Dataset Distillation

Radek Bartyzal

Let's talk ML in Prague

13. 12. 2018

Knowledge distillation

Teacher

- high-capacity model
- good performance

Student

- more compact model
- not as good performance as the teacher but better than if it was trained without it

By transferring knowledge, one hopes to benefit from the student's compactness while suffering only minimal degradation in performance [1], [4].

Dataset Distillation

1. pick an original dataset e.g. MNIST
2. select a specific architecture and initial weights
3. synthesize new smaller dataset using the selected architecture and weights
4. train the model on this small dataset
5. evaluate the model on the test set of the original dataset

Problems

Problem:
Why not just remember the final trained weights instead of the initial ones?

Solution:

- init weights are from a distribution D
- create such examples that work for multiple sampled init weights

Uses

Advantages:

- compressed representation
- faster training - only couple GD steps
- fine-tune pre-trained weights to new dataset
- dataset poisoning = fine-tune the model to predict trash
- how much can we compress the information of the dataset?
- can we train the model by synthetic images that are not on the natural image manifold?

Compressed representation:

- architecture
- init weights distribution
- learned learning rate
- created small train dataset

Algorithm

Algorithm 1 Dataset Distillation

Input: $p(\theta_0)$: distribution of initial weights; M: the number of distilled data
Input: α: step size; n: batch size; T: the number of optimization iterations; $\tilde{\eta}_0$: initial value for $\tilde{\eta}$

Initialize $\tilde{\mathbf{x}} = \{\tilde{x}_i\}_{i=1}^{M}$ randomly, $\tilde{\eta} \leftarrow \tilde{\eta}_0$
for each training step $t = 1$ to T **do**
    Get a minibatch of real data $\mathbf{x}_t = \{x_{t,j}\}_{j=1}^{n}$
    Sample a batch of initial weights $\theta_0^{(j)} \sim p(\theta_0)$
    for each sampled $\theta_0^{(j)}$ **do**
        Compute updated parameter with GD: $\theta_1^{(j)} = \theta_0^{(j)} - \tilde{\eta}\nabla_{\theta_0^{(j)}}\ell(\tilde{\mathbf{x}}, \theta_0^{(j)})$
        Evaluate the objective function on real data: $\mathcal{L}^{(j)} = \ell(\mathbf{x}_t, \theta_1^{(j)})$
    end for
    Update $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} - \alpha\nabla_{\tilde{\mathbf{x}}}\sum_j \mathcal{L}^{(j)}$, and $\tilde{\eta} \leftarrow \tilde{\eta} - \alpha\nabla_{\tilde{\eta}}\sum_j \mathcal{L}^{(j)}$
end for

Output: distilled data $\tilde{\mathbf{x}}$ and the optimized learning rate $\tilde{\eta}$

Results

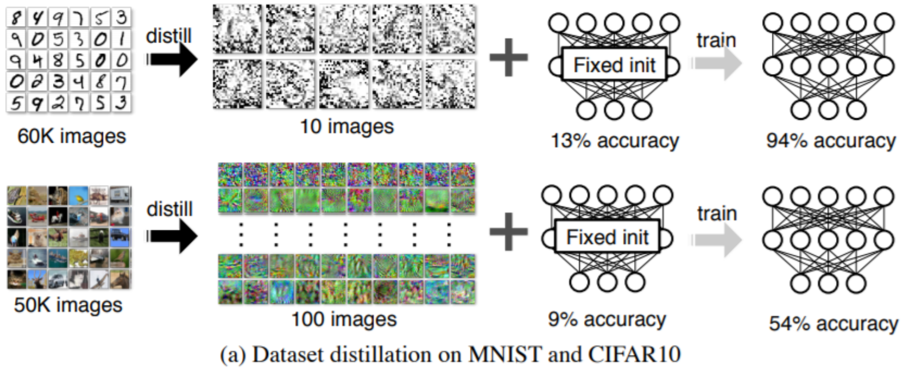
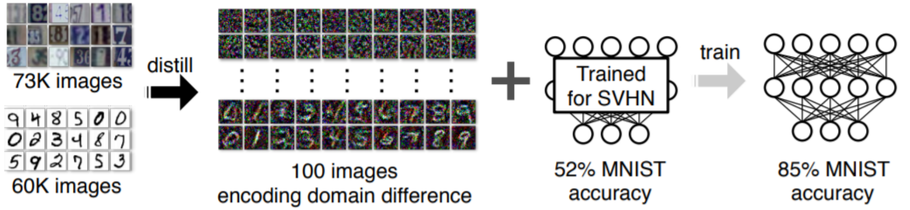
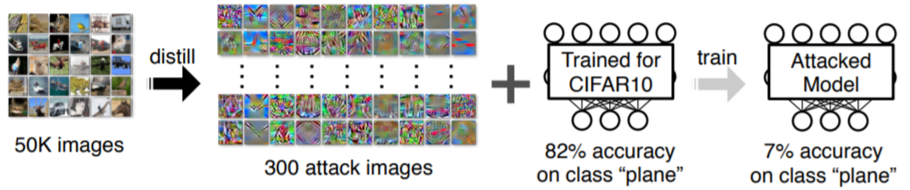


Figure: Fixed = sampled from

Results

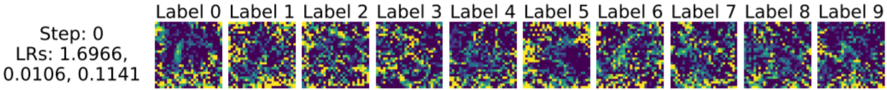


(b) Distillation for classifiers pre-trained on SVHN to perform well on MNIST

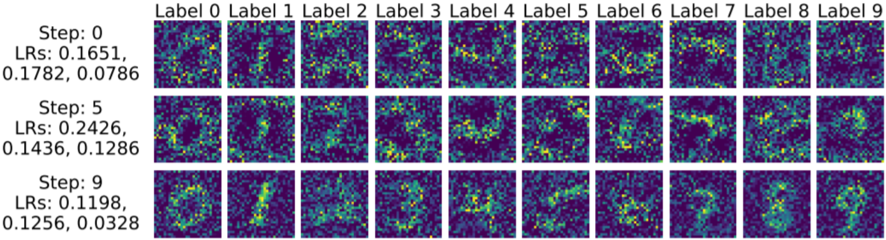


(c) Distillation for a malicious objective on well-trained CIFAR10 classifiers

Results

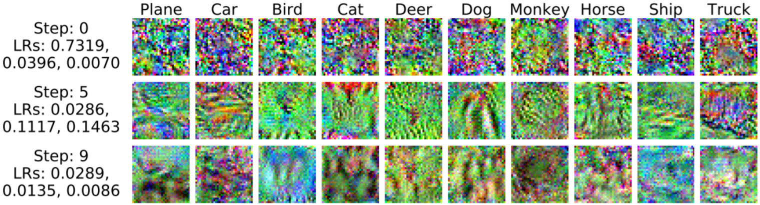


(a) MNIST. These images train networks with a particular initialization from 12.9% test accuracy to 93.76%.

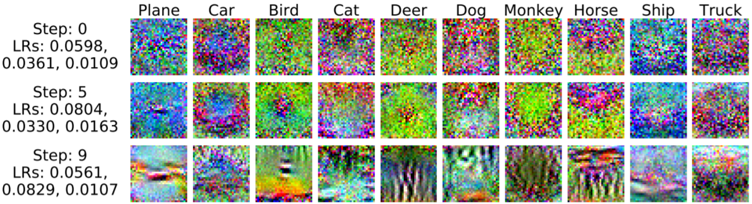


(a) MNIST. These images train networks with unknown initialization to 79.50% $\pm$ 8.08% test accuracy.

Results



(b) CIFAR10. These images train networks with a particular initialization from 8.82% test accuracy to 54.03%.



(b) CIFAR10. These images train networks with unknown initialization to 36.79% $\pm$ 1.18% test accuracy.

Sources

1. Belagiannis, Vasileios, Azade Farshad, and Fabio Galasso. "Adversarial Network Compression." arXiv preprint arXiv:1803.10750 (2018). Accessible from: https://arxiv.org/abs/1803.10750

2. Wang, Tongzhou, et al. "Dataset Distillation." Arxiv preprint. 2018. Accessible from: https://arxiv.org/abs/1811.10959v1

Sources

4. Breiman, Leo, and Nong Shang. "Born again trees." ps (1996). Accessible from: https://www.stat.berkeley.edu/~breiman/BAtrees.pdf

5. Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. "Distilling the knowledge in a neural network." arXiv preprint arXiv:1503.02531 (2015). Accessible from: https://arxiv.org/pdf/1503.02531.pdf