, such as the one shown in Figure 1. In summary, in this paper we make these four main contributions:

1. A minimal framework for training neural networks with a single datum from scratch using distillation.
2. Extensive ablations of the proposed method, such as the dependency on the source image, the choice of augmentations and network architectures.
3. Large scale empirical evidence of neural networks’ ability to extrapolate on >12 vision and audio datasets.
4. Qualitative insights on what and how neural networks trained with a single image learn.

2. Related Work

The work presented builds on top of insights from the topics of knowledge distillation and single- and no-image training of visual representations and yields insights into neural networks’ ability to extrapolate.

Distillation. In knowledge distillation (KD), the goal is to train a typically lower capacity student model from a pretrained teacher model in order to surpass the performance of solely training with a label-supervised objective. Specifically, KD has been explored extensively to train more performant and, or compressed student models from the soft-target predictions of teacher models [2, 16, 20, 22, 28]. Similarly, other approaches have also been developed to improve transfer from teacher to student, including sharing intermediate layers’ features [56], spatial attention transfer [77], similarity preservation between activations of the networks [65], and contrastive distillation [63]. KD has also been shown to be an effective approach for learning from noisy labels [35].

More recently, [6] conducted a comprehensive empirical investigation to identify important design choices for successful distillation from large-scale vision models. In particular, they show that long training schedules, paired with consistent augmentations (including MixUp) for both student and teacher, result in better performances.

Without the training data. Distilling knowledge without access to the original training dataset was originally proposed in 2017 [37]. Yet, this paradigm is gaining importance as many recent advancements have been made possible due to extremely large proprietary datasets that are kept private. This has lead to either the sole release of trained models [21, 40, 52] or even more restricted access to only the model outputs via APIs [9]. While the original work [37], still required activation statistics from the training dataset of the network, more recent works do not require this “meta-data”. These approaches are typically GAN based [13, 41, 74], for example generating datasets of synthetic images that maximally activate neurons in the final layer of teacher. Similarly, in [75], features of pretrained networks are inverted, a large dataset generated and then used for distillation with high accuracies.

Related to this, there are works which conduct “dataset” distillation, where the objective is to distill large-scale datasets into much smaller ones, such that models trained on it reach similar levels of performance as on the original data. These methods generate synthetic images [11, 36, 53, 69, 79], labels [8], or both [42], but due to their meta-learning nature have only been applied to small-scale datasets, such as MNIST or CIFAR. Finally, the ability to copy models based on API outputs while having access to a dataset with only a partial overlap to the original one has been explored.
in [48,68], though [48] relies on pretrained models.

In contrast to GAN- and inversion-based methods, as well as to dataset distillation, our approach does not require the knowledge of the weights and architecture of the teacher model, and instead works with black-box “API”-style teacher models and much smaller “datasets” of just a single datum plus augmentations. Compared to model copying, we do not assume access to any meaningfully similar training dataset, nor start with a pretrained model. Compared to both, this paper follows a radically different goal: We are instead interested in analysing the potential of extrapolating from a single image to the manifold of natural images, for which we choose KD as a well-suited and simple tool.

**Neural network extrapolation.** Work on extrapolation behavior mostly focuses on MLPs [5,23,27] and GNNs [73]. For vision models, only model generalization behavior is analysed, e.g., in [54,57,81], however this confounds training data similarity with re-usability of learned features. In this work, we instead investigate the extrapolation performance of vision models in isolation – to the best of our knowledge, for the very first time.

**Single and no-image training of visual representations.** Several prior works propose to learn representations with a single training example, such as a single image, while focusing on specific tasks, such as super resolution or inpainting [59,66], generative modelling [58] and tracking [55]. In self-supervised learning, a single image was shown to be enough for pretraining low-level visual representations when combined with strong data augmentations [1]. Related to this, [29] pretrained deep visual models with images of fractals and [4] with uninitialized GAN outputs. While these lines of work provide a compelling argument and motivation for this paper by demonstrating the potential of augmentations and limited data, our work is both complementary and goes beyond unsupervised pretraining by showing that KD allows us to learn semantic categories, such as those of ImageNet, from one image.

### 3. Method

We believe that simplicity is the key for demonstrating and analysing the question of how far a single image can take us. Any bells-and-whistles will surely improve performance, but would distract from the science. Inspired by recent work of Asano et al. [1], which "patchifies" a single-image using augmentations, and the knowledge distillation method presented by Beyer et al. [6], our technical contribution lies in unifying them to develop a single-image distillation framework.

#### i. Dataset generation.

In [1], a single image is augmented many times to generate a static dataset of fixed size. This is done by applying the following augmentations in sequence: cropping, rotation and shearing, and color jittering. We follow their official implementation², and do not change any hyperparameters except for the final crop size, for which we choose 32 for CIFAR-based experiments and 256 for our large-scale experiments. In addition, we also conduct experiments on audio classification. Audio is typically represented by spectrograms, which convert the sound wave signal to a single-channel intensity image with axes that represent time-passage and frequency. For generating a dataset of augmented audio-clips, we apply the set of audio-augmentations from [7], which consist of operations, such as random volume increasing, background noise addition and pitch shifting (see Appendix for the complete list) to yield log-Mel spectrograms from raw waveforms. It is important to note that we compute these spectrograms on-the-fly during training, which can be seen as images (see Figure 11 in Appendix). While all augmentations can be generated online and on a GPU, we follow [1] in saving the initially augmented patches as distinct datasets to ensure reproducibility, and have a simple measure of epochs. The implementation details of patch generation and spectrogram computation are provided in Appendix C.

#### ii. Knowledge distillation.

The knowledge distillation objective is proposed in [28] to transfer the knowledge of a pretrained teacher to a lower capacity student model. In this case, the optimization objective for the student network is a weighted combination of dual losses: a standard supervised cross-entropy loss and a “distribution-matching” objective that aims to mimic the teacher’s output. However, in our case there are no class-labels for the patches generated from a single image, so we solely use the second objective formulated as a Kullback–Leibler (KL) divergence between the student output $p_s$ and the teacher’s output $p_t$:

$$
L_{KL} = \sum_{c \in C} -p^t_c \log p^s_c + p^s_c \log p^t_c
$$

where $c$ are the teachers’ classes and the outputs of both student and teacher are temperature-flattened probabilities, $p = \text{softmax}(l/\tau)$, that are generated from logits $l$ and a temperature $\tau$.

For training, we follow [6] in employing a function matching strategy, where the teacher and student models are fed consistently augmented instances, that include heavy augmentations, such as MixUp [78] or CutMix [76]. However, in contrast to [6], we neither have access to TPUs nor can train 10K epochs on ImageNet-sized datasets. While both of these would likely improve the quantitative results, we believe that this handicap is actually blessing in disguise: This means that the results we show in this paper

---

²https://github.com/yukimasano/single_img_pretraining
are not specific to heavy-compute, or extremely large batch-sizes, but instead are fundamental.

4. Experiments

4.1. Implementation details

Source data. For the source data, we utilize the single images of [1], except for one image, which we replace by a similar one as we could not retrieve its licence. In addition, we collect one satellite image of an airport for testing performance in a different remote-sensing domain. For the audio experiments, we use two short audio-clips from Youtube, a 5mins BBC newsclip, as well as a 5mins clip showing 11 Germanic languages. Images and audio visualizations, sources and licences are provided in Appendix A.

Tasks. For simplicity, we focus on classification tasks. For our ablations and small-scale experiments we focus on CIFAR-10/100 [32] and also on the different domain of satellite image classification with the EuroSAT dataset [25]. For larger-scale experiments using 224 × 224 sized images we evaluate our method on datasets with varying number of classes (see Table 8): STL-10 [14], Pets [51], ImageNet-100 [63], Flowers [44], Places365 [80] and finally ImageNet LSVRCl2 (IN-1k) [17]. For our audio experiments in Table 7, we use MUSAN [60], Voxforge [39], Speech-Commands [70], and LibriSpeech [50] datasets for varying complexity and categories of sounds to be recognized.

Optimization. For each of these datasets, we first train a teacher network in a usual supervised manner, that is then used for the distillation. For the distillation to the student model, we use a temperature \( \tau = 8 \) motivated by findings in [6]. We keep this temperature fixed throughout the whole of the paper due to limited compute. For optimization, we use AdamW [38]. Further implementation details are provided in Appendix C.

4.2. Ablation

1-image vs full training set. We first examine the capability to extrapolate from one image to small-scale datasets, such as CIFAR-10 and CIFAR-100. In Table 1, we compare various datasets for distilling a teacher model into a student. We find that while distillation using the source dataset always works best (95.26% and 78.06%) on CIFAR-10/100, using a single image can yield models which almost reach this upper bar (94.1% and 73.8%). Moreover, we find that one image distillation even outperforms using 10K images of CIFAR-10 when teaching CIFAR-100 classes even though these two datasets are remarkably similar. To better understand why using a single image works, we next disentangle the various components used in the training procedure: (a) the source image, (b) the generated image dataset size and (c) the augmentations used during distillation, corresponding to Tables 2a to 2c.

1-image training parameters. From Table 2a, we find that the choice of source image content is crucial: Random noise or sparser images perform significantly worse compared to the denser “City” and “Animals” pictures. This is in contrast to self-supervised pretraining analyzed in [1], suggesting that the underlying mechanisms are different. Next, from Table 2b, we find that adding more variation to the initial set of crops improves performance in a progressive manner, where the first 10x in the number of initial crops adds 4.9% in CIFAR-100 performance and the last one only adds 0.9%. While out of scope for this paper, this suggest that there is still room to optimize the initial set of crops used to generate this dataset. Finally in Table 2c, we ablate the augmentation we use during knowledge distillation. Besides observing the general trend of “more augmentations are better”, we find that CutMix performs better than MixUp on our single-image distillation task. This might be because in our case the model is tasked with learning how to extrapolate towards real datapoints, while MixUp is derived and therefore might be more useful for interpolating between (real) data samples [6, 78].

1-image vs other datasets. In Table 4, we find that there is something unique about using a single image, as our method outperforms several synthetic datasets, as well as the GAN-based approach of [41]. This is despite the fact that these synthetic datasets contain ~50K images. We provide insights on the characteristics of this 1-image dataset in Section 4.3.3.

Teacher signal. In Table 3, we also ablate the type of learning signal the student receives. We find that even if the student receives only the top-5 predictions, or even just the argmax prediction (i.e., a hard label) of the teacher without any confidence value, the student is still able to extrapolate to CIFAR images at a significant level. While these datasets are small, this might suggest copying larger models from API outputs is possible, similar to [48].

Next, using the insights gained in this section, we scale the experiments towards other network architectures, dataset domains and dataset sizes.
only around one-percentage point lower than supervised original source data; achieving accuracy of 94% on WideR40-4 on CIFAR-10 is very close to distilling from We find that the distillation performance of WideR40-4 to be attributed to the lower capacity of the student model. Except for the case of ResNet-56 to ResNet-20 which could find that almost all architectures perform similarly well, except for CIFAR-10 and CIFAR-100. On CIFAR-10, we experiment with different common architectures on CIFAR-10. We analyze key components in our experimental setup. We report top-1 accuracies for C10 and C100.

Table 2. Single image distillation. We analyze key components in our experimental setup. We report top-1 accuracies for CIFAR-10/100. For the teacher we use WideR40-4 and for the student a WideR16-4 and train for 1K epochs.

Table 3. Teacher signal. Even with only top-5 predictions (class & probability) or hard distillation, performance only slightly degrades. Same setup as in Table 2.

Table 4. Comparison against other datasets on CIFAR-10. Here the teacher model is WideR40-2 and student is WideR40-1 for comparability to [41].

Table 5. Distilling various architectures on CIFAR-10. We compare student accuracy when distilling with full training set vs our 1-image dataset.

4.3. Scalability

4.3.1 Architectures

In Table 5, we experiment with different common architectures on CIFAR-10 and CIFAR-100. On CIFAR-10, we find that almost all architectures perform similarly well, except for the case of ResNet-56 to ResNet-20 which could be attributed to the lower capacity of the student model. We find that the distillation performance of WideR40-4 to WideR40-4 on CIFAR-10 is very close to distilling from original source data; achieving accuracy of 94.14% and only around one-percentage point lower than supervised training. This analysis shows that our method generalizes to other architectures hints that larger capacity student models might be important for high performances.

4.3.2 Data domain

Satellite images. We investigate how the proposed single-image distillation performs on varying the data domain. To this end, we focus on satellite imagery classification following the same experimental protocol as before. In Table 6, we first report distillation performance when using EuroSAT as the source data for CIFAR-100 and vice versa. The performance on the latter drops significantly to 9.3% when input domain changes as these datasets are largely unrelated and EuroSAT is less diverse. Next, we perform distillation with a single-image datasets generated from an ‘Animal’ and ‘Satellite-picture’ images. To our surprise, we notice significantly high accuracy of 95% even when using completely unrelated data, while combining the data from varying sources equally slightly improves the distillation performance on EuroSAT with only two-percentage point lower accuracy on CIFAR-100. From this evaluation, we can conclude that our distillation framework works equally well on different data domains, and being strictly in-domain is not necessary as long as the image is diverse.

Audio classification. So far, our proposed approach has shown its success in distilling useful knowledge for images. To further test the generalizability of our approach on other modalities, we conduct experiments on several audio recognition tasks of varying difficulty. We perform distillation via 50K randomly generated short audio clips from two

Table 6. Distilling dataset. Combining single images works across domains. The teacher is a WideR40-4 and the student a WideR16-4 network.
(i.e., Ours_A and Ours_B), 5mins YouTube videos (see Appendix A for more details). In Table 7, we compare the results against the performance of the teacher model and distilling directly using source dataset. We find that even for audio, distilling with merely a single audio clip’s data provides enough supervisory signal to train the student model to reach above 80% accuracy in the majority of the cases. In particular, we see significant improvement in distillation performance when a single audio has a wide variety of sounds (including speakers and background noises) to boost accuracy on challenging Voxforge dataset from 72.8% to 78.4%. Our results on audio recognition tasks demonstrate the modality agnostic nature of our approach and further highlight the capability to perform knowledge distillation in the absence of large amount of data.

### 4.3.3 Dataset and image size

In this section, we scale our experiments to large vision models, and larger-scale image datasets. We show our results in Table 8. We find that, overall, using the setting in Table 8, a single image is not enough to recover the full performance on these more difficult datasets. This might be to a large degree because of the fine-grained nature of these datasets, e.g., ImageNet includes more than 120 kinds of Extrapolating from a single image to these very similar categories might require a more diverse source image. Nevertheless, we find a surprisingly high accuracy of 53.5% on ImageNet’s validation set, even though this dataset comprises 1000 classes, and the student only having seen heavily augmented crops of a single image. In Table 9, we conduct further analyses on distilling ImageNet models.

#### Patchification vs more data

At the top part of the Table 9, we analyse the effect of using original vs patchified random images from the training set of ImageNet. We find that while for 10 and 100 source images, patchification improves performance, this is not the case when a larger number of images is used. This indicates that increased diversity is especially a crucial component for the small data regime, while capturing larger parts of objects becomes important only beyond this stage. We also make the surprising observation that patches from a single high-quality image (“City”) obtain better performances than patches generated from 1000 ImageNet training images. To understand why this might be the case, we note that the high coherency of training patches from a single image strikingly resembles what babies see during their early visual development, e.g., looking at only few toys and people but from many angles [3,49]. It is hypothesized that this unique combination of coherence and variability is ideal for learning to recognize objects [49].

To make this more concrete, we compare low-level GIST [46] features of our 1-image dataset against those computed from visual inputs of a toddler in [3] (see Appendix C.5 for details). As we do not have access to the dataset of [3], we copy their Figure 3b for reference in Fig. 2b. While the GIST distances for ImageNet are very different with a mean around 0.75 [3], we find that our single image dataset’s distribution in Fig. 2a closely resembles that of the visual inputs of a toddler. While this is one possible explanation for why this type of data might work well for developing visual representations, further research is still required.
The self-supervised variant. A similar trend is observed for ImageNet and Places, respectively, vastly outperforming the ground-truth teacher. In particular, we can train a ResNet with only a single image and outputs of a pretrained teacher model, it is natural to ask how well the student model exhibits a much broader distribution of different settings are those in which the student’s capacity is higher than that of the teacher. In particular, we can improve from 32.7% top-1 accuracy to 52.0% when switching the architecture of student and teacher. We find that by using a R18→WR50-2 setup, we obtain our highest performance of 59.1% on ImageNet-12, which – though it uses a different architecture and a teacher – surpasses the groundbreaking AlexNet [33] performance of 56.5%.

**Teacher-student architectures.** In the bottom half of Table 9, we vary the different teacher and student architectures and find surprising results: First, in contrast to normal KD, we find that the teacher’s performance is not directly related to the final student performance. A case in point is the model R50*, which yields the worst student performances, likely because the teacher was trained to ignore heavy RandAugment [15] augmentations. Second, we find that the best settings are those in which the student’s capacity is higher than that of the teacher. In particular, we can improve from 32.7% top-1 accuracy to 52.0% when switching the architecture of student and teacher. We find that by using a R18→WR50-2 setup, we obtain our highest performance of 59.1% on ImageNet-12, which – though it uses a different architecture and a teacher – surpasses the groundbreaking AlexNet [33] performance of 56.5%.

**Representation learning experiments.** Given that we can train a ResNet with only a single image and outputs of a pretrained teacher model, it is natural to ask how well this model serves as a general visual encoder. For this, we conduct a number of standard representation learning evaluations in Table 10 and also compare against running a common self-supervised learning method, MoCo-v2 [24], on the same dataset (see Appendix C for details). We find that freezing the backbone and only training a linear layer on top of it, can achieve performances of 66.1% and 47.1% for ImageNet and Places, respectively, vastly outperforming the self-supervised variant. A similar trend is observed for data-efficient classification using an SVM for Pascal VOC and Places, with performances of more than double that of MoCo-v2.

### 4.4. Analysis

To get a deeper understanding of how the network is achieving these performances, we next analyse the learned models.

**Output confidence scores.** In Fig. 3, we visualize the distribution of the confidence scores of the predicted classes for student and teacher models for the training and evaluation data. We observe that the student indeed learns to mimic the teacher’s predictions well for the patch training data. However, during evaluation on CIFAR images, the student model exhibits a much broader distribution of values, some with extremely high confidence scores. This might indicate that the student has learned to identify many discriminative features of object classes from the training data, which are all triggered when shown real examples, leading to much higher confidence values.

**Feature space analysis.** Next, we analyse the network’s embedding of training patches when compared to validation-set inputs. For this, we fit a t-SNE [67] using the features of 5K training patches and 5K test set images in Fig. 15. We find that, as expected, the training patches mapped to individual CIFAR classes do not resemble real counter parts. Moreover, we observe that all the training patches are embedded close to each other in the center, while CIFAR images are clustered towards the outside. This shows that the network is indeed learning features that are well suited for effective extrapolation.

**Per-class evaluations.** We now turn to models trained to predict IN-1k classes and analyze per-class performances against training time. In Fig. 5, we find that the validation performance of IN-1k classes is very diverse: some are identified with > 80% accuracy after just a few epochs of training, such as the class “Black Swan”, while others require longer training and some are almost impossible to learn, even after 200 epochs. In Fig. 6, we plot the validation performance against the frequency of how often the

---

**Table 10. Representation learning performance.** We compare our distillation approach (R50→R50) against self-supervised pre-training via MoCo-v2 on the same data. We report: linear eval. accuracy on IN-1k and Places; accuracy when finetuning and using 1% of ImageNet with labels; SVM classification mAP on PVOC07 and accuracy on Places with varying numbers of images per class.

| Model              | IN-1k   | Places | IN-1k (1%) | PVOC   | Places |
|--------------------|---------|--------|------------|--------|--------|
|                   | ~1K     | ~10K   | ~13        | 4      | 4      |
| 1-img + MoCo-v2   | 28.5    | 28.8   | 14.4       | 17.2   | 6.5    |
| 1-img + Distill   | 66.1    | 47.1   | 65.2       | 59.4   | 20.6   |
| 1-img + MoCo-v2   | 28.5    | 28.8   | 14.4       | 17.2   | 6.5    |
| 1-img + Distill   | 66.1    | 47.1   | 65.2       | 59.4   | 20.6   |

**Figure 2.** Pair-wise distances of $L_2$-normalized GIST features for 10K images from our 1-image training dataset used for our ImageNet experiments.

**Figure 3.** Output confidence scores. Temperature scaled softmax scores of the predicted classes for 5K patches and the 5K CIFAR-10 validation set images are shown.
class appears as a top-1 prediction of the teacher during 1 epoch of training and find that it is unrelated, showing that the teacher is being very “creative” and the student is profiting heavily from the knowledge contained in the soft-predictions, echoing findings from [20, 28].

Neuron visualizations. Finally, in Fig. 7, we visualize four final-layer neurons using the Lucid [45] library. When we compare neurons of the standard ImageNet supervised and our distilled model, we find that the neurons activate for very similar looking inputs. The clear neuron visualizations for “panda” or “lifeboat” (and more are provided in the Appendix B.2) are especially surprising since this network was trained using only patches from the “City” Image, and has never seen any of these objects during its learning phase.

5. Discussion

Limitations. In this paper, we were mainly concerned with showing that extrapolating from a single image works, empirically. While we do conduct several quantitative and qualitative analyses, we do not provide a theoretical framework but instead believe that theory will follow experimental evidence. Due to the limited nature of our compute resources, we have also not extensively analyzed the choice of initial image or patchification augmentations, instead relying on insights from previous works.

Potential negative societal impact. While our research question is of fundamental nature, one possible negative impact could be that the method is used to steal models and thus intellectual property from API providers – although in practice the performance varies widely especially for large-scale datasets (see Table 8).

Conclusion and outlook. In this work, we have analyzed whether it is possible to train neural networks to extrapolate to unseen semantic classes. Our quantitative and qualitative results demonstrate that our novel single-image knowledge distillation framework can indeed enable training networks from scratch to achieve high accuracies on several architectures, datasets and domains. While this paper has set out to answer the question posed in the abstract, in so doing it has raised several further research questions, such as the dependency of the source image and the target semantic classes; how networks combine features for extrapolation; and the role and informational content of augmentations; all of which we hope inspires further research.
[67] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 7

[68] Dongdong Wang, Yandong Li, Liqiang Wang, and Boqing Gong. Neural networks are more productive teachers than human raters: Active mixup for data-efficient knowledge distillation from a blackbox model. In *CVPR*, pages 1498–1507, 2020. 3

[69] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 2

[70] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018. 4, 6

[71] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021. 6

[72] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, pages 3–19, 2018. 6

[73] Keyulu Xu, Mozhi Zhang, Jingling Li, Simon Shaolei Du, Ken-Ichi Kawarabayashi, and Stefanie Jegelka. How neural networks extrapolate: From feedforward to graph neural networks. In *ICLR*, 2020. 3

[74] Jingwen Ye, Yixin Ji, Xinchao Wang, Xin Gao, and Mingli Song. Data-free knowledge amalgamation via group-stack dual-gan. In *CVPR*, pages 12516–12525, 2020. 2

[75] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deep-inversion. In *CVPR*, pages 8715–8724, 2020. 2

[76] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pages 6023–6032, 2019. 3, 4, 7

[77] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016. 2

[78] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 3, 4

[79] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *ICML*, 2021. 2

[80] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *PAMI*, 2017. 4, 6

[81] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *arXiv preprint arXiv:2103.02503*, 2021. 3
Appendix

Table of Contents

A. Additional Visualizations 1
   A.1 Input images .................................. 1
   A.2 Training patches and predictions ............ 3

B. Additional Experiments 3
   B.1 Video action recognition ....................... 3
   B.2 Neuron activation maximizations .............. 4
   B.3 t-SNE of feature space ......................... 4
   B.4 Centered Kernel Alignment comparison 5

C. Implementation Details 5
   C.1 Initial patch generation ....................... 5
   C.2 Computing log-Mel spectrograms ............. 6
   C.3 Audio neural network architecture .......... 6
   C.4 Training ..................................... 6
   C.5 GIST features comparison ................... 7
   C.6 Representation learning experiments ....... 7

A. Additional Visualizations

A.1. Input images

Image sources and licences. From top to bottom, the images in Fig. 8 have the following licences and sources: (a): CC-BY (we created it), (b) public domain (NASA) (c) CC BY-SA 3.0 (by David Ball), (d) pixabay licence (personal and commercial use allowed), (e) personal use licence.

Images (b), (d) and (e) are identical to the ones in the repo of [1], while (a) we have recreated on our own and (c) is a replacement for a similar one whose licence we could not retrieve.

---

(a) The “Noise” Image. From uniform noise [0,255]. Size: 2,560x1,920, PNG: 16.3MB.

(b) The “Universe” Image. Size: 2,300x2,100, JPEG: 7.2MB.

(c) The “Bridge” Image. Size: 1,280x853, JPEG: 288KB.

(d) The “City” Image. Size: 2,560x1,920, JPEG: 1.9MB.

(e) The “Animals” Image. Size: 1,300x600, JPEG: 267KB.

---

Figure 8. Single Images analysed. Here we show the images analysed in Table 1.
**Audio source.** For the experiment on audio representation distillation, we use a 5.5mins English news-clip taken from BBC about how can Europe tackle climate change and a 5mins clip about 11 Germanic languages.

A.2. Training patches and predictions

![ImageNet training data](image)

**Figure 9. ImageNet training data.** Here we visualize the augmented training data used for training along with the temperaturer-scaled Top-5 predictions of the ImageNet pretrained ResNet-50 teacher.

In Fig. 9, we show several training patches as they are being used during training along-side the teacher-network’s temperature-adjusted predictions. Only very few training samples can be interpreted (e.g. 3rd row: “nail” or “slide rule”), while for most other patches, the teacher is being very “creative”.

In Fig. 10, we show a similar plot for the training patches used in CIFAR-10 training. Here we show the whole teaching signal, and highlight the top-5 predictions of the teacher for visualization.

Finally, in Fig. 11, we show some some augmented training samples for the audio classification experiment. This figure shows how various spectrograms can be generated from a single clip by utilizing many augmentations.

---

7https://www.youtube.com/watch?v=nZgD4iPapVo
8https://www.youtube.com/watch?v=lq2qTETBXM

B. Additional Experiments

B.1. Video action recognition

We conduct further experiments on video action recognition tasks. For this we use the common UCF-101 [61] dataset, more specifically the first split. As distillation data,
we generate simple source videos of 12 frames that show a linear interpolation between two crops of the City image. We generate 200K of these videos and apply AugMix [26] and CutMix [76] during the distillation for additional augmentations. For the architectures we use the recent, state-of-the-art X3D architectures [19] as they obtain strong performances and allow for flexible scaling of network architectures. We use a X3D-XS model as the teacher, which is trained with inputs of $160 \times 160$ and 4 frames and a temporal subsampling factor of 12. For the student model we scale this X3D-XS model in terms of width and depth by increasing the depth factor and width factor from 2.2 to 3 and 4 respectively, yielding a network with 18.1M parameters compared to 3.12M. For UCF-101 we finetune the teacher model starting from supervised Kinetics-400 pre-training, which achieves a 90.4% performance. We use a batch size of 128 per GPU on two GPUs, a learning rate of $10^{-3}$, a temperature of 5, cosine learning rate schedule with a warmup of 5 epochs, with no weight decay and no dropout. As this experiment takes even longer than the image ones, we have not conducted systematic ablations on these parameters. Instead, we merely picked ones which looked promising after a few epochs but believe that even this small experimental evidence is enough to investigate to what extent these 1-image “fake-videos” can be used for learning to extrapolate.

The results are given in Table 11. We observe a performance of over 72% on UCF-101 using just a single fake-video as the training data along with the pretrained teacher. While this number is far below the state of the art or the teacher’s performance, it for example outperforms CLIP’s 69.8% zero-shot performance with a ViT-B16 model and shows that the student is able to extrapolate to the action classes of UCF from just one datum.

### B.2. Neuron activation maximizations

In Fig. 13, we show further class-neuron activation maximization visualizations for the $R50 \rightarrow R50$ distillation experiment and compare them to the corresponding ones of the supervised teacher model. Similar to the examples in the main paper, we find that the neurons visualizations are remarkably similar.

In Fig. 14, we show further visualizations of the trained student model from the $R50 \rightarrow R50$ distillation experiment. We show five randomly chosen neurons for every layer in the ResNet.

### B.3. t-SNE of feature space

We visualize representations of distilled student model on CIFAR-10 validation set in order to highlight semantic relevance of the features in Fig. 15. We extract the feature from penultimate layer of WideR16-4 model for computing 2-D t-SNE embeddings. The visualization clearly highlights that distilled features contain substantial amount of semantic information to correctly differentiate between different object categories.
Figure 14. **Visualizing intermediate neurons** via activation maximization. We visualize five randomly picked neurons for each layer and show their maximally responding input images.

Figure 15. **t-SNE embeddings of CIFAR-10**. We visualize t-SNE embeddings of a WideR16-4 model distilled from WideR40-4 with random patches generated from an ‘Animal’ image.

### B.4. Centered Kernel Alignment comparison

In Fig. 16, we visualize similarity between different layers of the models trained in a general supervised manner and with distillation using patches of 1-image for CIFAR-10 and ImageNet datasets. We use centered kernel alignment [31] method to identify the resemblance of features learned with these different training approaches. We notice high similarity in representations, which reinforce our empirical evaluation that the student models indeed learn useful features required for differentiating between semantic categories.

#### Representation similarity with CKA [31]

We visualize the similarity between supervised representations and those distilled using random patches of 1-image. As in [43], we compute CKA for all the layers in a model using CIFAR-10 testset. The layers in the same block group shows high similarity though both models are trained in different ways, indicating that distilled features are highly similar to those learned in a supervised way.

#### C. Implementation Details

All code will be released open-source and is attached as supplementary information to this submission.

#### C.1. Initial patch generation

We do not tune the individual patch generation and instead adapt the procedure directly from [1]\(^9\). The augmentations applied to the input image of size \(H \times W\) to arrive at individual patches of size \(P \times P\) is (in order) as follows:

1) \(RC(\text{size}=0.5\times\min(H,W))\)
2) \(RRC(\text{size}=(1.42\times P), \text{scale}=(2e^{-3}, 1))\)
3) \(\text{RandomAffine}(\text{degrees}=30, \text{shear}=30)\)
4) \(\text{RandomVFlip}(p=0.5)\)
5) \(\text{RandomHFlip}(p=0.5)\)
6) \(\text{CenterCrop}(\text{size}=(P,P))\)

\(^9\)https://github.com/yukimasano/single_img_pretraining
7) ColorJitter(0.4,0.4,0.4,0.1, p=0.5)

All transformations are standard operations in PyTorch: RC stands for RandomCrop, i.e. taking a random crop in the image with a specific size; RRC for RandomResizedCrop, i.e. taking a random sized crop withing the size specified in the scale tuple (relative to the input); RandomAffine for random affine transformations (rotation and shear); RandomVFlip and RandomHFlip for random flipping operations in the vertical and horizontal direction, and CenterCrop crops the image in the center to a square image of size $P \times P$. Finally, ColorJitter computes photometric jitering, where the parameters for the strengths are given in the order of brightness, contrast, saturation and hue and applied with a certain probability.

Similarly, for audio clips generation given a single audio, we use augmentation operations from [7] with default settings. Specifically, to create a single example we apply the following procedure: we randomly crop a segment of 2-seconds, and use randomly sample an augmentation function to create transformed instances and save them in mono format. In our work, we use these augmentations, all with their default settings:

- 1) add-background-noise
- 2) apply-lambda
- 3) change-volume
- 4) clicks
- 5) clip
- 6) harmonic
- 7) high-pass-filter
- 8) low-pass-filter
- 9) normalize
- 10) peaking-equalizer
- 11) percussive
- 12) pitch-shift
- 13) reverb
- 14) speed
- 15) time-stretch

C.2. Computing log-Mel spectrograms

The log-Mel spectrograms are generated on-the-fly during training from a randomly selected 1-second crop of an audio waveform as the model’s input. We compute it by applying a short-time Fourier transform with a window size of 25ms and a hop size equal to 10ms to extract 64 Mel-spaced frequency bins for each window. During evaluation, we average over the predictions of non-overlapping segments of an entire audio clip.

C.3. Audio neural network architecture

Our audio convolutional neural network is inspired by [62] and it consists of four blocks. We perform separate convolutions in each block with a kernel size of 4. One on the temporal and another on the frequency dimension, we concatenate their outputs afterward to perform a joint $1 \times 1$ convolution. It allows model to capture fine-grained features from each dimension and discover high-level features from shared output. We apply L2 regularization with a rate of 0.0001 in each convolution layer and also use group normalization [72]. Between the blocks, we utilize max-pooling to reduce the time-frequency dimensions by a factor of 2 and use a spatial dropout rate with a rate of 0.2 to avoid over-fitting. We apply ReLU as a non-linear activation function and feature maps in the convolutions blocks are 24, 32, 64, and 128. Finally, we aggregate the feature with a global max pooling layer which are fed into a fully-connected layer with number of units equivalent to the number of classes.

C.4. Training

C.4.1 Small-scale experiments

For experiments on CIFAR-10, CIFAR-100, and other smaller datasets, we use Tensorflow for running experiments on a single T4 GPU with a batch size of 512 using an Adam [30] optimizer with a fixed learning rate of 0.001. We use standard augmentations including, random left right flip, and random crop. The supervised models also uses cutout augmentations with a cutout size of 16 × 16. For Mix-up, we sample $\alpha$ uniformly at random between zero and one. In Cut-Mix, we use a fixed value of 0.25 for $\alpha$ and $\beta$. With the following setup, each of the single image distillation experiment of 1K epochs took around 2–3 days. The supervised and standard (using source data) distillation models are trained for 100 epochs (per epoch 2000 steps) with a batch size of 128 using SGD for optimization. For VGG and ResNet models, we use a learning rate schedule of 0.01, 0.1, 0.01, 0.001 decayed at following steps 400, 32000, 48000, 64000 with momentum of 0.9. For WideResNet, we use a learning rate schedule of 0.1, 0.02, 0.004, 0.0008 that is decayed at following steps 24000, 48000, 64000, 80000 with nesterov enabled.

In audio experiments, we use an Adam [30] optimizer with a fixed learning rate of 0.001 and use batch size of 128 and 512 for standard models and single-clip distillation, respectively. Furthermore, we also utilize Mix-Up augmentation during our knowledge distillation experiments.

C.4.2 Large-scale experiments

We use PyTorch’s DistributedDataParallel engine for running experiments on 2 A6000 GPUs in parallel with batch-sizes of 512 each. For optimization we use AdamW [38] with a learning rate of 0.01 and a weight-decay of 10$^{-4}$. These values were determined by eyeballing the results from [6] to find a setting that might generalize across datasets, as we do not have enough compute to run hyper-parameter sweeps. With this setup, each 200 epoch
experiment took around 5 days. We have also found that halving the batch-size performs equally well, and is more amenable to lower-memory GPUs. For the distillation experiments, we use Cut-Mix [76] with its default parameters of $\alpha = \beta = 1.0$.

C.5. GIST features comparison

In Fig. 2a, we compute the distances of GIST features [46] of 10K training images of the “City” image at resolution $256 \times 256$ in Fig. 2a. Following [3], we L2 normalize these GIST features before computing pair-wise distances and plotting the histogram of the values.

C.6. Representation learning experiments

**Pretraining.** We pretrain using the official code of MoCo-v2 [24] in the standard 200 epoch setting, while applying their default “v2” augmentations during training.

**Evaluation.** We follow standard linear evaluation procedure from the MoCo-v2 repo. which uses a batch size of 256, learning rate of 0.03 which is multiplied by 0.1 at epochs 60 and 80 for a total of 90 epochs. For the data-efficient full-fine tuning, we utilize the implementation from SwAV [12], which trains for 20 epochs using a cosine decay learning rate schedule and a batch size of 256. For the data-efficient SVM classification experiments, we follow the implementation of [34] and report average results from 5 trials and keep the SVM’s cost parameter fixed at a value of 0.5.