Is Generator Conditioning Causally Related to GAN performance*

Odena et al.

2018

1 What

The authors discover that Generator's Jacobian becomes ill-conditioned at the beginning of training. They also find that average conditioning of G predicts the behaviour of two metrics usually used for GAN training quality measuring: the Inception Score [Salimans et al., 2016] and the Freshet Inception Distance (FID) [Heusel et al., 2017]. They try Jacobian Clamping (penalising conditioning number) and this improves both metrics on several datasets. Moreover, proposed regularisation improves the stability of training (reduces the variance of the two metrics).

2 Why

Controlling Jacobian singular values is very important in DL [Pennington et al., 2017]. One of the problems with GANs is the 'mode collapse', situation when G outputs samples from a small subset of modes in training data. They decide not to study it from the probabilistic perspective but apply linear algebra/functional analysis kungfu to deal with it.

3 How

[1]

Inception score:

$$\exp\left(\mathbb{E}_{\mathbf{x}\in P_g}[KL(p(y|\mathbf{x})\|p(y)]\right) \tag{1}$$

*Notes by Vitaly Kurin `https://yobibyte.github.io/`

[1]Formulas/notation explanation are copy pasted from the source. Kudos to the authors for not doing `"include paper.pdf"`

where $\mathbf{x}$ is a GAN sample, $p(y|\mathbf{x})$ is the probability for labels y given by a pre-trained classifier on $\mathbf{x}$, and $p(y)$ is the overall distribution of labels in the generated samples (according to that classifier).

Frechet Inception Distance [Heusel et al., 2017][2]: Assumption: the activations in the coding layer of the pre-trained classifier come from a multivariate Gaussian. If the activations on the real data are $N(m, C)$ and the activations on the fake data are $N(m_w, C_w)$, the FID is given by:

$$\|m - m_w\|_2^2 + \mathrm{Tr}\big(C + C_w - 2\big(CC_w\big)^{1/2}\big) \tag{2}$$

Generator G is a mapping from latent state to observation space: $G : Z \to X$. The main hero of the paper is mapping $M : Z \to J_z^T J_z$, where J_z is a Jacobian matrix $(J_z)_{i,j} \equiv \frac{\partial G(z)_i}{\partial z_j}$. M_z denotes $J_z^T J_z$.

If we fix some point z:

$$\lim_{||\epsilon|| \to 0} \frac{||G(z) - G(z + \epsilon v_k)||}{||\epsilon v_k||} = \sqrt{\lambda_k}, \tag{3}$$

where λ_k, v_k are k-th eigenvalue and eigenvector of M_z.

I'll copy the next gem from the paper:

Less formally, the eigenvectors corresponding to the large eigenvalues of M_z at some point $z \in Z$ give directions in which taking a very small "step" in Z will result in a large change in $G(z)$ (and analogously with the eigenvectors corresponding to the small eigenvalues).

Working with the whole spectrum (set of eigenvalues) is hard, we need some metric. Metric tensor M_z's condition number $\frac{\lambda_{max}}{\lambda_{min}}$ is the metric. The higher here, the worse.

Eigenvalues of M_z are equivalent to squared singular values of J_z. Practically, all the manipulations are done on singular values of J_z.

They use tensorflow for constructing the Jacobian J_z.

Is there a casual relationship between the conditioning number and the two metrics (after we see that there is a correlation)? The authors do an intervention study to find out: feed two mini-batches at a time to the Generator, noise from p_z and same noise with some perturbations controlled by ϵ. Then they divide the norm of change in outputs from batch to batch by the norm of the change in inputs from batch to batch. They penalise if the number is larger/smaller than hyperparameters $\lambda_{max}/\lambda_{min}$. I'll copy paste the complete algorithm below.

Jacobian Clamping improves the average performance of GANs, but not the best-case performance (but who cares about the single run?).

4 Evaluation

The main evaluation procedure is the following: fix a batch of $z \sim p(z)$ and examine the condition number of M_z along training.

[2]Didn't get this one. Need to read the original paper [Heusel et al., 2017]

Algorithm 1 Jacobian Clamping

Input: norm ϵ, target quotients $\lambda_{max}, \lambda_{min}$, batch size B
repeat
  $z \in R^{B \times n_z} \sim p_z$.
  $\delta \in R^{B \times n_z} \sim N(0, 1)$.
  $\delta := (\delta / ||\delta||)\epsilon$
  $z' := z + \delta$
  $Q := ||G(z) - G(z')|| / ||z - z'||$
  $L_{max} = (\max(Q, \lambda_{max}) - \lambda_{max})^2$
  $L_{min} = (\min(Q, \lambda_{min}) - \lambda_{min})^2$
  $L = L_{max} + L_{min}$
  Perform normal GAN update on z with L added to generator loss.
until Training Finished

Datasets used:

- MNIST
- CIFAR-10
- STL-10

The before/after plots clearly show that the proposed solution (Jacobian clamping) significantly improves GAN training procedure as well as final results (getting an almost uniform distribution of samples).

Before penalising conditioning number, it increases for half of the runs (with different initialisations) and decreases (stable) for another half.

Another interesting experiment the authors conduct is training 10 runs of GAN and comparing it with 10 runs of VAEs. VAE runs display almost the same conditioning, whereas GANs do not.

Both mentioned metrics show the extent the Generator is missing modes. The authors show, that ill-conditioned Generators often have 0 samples from the least sampled class (after drawing 360 samples). At the same time, well-conditioned models induce almost uniform distributions.

Just to intercept all lines of retreat, they train a state-of-the-art WGAN-GP (maybe it's not already after a month, who knows these GAN researchers). With the same hyperparams as in previous experiments, the authors show that *reducing the number of discriminator updates* + Jacobian Clamping more than halves the wall-clock time without harming much the score.

5 Comments

- This paper is a gem. This is another reminder why you should do less `import tensorflow as tf`, and more `from functional_analysis import Kolmogorov, Fomin`

- apart from mathematical formulations, the authors also describe it less formally providing an amazingly clear picture
- another gem: *Our aim here is not to make claims of State-of-the-Art scores*[3] *but to provide evidence of a causal relationship between the spectrum of M_z and the scores.*

References

[Heusel et al., 2017] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6629–6640.

[Pennington et al., 2017] Pennington, J., Schoenholz, S., and Ganguli, S. (2017). Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In *Advances in neural information processing systems*, pages 4788–4798.

[Salimans et al., 2016] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242.

[3] We regard these claims as problematic anyway. One issue (among many) is that scores are often reported from a single run, while the improvement in score associated with a given method tends to be of the same scale as the inter-run variance in scores.