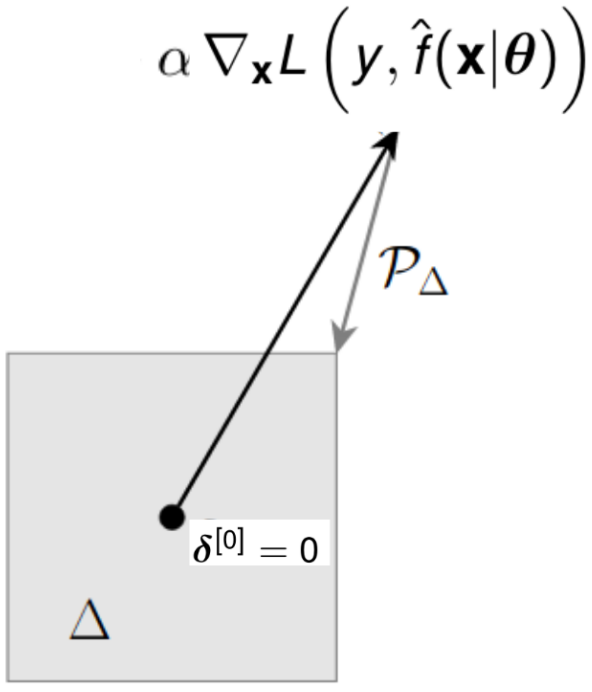
Deep Learning

Adversarial Training Advances



Learning goals

- Advanced adversarial training
- Projected gradient descent
- Fast gradient sign method

PROJECTED GRADIENT DESCENT

- In contrast to logistic regression, neural networks can have a bumpier, non-convex loss surface.
- As a consequence, Danskin's theorem does not longer hold and the inner optimization problem must be solved approximately.
- One approximation method is projected gradient descent (PGD)[1].
- Let $\hat{f}$ be the pretrained model, $\mathbf{x}$ the input to the model, y the target and $L\left(y, \hat{f}(\mathbf{x}|\boldsymbol{\theta})\right)$ the loss function used to train the network.
- In each gradient descent step, the basic PGD algorithm updates $\boldsymbol{\delta}$ as:

$$\boldsymbol{\delta}^{[t+1]} = \mathcal{P}\left(\boldsymbol{\delta}^{[t]} + \alpha \nabla_{\boldsymbol{\delta}} L\left(y, \hat{f}(\mathbf{x} + \boldsymbol{\delta}^{[t]}|\boldsymbol{\theta})\right)\right)$$

  where $\mathcal{P}$ denotes the projection onto the ball of interest.

[1] Technically speaking, it is gradient *ascent* since we are maximizing a function, but for the sake of generality, we just refer to the process here as gradient descent.

PROJECTED GRADIENT DESCENT

- In essence, $\boldsymbol{\delta}^{[t+1]}$ is obtained by moving from $\boldsymbol{\delta}^{[t]}$ (with a step size α) **in the direction** of the gradient of the loss with respect to $\boldsymbol{\delta}$ (evaluated at $\boldsymbol{\delta}^{[t]}$) and then projecting back to Δ.
- In case where the feasible set is $\mathcal{B}_{\epsilon}^{\infty}$, the projection of an arbitrary vector **z** is simply:

$$\mathcal{P}(\mathbf{z}) = \mathit{clip}(\mathbf{z}, [-\epsilon, \epsilon])$$

- Therefore, when Δ is $\mathcal{B}_{\epsilon}^{\infty}$, one gradient step is then defined as:

$$\boldsymbol{\delta}^{[t+1]} = \mathit{clip}\left(\boldsymbol{\delta}^{[t]} + \alpha \nabla_{\boldsymbol{\delta}} L\left(y, \hat{f}(\mathbf{x} + \boldsymbol{\delta}^{[t]} | \boldsymbol{\theta})\right), [-\epsilon, \epsilon]\right)$$

PROJECTED GRADIENT DESCENT

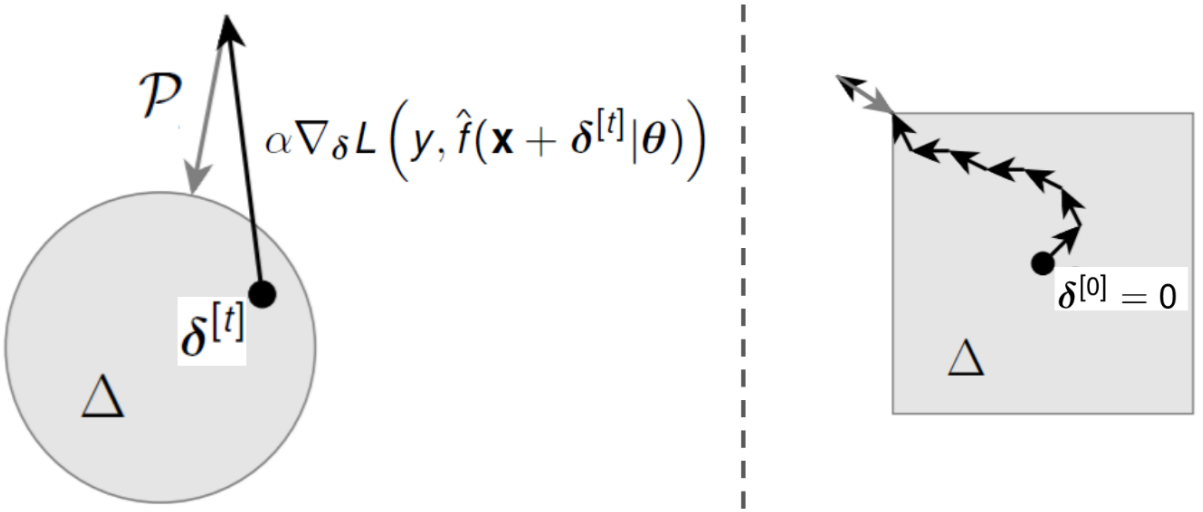


Figure: *Left*: $\delta^{[t]} + \alpha \nabla_{\delta} L(y, \hat{f}(\mathbf{x} + \delta^{[t]}|\theta))$ is projected back to the perturbation set Δ using $\mathcal{P}$. Here, Δ is $\mathcal{B}_{\epsilon}^{2}$. *Right*: Multiple steps of PGD are shown in case of $\Delta = \mathcal{B}_{\epsilon}^{\infty}$; the projection must be only executed in the last step (Kolter & Madry, 2019). (Note: A variant of PGD known as *normalized* PGD is shown here. This is why each step has the same size. See Kolter et al. (2019) for details.)

FAST GRADIENT SIGN METHOD

- Fast Gradient Sign Method (FGSM) is a special case of PGD when $\Delta = \mathcal{B}_\epsilon^\infty$ and $\alpha \to \infty$.
- As before, the projection of an arbitrary vector $\mathbf{z}$ onto $\mathcal{B}_\epsilon^\infty$ is $\mathcal{P}(\mathbf{z}) = \mathit{clip}(\mathbf{z}, [-\epsilon, \epsilon])$.
- As $\alpha \to \infty$, the elements of $\boldsymbol{\delta}$ are either set to $-\epsilon$ or ϵ depending on the sign of the corresponding component of the gradient.
- Thus, the FGSM algorithm computes $\boldsymbol{\delta}$ as

$$\boldsymbol{\delta} = \epsilon \mathit{sign}\left(\nabla_{\mathbf{x}} L\left(y, \hat{f}(\mathbf{x}|\boldsymbol{\theta})\right)\right)$$

- Note that for FGSM we only conduct one calculation and not multiple gradient descent steps.
- Furthermore, because we usually initialize $\boldsymbol{\delta}^{[0]}$ as 0

$$\nabla_{\boldsymbol{\delta}} L\left(y, \hat{f}(\mathbf{x} + \boldsymbol{\delta}|\boldsymbol{\theta})\right) = \nabla_{\mathbf{x}} L\left(y, \hat{f}(\mathbf{x}|\boldsymbol{\theta})\right)$$

FAST GRADIENT SIGN METHOD

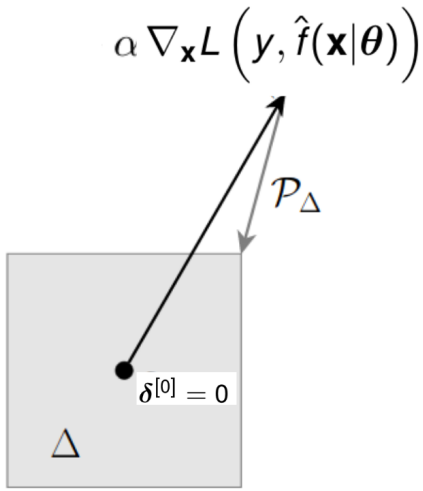


Figure: δ is obtained by setting each element of $\nabla_{\mathbf{x}} L\left(y, \hat{f}(\mathbf{x}|\theta)\right)$ to $-\epsilon$ or ϵ depending on its sign. Note that this slightly changes the direction of the step that is taken (Kolter & Madry, 2019).

- Implicitly, the (element-wise) *sign* function is simply a way to constrain the size of the "step" that we take in the direction of the gradient. It is basically equal to a projection of the step back on Δ (which is $\mathcal{B}_{\epsilon}^{\infty}$).

FAST GRADIENT SIGN METHOD

- Note: The optimal attack against the linear binary classifier we saw in the last section was also FGSM!

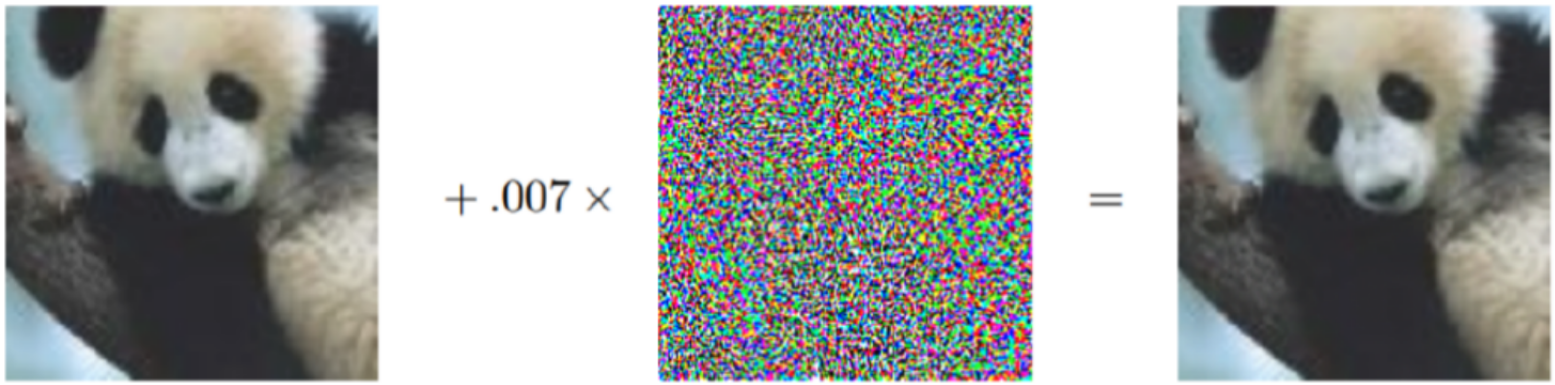


Figure: By adding an imperceptibly small vector whose elements are equal to the sign of the elements of the gradient of the cost function with respect to the input, GoogLeNet's classification of the image was changed from 'panda' to 'gibbon'. In this example, the ϵ is 0.007 (Goodfellow, 2017).

REFERENCES

Zico Kolter and Aleksander Madry (2019)

Adversarial Robustness - Theory and Practice

https://adversarial-ml-tutorial.org/

Ian Goodfellow, Yoshua Bengio and Aaron Courville (2016)

Deep Learning

http://www.deeplearningbook.org/

Ian Goodfellow (2017)

Lecture 16 | Adversarial Examples and Adversarial Training

https://www.youtube.com/watch?v=CIfsB_EYsVI

Ian GoodfellowNicolas PapernotSandy HuangRocky DuanPieter AbbeelJack Clark (2017)

Attacking Machine Learning with Adversarial Examples

https://openai.com/blog/adversarial-example-research/

REFERENCES

Anh Nguyen, Jason Yosinski and Jeff Clune (2015)

Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images

https://arxiv.org/abs/1412.1897

Alex Krizhevsky, Ilya Sutskever and Geoffrey E. Hinto (2012)

ImageNet Classification with Deep Convolutional Neural Networks. NIPS.

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks

Maxence Prevost (2018)

Adversarial ResNet50

http://arxiv.org/abs/1207.0580

Mahmood Sharif, Sruti Bhagavatula and Lujo Bauer (2016)

Accessorize to a Crime: Real and Stealthy Attacks onState-of-the-Art Face Recognition. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security.

https://dl.acm.org/doi/10.1145/2976749.2978392

REFERENCES

Goodfellow, Shlens (2014)

Explaining and Harnessing Adversarial Examples

https://github.com/maxpv/maxpv.github.io/blob/master/notebooks/Adversarial_ResNet50.ipynb

Papernot , McDaniel, Goodfellow, Jha, Celik, Swamy (2016)

Practical Black-Box Attacks against Machine Learning

https://arxiv.org/abs/1602.02697

Athalye , Engstrom, Ilyas, Kwok (2017)

Synthesizing Robust Adversarial Examples

https://arxiv.org/abs/1707.07397

Tom B. Brown and Catherine Olsson, Research Engineers, Google Brain Team (2018)

Introducing the Unrestricted Adversarial Examples Challenge

https://ai.googleblog.com/2018/09/introducing-unrestricted-adversarial.html