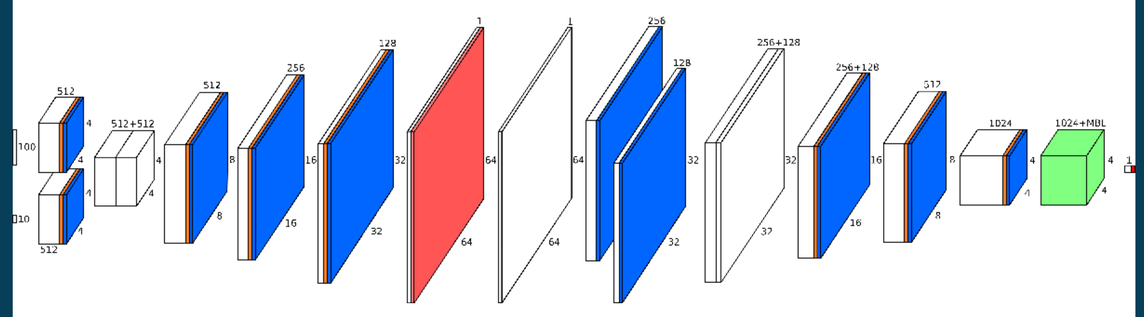
Semi-Supervised Training of a Generative Adversarial Network

Anna Baumeister (w270) anna.baumeister@tum.de
Nicholas Gao (w348) nicholas.gao@tum.de
Matthias Mayer (w213) matthias.mayer@tum.de
Yannick Spreen (w231) y.spreen@tum.de
Frederic Rackwitz (w231) rackwitz@in.tum.de

Generative Adversarial Networks

Generative adversarial networks (GANs) work trough interaction of a fake/real classifier network (discriminator) and a generator network. The discriminator trains on real data with ground truth 1 and generated data with ground truth 0. The generator trains on the discriminators output of a generated image.The architecture we use is a Deep Convolutional GAN (DCGAN). The generator works by deconvolving a gaussian distributed random input vector into an image, where the convolutions weights learn how to generate images similar to the dataset. The discriminator uses convolutions to reduce an image into a score like any other DCNN-classifier. We use stride 2 convolutions instead of pooling since literature suggests that this helps preserving feature dependencies. In order to prevent network collapse, batch normalization layers were appended to the layers. Training was realized using the ADAM optimizer.

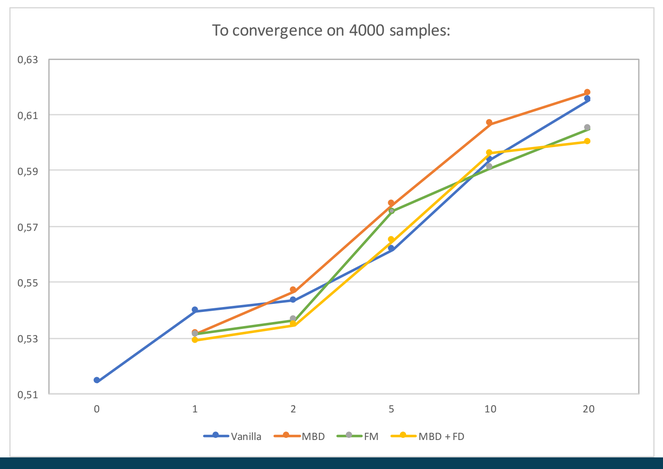


Conditional GANs

In contrast to classical generative adversarial networks (GANs) which try to model the distribution p(x) to generate new data, conditional GANs (cGANs) aim to model the conditional distribution p(x|y). This is realized by adding class labels for both nets as an input. This approach allows the specific generation of an image of a certain class and the usage of more training data since the labels of each image must be part of the training process. Furthermore, it prevents the generator from interpolating between multiple classes, since a distribution is modeled for each class. In contrast to training a single GAN for each class cGANs allow the share of features if the classes are correlated which reduces training time and the amount of required data. A major drawback of cGANs is the need of labeled data in order to train the network while classical GANs can be trained unsupervised and in case of un- or weakly correlated classes cGANs tend to offer worse results due to the share of features.

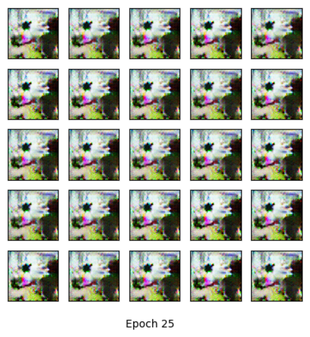
Semi-Supervised Learning

One major reason for the recent success of deep CNNs is the availability of huge labeled datasets, such as ImageNet. In order to apply CNNs to different fields fine tuning is the predominant technique if domains are sufficiently similar. Semi-supervised leaning tries to establish another approach for big datasets that have not been labeled yet (which would be costly). It encompasses methods that are able to learn classification from datasets with sparse labels only.
We tried to achieve this with GANs. In theory, their discriminators should learn an effective feature representation of images during training (no labels are needed in this stage). The discriminator's first several layers may than be reused for a classification task, which is learned on a labeled subset of the data. By replacing layer 4 and 5 from our discriminator, which was pretrained in a GAN for different number of epochs, and training the new parts on labeled data (4000 of 50000 images) we could show significant improvements in CIFAR-10 classification in comparison to the whole network trained from scratch.



Minibatch Discrimination

Mode collapse is a commonly encountered failure case in training GANs in which the generator emits images only from a single point of the distribution which the discriminator has judged to be highly realistic. As a result, the generator converges to producing images with very little to no variety as shown below. Because the Discriminator looks at each example from the batch in isolation, it has no way of judging whether the images produced by the generator have the correct ammount of entropy. Minibatch discrimination attempts to solve this issue by allowing the Discriminator to look at several examples from the batch at the same time.



Feature Matching

Feature matching introduces another loss function for the Generator to minimize. Additionally to maximizing the missclassification rate of its outputs the generator tries to meet the expectation of a feature map f(x) within the discriminator. With stochastic optimization this results in the additional loss term

$$||\text{Mean}(f(x)) - \text{Mean}(f(G(z)))||_2^2$$

for the generator, where x is the feature map from a batch of real images, z is the random latent space variable for the generator and f is the last hidden featuremap of the discriminator.

(This is motivated by the observation that the Discriminator is searching for the most discriminative features to tell real and fake apart. Minimizing the distance between real and fake feature maps should therefore result in a more convincing Generator. [OpenAI] observed that this approach lead to stronger semi-supervised learning but weaker subjective image quality.)

Conclusion

GANs aren't easy. We quickly found that our original idea of enhancing small datasets to improve classification learning is only feasible within the limits of our application of semi supervised learning. The augmentations we tried to enhance the GANs performance lead to the expected results. More training time and optimization of hyperparameters might lead to big improvemnets in image quality. Unfortunately this exceeds the available compute time for this project. But: We see the power of GANs and presume future advancements using GANs.

References

SALIMANS, Tim, et al. Improved techniques for training gans. In: Advances in Neural Information Processing Systems. 2016. S. 2234-2242.