Generating training data for supervised machine learning using GANs

Seminar paper

"Künstliche Intelligenz in RoboCup"

Jan Brehmer

***Abstract*— A common problem of deep machine learning is its high demand for training data. Given the recent popularity of generative adverserial networks (GANs), which are able to create completely new samples from a learned training dataset distribution, this work attempts to "just generate some more training data". It shows that training a classifier on a GAN-augmented dataset can exceed the performance of the same classifier trained on the original dataset exclusively.**

I. Introduction

Deep machine learning has been on an undisputable rise for the last couple of years, achieving superhuman performance accross a vast variety of tasks of several domains, e.g. image recognition and segmentation, speech recognition, reading comprehension and gaming. [1]

In order to make this possible, new kinds of neural networks, activation functions, training methods and optimizations have been developed and are still being researched. However, even the performance of highly sophisticated machine learning architectures is limited by the amount and distribution of the provided training data, which has to be compiled (and labeled) manually – a lengthy and tedious task, which researchers and users try to avoid.

This problem becomes even more apparent in the context of reinforcement learning, where the training data for learning a policy has to be gathered by trial and error while interacting with an environment. Performed in the real world, this process can be very slow and expensive, since the physical agent could damage itself while exploring its action space. Hence, learning in a simulation seems like the obvious solution. While simulation is broadly used across a variety of domains with success, often times it is a valid option for reinforcement learning as well. But again, this simulation has to be built manually. Sometimes a usable simulation demands implementing complex interrelations, which should have been learned by the agent in the first place.

Now given that a reinforcement learning agent alternates between applying actions to an environment and observing the environment, we can let it learn its own estimate ("simulation") of the environment using $(\text{environment}, \text{action}) \rightarrow \text{environment}'$ mappings as training data. This can be done additionally to learning the policy, but without the need for additional interactions with the environment. Using the learned model of the environment to train the policy on *imagination rollouts* (virtual training runs) can substantially improve learning speed. [2, 3]

Reinforcement learning has also coined the term *sample efficiency* for learning speed, i.e. the fewer environment interactions are neccessary to achieve a certain level of performance, the more sample efficient a model is. This is not to be confused with the maximum performance capacity of a model, which is the upper limit of what the model is capable of (given infinite training samples). In the case of neural networks the model capacity usually increases with the number and widths of layers. The model capacity is not improved by the use of imagination rollouts.

In image-based machine learning such as image classification, training datasets may be enriched by image transformations, such as rotation, reflection and scaling. [4]

This work attempts to apply GANs (see sections II "Model Overview" and III "Learning") on an image training dataset to learn to generate additional samples, similar to model-based reinforcement learning. These generated samples are used for training of a baseline classifier among the original samples to increase the classifier accuracy.

This is a relatively new approach, as interest in GANs has risen only over the last few years. However, similar research is already available: In [5] a simulator is used to generate labeled image data, while GANs are trained on unlabeled real data to improve the realism of the generated output. And in [6] GANs are trained on unlabeled image data of a person re-identification problem, generating person mixture images. These images are fed into classifier training along a uniformly distributed person vector, which increases the classifier accuracy.

II. Model Overview

The neural network setup used for experiments consists of a convolutional neural network (CNN) classifier and two GANs. To specify a networks' architecture the following notation is used.

$\mathcal{C}_{N,w,s,f_a}$ denotes a convolutional layer with N kernels of size $w \times w$, stride s and an activation function f_a, where lReLU is the leaky rectified linear unit. $\mathcal{F}_{N,f_a}$ denotes a fully connected layer with N neurons and an activation function f_a. $\mathcal{D}$ denotes a dropout layer with a dropout rate of 0.5. $\mathcal{B}$ denotes a batch normalization layer with a moving average momentum of 0.99.

GANs have two components: a generator and a discriminator. Both GANs' discriminator networks and the classifier are designed to process 64×64 pixel RGB im-

ages. Their network architecture has the following layers: $\mathcal{C}_{256,4,2,\text{lReLU}} \rightarrow \mathcal{C}_{512,4,2,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{B} \rightarrow \mathcal{C}_{1024,4,2,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{B} \rightarrow \mathcal{C}_{2048,4,2,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{B} \rightarrow \mathcal{F}_{400,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{F}_{1,\text{sigmoid}} \rightarrow \mathcal{D}$.

The GANs' generator networks are arranged symmetrically to their discriminator counterparts and output 64×64 pixel RGB images respectively. The inputs are normally distributed random vectors of length 100. Their architecture consists of the following layers: $\mathcal{F}_{400,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{B} \rightarrow \mathcal{F}_{32768,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{B} \rightarrow \mathcal{C}^T_{1024,4,2,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{B} \rightarrow \mathcal{C}^T_{512,4,2,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{B} \rightarrow \mathcal{C}^T_{256,4,2,\text{lReLU}} \rightarrow \mathcal{D} \rightarrow \mathcal{B} \rightarrow \mathcal{C}^T_{3,4,2,\tanh}$.

Note that the convolution layers are transposed in the generators, which is also called deconvolution. To feed in the output of a fully connected layer to a (de)convolution layer or vice versa, the matrices are reshaped accordingly, so e.g. the 32768 values long output vector of the generators' second FC layer is reshaped to 2048 kernels of size 4×4.

III. Learning

A GAN is a network pair that learns a model of the input data. Each generated sample mimics the learned input data. To achieve that, a discriminator network is trained to classify *real* and *generated* samples. Assuming the discriminator returns the probability p_r of the input image being real, the loss function used for training on real images is

$$L_r = -p_r$$

and when training on generated (fake) images it is

$$L_f = p_r.$$

The loss used to train the generator network is based on the discriminator output when feeding it a generated image:

$$L = -p_r$$

That way the discriminator gets better on identifying generated images and the generator is trained to create images, which the discriminator classifies as being real.

(a) (b)

Fig. 1: A sample from each class to learn: a) ball and b) robot.

In the experiment a CNN classifier is trained on a dataset of robot and ball images (fig. 1) as a baseline over 350 episodes. Only $n = 10$ images are used per class[1], so batch size is set to $n-1 = 9$, which yields a maximum of 10 distinct batches when rotating over the training data. To generate more training data for the classifier, two GANs are trained on the same 10 images over 1500 episodes: one for robot images and one for ball images. Again, the batch size is $n-1 = 9$. Figure 2 shows a few examples of generated images.

Fig. 2: Typical GAN output images. Left half: GAN trained on ball images. Right half: GAN trained on robot images.

The generated images are then used for classifier training either by alternating original and generated image batches or by using mixed batches.

IV. Results

For evaluation, the classifier accuracy is used as the measure of performance. It is evaluated on a test data set every 5 episodes during classifier training. Each evaluation averages several distinct training runs (mostly 20) with a randomly selected training data set, using the rest of the images as the test data set.

As a first result, the experiments show that training on mixed batches is superior to alternating batches, as the average classifier accuracy was higher at each episode checkpoint.

The main insight is however, that adding the generated images to the training data increases accuracy growth early in the training process and even leads to a more accurate classifier after all (fig. 3).

For further comparison, random normally-distributed pixel arrays were used to augment training instead of the GAN-generated images. This method has shown to be of great benefit in the early episodes, but hindered the training later on, not improving the final classifier accuracy.

After 20 training runs each, the average accuracy was 0.85 for the baseline classifier trained only on the original images, 0.86 for the noise-augmented training data set and 0.88 for the GAN-augmented training data set.

Further findings strongly suggest that using GAN-augmented training data also provides the most stable results.

[1]The CNN reaches near-perfect accuracy using 30+ training samples, $n = 10$ yields a comparable baseline.

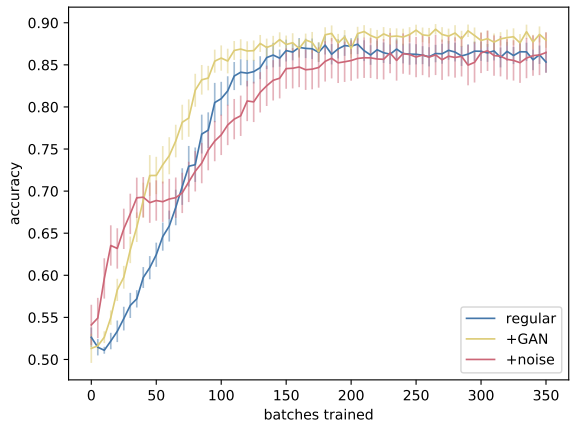

Fig. 3: Classifier accuracy progression while learning on the given training data (blue), additionally using GAN-generated images (yellow) or noise (red), averaged over 20 training runs each, with standard errors.

TABLE I shows the standard deviation of the accuracy for all three training variants over 20 runs each as an average across all training checkpoints and the individual standard deviation at the last checkpoint.

	regular	+GAN	+noise
mean	0.0593	**0.0525**	0.0631
final	0.0563	**0.0412**	0.0442

TABLE I: Classifier accuracy standard deviations during and after training.

When used in an actual attempt to augment training data, manual GAN selection could be applied additionally, i.e. discarding poor GANs after visually checking their outputs. It was found that this would further increase classifier accuracy, but because of the lack of an objective selection metric this method was considered unsuitable for actual evaluation.

V. Conclusion

Having decent success with very basic GANs on a simple classifying task by improving performance and robustness, this work implies a rather high potential of GAN-augmenting training data, especially when dealing with sparse training data while having some spare computation power.

In future work, more advanced GAN aritectures could be evaluated along with a harder classification tasks, which would need more original training data to provide a comparable baseline, since training on 10 images is fairly uncommon and only allows for very small batches.

Another thread of research would be merging each class' GAN into one by supplementing the class label to the generator's input, or even skipping actual classifier training at all by making the GAN discriminator estimate the class.

References

[1] P. Eckersley, Y. Nasser *et al.*, "EFF AI Progress Measurement Project," https://www.eff.org/de/ai/metrics, 2017–, accessed 2019-04-20.

[2] N. Wahlström, T. B. Schön, and M. P. Deisenroth, "From Pixels to Torques: Policy Learning with Deep Dynamical Models," *ArXiv e-prints*, Feb. 2015.

[3] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous Deep Q-Learning with Model-based Acceleration," in *International Conference on Machine Learning*, 2016, pp. 2829–2838.

[4] P. Y. Simard, D. Steinkraus, J. C. Platt *et al.*, "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis." in *Icdar*, vol. 3, no. 2003, 2003.

[5] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from Simulated and Unsupervised Images through Adversarial Training," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2107–2116.

[6] Z. Zheng, L. Zheng, and Y. Yang, "Unlabeled Samples Generated by GAN Improve the Person Re-identification Baseline in vitro," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3754–3762.