Generating Ad-hoc Curricula

Andrew Lampinen (CS 229, Stanford University)

Introduction

Often in machine learning, proofs of convergence of learning algorithms assume that the data distribtuion is static over the course of learning. However, work on Curriculum Learning for neural networks [1] has shown that in practice, altering the data distribution over the course of learning can improve results, at least in some cases. Specifically, starting with easier examples first and then progressing to harder examples can be beneficial, possibly even making a task learnable when it is not learnable from a reasonable amount of data using the standard strategy.

However, in many cases we lack explicit knowledge about the structure of the problem from which we could construct a curriculum. For example, on the MNIST task, how do we decide which examples are “easy” or “hard” without expensive human input? Is there a way we can craft curricula for a task while remaining more agnostic to what the task actually is (ideally just from the data and labels)?

Potential Curricula

We considered two possible strategies for crafting a curriculum, and validated them on the toy example from [1] (results not shown).

Previous Classifier: Ranking example easiness by margin for a classifier trained without curriculum. This can be seen as an approximation to one of the original curricula suggested by Bengio and colleagues, where instead of using the unknown true margin as a measure of difficulty we are using the margin from our approximation to the classification boundary. This strategy performed poorly on the validation task, see discussion for more details.

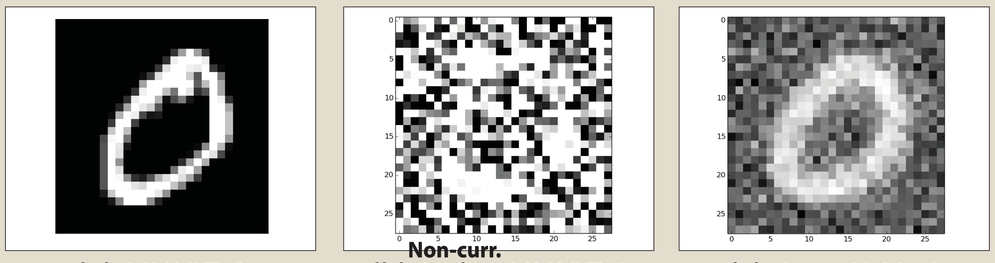
Data Recombination: If our data are noisy samples from a distribution, we might think that by recombining the data (e.g. averaging) we would get more representative/easier examples. This requires some assumptions about the data distribution, however, see discussion. This strategy performed as well on the validation task as Bengio’s curriculum which incorporated explicit task knowledge.

Main Tasks: MNIST & Noisy MNIST

Because the data-recombination curriculum succeeded on the validation task, we decided to evaluate it on a more complex task: the MNIST hand-written digit recognition problem (see Fig. 1a for an example digit). Because performance on MNIST is already quite good, we also tested on a Noisy MNIST dataset we created, where the images had a great deal of independent gaussian and bernoulli noise added and the result clipped to [0,1] (see Fig. 1b for an example).

We used TensorFlow [2] to create two identical convolutional neural networks (two convolutional layers, each with 5 x 5 filters and stride 1, and 2 x2 max-pooling, 32 filters in the first layer, 64 in the second, followed by a final fully-connected classification layer) and trained them (cross-entropy error, using the Adam optimizer [3] with a learning rate of 0.001 and batch size of 50) to an optimal error on a held-out validation set. The first network was trained with no curriculum, the second with a curriculum where the first 20 batches’ training examples were actually averages of 50 training examples from a given class (see Fig. 1c for an example average from Noisy MNIST), the following 40 batches contained averages of 20 images, and the remaining batches were identical to those seen by the non-curriculum strategy.

Fig. 1:



(a) MNIST 0 (b) Noisy MNIST 0 (c) Avg. N.M. 0

Results

MNIST: The network trained with the data recombination currriculum achieved a mean test accuracy of 99.1%, compared to 99.0% with no curriculum. (See Fig. 2a.) This difference was statistically significant (paired t-test, $t(99) = 4.1$, $p < 1e\text{-}4$). However, it is probably practically insignificant.

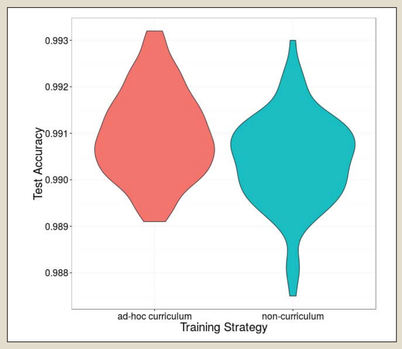


Fig. 2a: MNIST

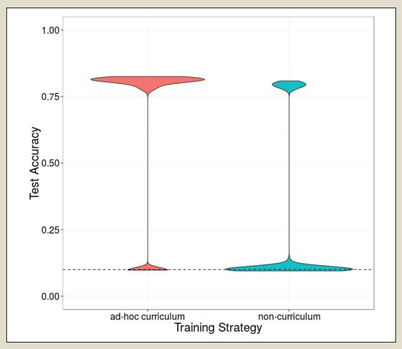


Fig. 2b: Noisy MNIST

Noisy MNIST: The network trained with the data recombination currriculum achieved a mean test accuracy of 66.1%, compared to 26.2% with no curriculum. (See Fig. 2b.) This difference was statistically significant (paired t-test, $t(99) = 11$, $p < 1e\text{-}15$). This is a substantial improvement.

Table 1: Test accuracies on each task

	MNIST	Noisy MNIST
Non-curr.	99.0%	26.2%
Ad-hoc curr.	**99.1%**	**66.1%**

Discussion / Conclusions

We proposed creating ad-hoc curricula for training neural networks, and tested two strategies for this. The first strategy, creating curricula based on a previous classifier’s margin performed poorly, perhaps because the margins are systematically biased by the error in the previous classifier. The second strategy, creating curricula by recombining data gave substantial improvements in performance on our Noisy MNIST task.

This strategy seems most effective in settings where the data are particularly noisy, and are well represented by their central tendency. On less noisy tasks like standard MNIST, the performance improvements are quite small. On a more complex task like ImageNet, a strategy like this might be less effective since each object can have many poses relative to the viewer, so the average may not be representative. However, it is still possible that an averaged image could obtain useful information about color, standard poses, etc.

This strategy could also be expanded in a variety of ways, e.g. clustering the data from each class and using the centroids from each cluster for “easy” examples. This might be able to handle issues like different poses (as long as there aren’t too many) or different ways of writing “2,” for example.

Conclusions:
Ad-hoc curricula may be beneficial for learning. Data-recombination based curricula in particular show benefits on several tasks.

References

[1] Bengio, Y., Louradour, J., Collobert, R., and Weston, J. *Curriculum learning*. Proceedings of the 26th annual international conference on machine learning (2009).
[2] GoogleResearch. *TensorFlow: Large-scale machine learning on heterogeneous systems.* (2015)
[3] Kingma, D., and Ba, J. *Adam: A Method for Stochastic Optimization.* International Conference on Learning Representations (2014).