Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem(CVPR 2019 Oral)

Juhyeong Kim

SKKU Visual Computing Lab

wngud0811@naver.com

September 22, 2020

Overview

Problem

- For many popular deep learning models,
- High confidence output can be made for out-of-distribution data.
- Also, often produce over-confident predictions in original tasks.

Importantce of uncertainty Quantification

- Deep learning models may fail in the case of noisy data or out-of-distribution data.

Train time

rabbit: **0.8** / wolf: 0.2

rabbit: 0.3 / wolf: **0.7**

Test time

rabbit: 0.1 / wolf: **0.9**

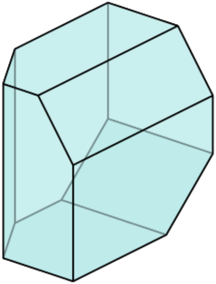
1. Useful Solutions
   - Dropout as a bayesian approximation: Representing model uncertainty in deep learning.(ICML 2016)
   - On calibration of modern neural networks.(ICML 2017)
   - Simple and scalable predictive uncertainty estimation using deep ensembles.(NIPS 2017)

   - Also, there exist classifiers which is not being confident in areas where one has never seen data.(ex. RBF networks)

Affine

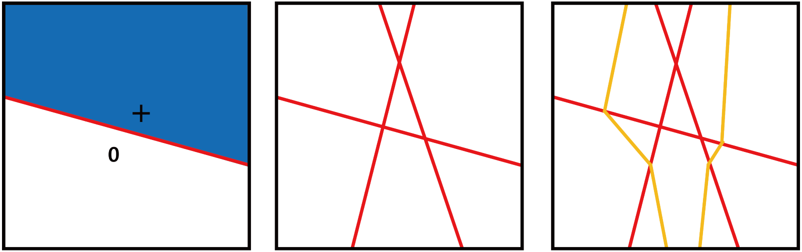
1. For a function $f : \mathrm{U} \to \mathrm{V}$, we call it **linear** if,
   - $f(x+y) = f(x) + f(y)$
   - $\alpha f(x) = f(\alpha x)$
   - For $\forall x, y \in \mathrm{U}$
2. For a function $f : \mathrm{U} \to \mathrm{V}$, we call it **affine** if,
   - $\exists b \in \mathrm{V}$ such that $f = \bar{f} + b$ and $\bar{f}$ is linear function.
3. Linear function $\subset$ Affine function

Polytope



- Polygon in arbitary dimensional space.

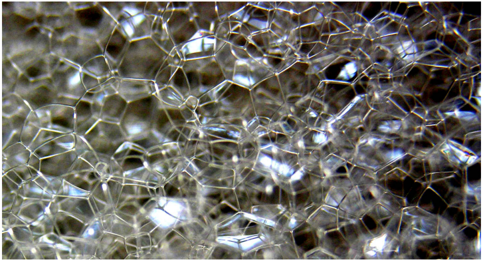
Polytope



- Let each red line in figure is equal to $\mathrm{w}_{\cdot,i}^{\mathsf{T}} x + b_i = 0$ at first layer.
- If the red line is separation at first layer, seperation at second layer is like orange line.
- Each ReLU activation can be considered as dividing input space into two parts.

Piecewise Affine

- A function f is called **piecewise affine** if there exists finite set of polytopes $\{\mathrm{Q}_r\}_{r=1}^{M}$ such that $\bigcup_{r=1}^{M} \mathrm{Q}_r = \mathbb{R}^d$ and f is affine function in each $\mathrm{Q}_r,\ \ r = 1, 2, ..., M$.



Neural Network

- Let $\mathrm{W}^{(l)} \in \mathbb{R}^{\mathrm{n}_l \times \mathrm{n}_{l-1}}$ and $\mathrm{b}^{(l)} \in \mathbb{R}^{\mathrm{n}_l}$ for $l = 1, 2, ..., L$ are parameters.
- Then, ReLU Network can be expressed as:
$f^{(k)}(x) = \mathrm{W}^{(k)}\mathsf{ReLU}\big(f^{(k-1)}\big) + b^{(k)}, \quad k = 1, ..., L$

- Define a function

$$\Sigma^{(l)}(x)_{ij} = \begin{cases} 1 & \text{if } i = j \text{ and } f_i^{(l)}(x) > 0, \\ 0 & \text{else.} \end{cases}$$

Neural Network

- $f^{(k)}(x) = \mathrm{W}^{(k)}\Sigma^{(k-1)}(x)\Big(\mathrm{W}^{(k-1)}\Sigma^{(k-2)}(x)$
$$\times \Big(\dots \Big(\mathrm{W}^{(1)}x + b^{(1)}\Big) \dots \Big) + b^{(k-1)}\Big) + b^{(k)}$$

$$= \mathrm{V}^{(k)} \cdot x + a^{(k)}$$

for $k = 1, \dots, \mathrm{L}$ with $\mathrm{V}^{(k)} \in \mathbb{R}^{n_k \times d}$ and $a^{(k)} \in \mathbb{R}^{n_k}$

- $$f^{(k)}(x) = \mathrm{W}^{(k)}\Sigma^{(k-1)}(x)\Big(\mathrm{W}^{(k-1)}\Sigma^{(k-2)}(x) \times \Big(\ \dots\ \Big(\mathrm{W}^{(1)}x + b^{(1)}\Big)\ \dots\ \Big) + b^{(k-1)}\Big) + b^{(k)}$$

$$= \ \mathrm{V}^{(k)} \cdot x + a^{(k)}$$

for $k = 1, \dots, \mathrm{L}$ with $\mathrm{V}^{(k)} \in \mathbb{R}^{n_k \times d}$ and $a^{(k)} \in \mathbb{R}^{n_k}$

$$\mathrm{V}^{(k)} = \mathrm{W}^{(k)}\left(\prod_{l=1}^{k-1}\Sigma^{(k-l)}(x)\mathrm{W}^{(k-l)}\right)$$

$$a^{(k)} = b^{(k)} + \sum_{l=1}^{k-1}\left(\prod_{m=1}^{k-l}\mathrm{W}^{(k+1-m)}\Sigma^{(k-m)}(x)\right)b^{(l)}$$

Conclusion

- ReLU network = piecewise affine function + softmax.
- Similary, many activation functions, including leaky ReLU can be shown to be piecewise affine.
- Fully connected, Convolution with average/max pooling, Residual layers are included.

Implications

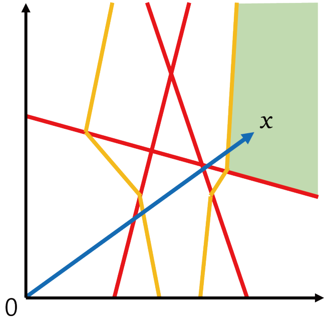
- Entire ReLU network is just simple softmax classifier when domain is restricted to each polytope.

Lemma 3.1

Let ReLU-classifier divide input space to set of linear regions $\{\mathrm{Q}_l\}_{l=1}^{R}$.

Then, any point $x \in \mathbb{R}^d$ in input space, there exist $a \in \mathbb{R}$ with $a > 0$,

such that $\beta x \in \mathrm{Q}_t$ for all $\beta \geq a,\ t \in \{1, ..., \mathrm{R}\}$.



Theorem 3.1

If the softmax input of ReLU network is piecewise affine function,

for almost any input x and any threshold level $0 < t < 1$,

there exists some constant $\alpha > 0$,

$$\frac{e^{f_k(\alpha x)}}{\sum_{r=1}^{\mathrm{K}} e^{f_r(\alpha x)}} \geq t$$

Theorem 3.1

And also, $$\lim_{\alpha \to \infty} \frac{e^{f_k(\alpha x)}}{\sum_{r=1}^{\mathrm{K}} e^{f_r(\alpha x)}} = 1$$

Limitation

- Author assumed $\mathbb{R}^d$ as input space, but many applications assume $[0, 1]^d$ as input space.
- In these cases, theorem does not directly applied.
- But empirically, training in bounded domain shows same problems.

- Author proposed two methods:
  - Confidence Enhancing Data Augmentation(CEDA)
  - Adversarial Confidence Enhanced Training(ACET)

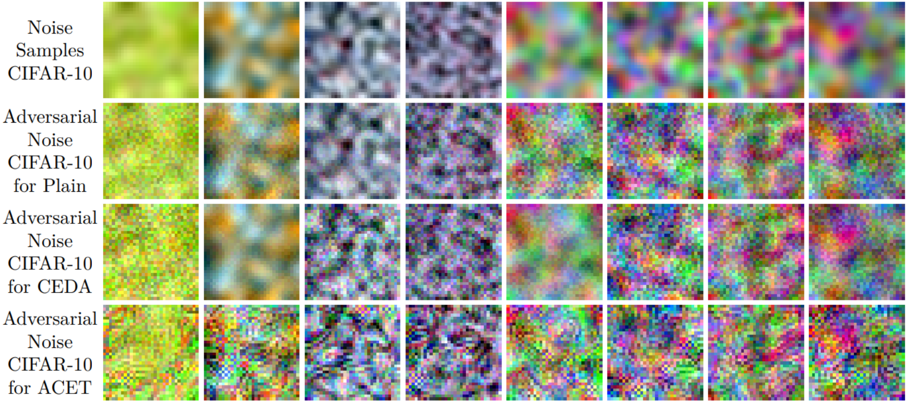
Confidence Enhancing Data Augmentation(CEDA)

- In image classification problem:
  1. Assume the type of out-of-distribution data in domain. (For example, Uniform distribution on $[0,1]^{c\times w\times h}$)
  2. Sample random noise from distribution.
  3. Forward noise image to model and get sofmax probabiliies.
  4. Minimize the maximum log sofmax value.

$$\Rightarrow \text{Regularizer} \quad \lambda\, \mathbb{E}\left[\max_{l=1,\ldots,K} \log\left(\frac{e^{f_l(z)}}{\sum_{k=1}^{K} e^{f_l(z)}}\right)\right]$$

Adversarial Confidence Enhanced Training(ACET)

- However, CEDA may need multiple sampling and forward which will yield high comutational costs.
- Inspired by adversarial training, author proposed more easier and more scalable method.
- Modified regularizer $\lambda \, \mathbb{E}\left[\max_{\|u-z\|_p \leq \epsilon} \max_{l=1,\ldots,K} \log\left(\frac{e^{f_l(z)}}{\sum_{k=1}^{K} e^{f_l(z)}} \right) \right]$


Noise Samples CIFAR-10
Adversarial Noise CIFAR-10 for Plain
Adversarial Noise CIFAR-10 for CEDA
Adversarial Noise CIFAR-10 for ACET

Trained on **MNIST**	Plain (TE: **0.51%**)			CEDA (TE: 0.74%)			ACET (TE: 0.66%)		
	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
MNIST	**0.991**	–	–	0.987	–	–	0.986	–	–
FMNIST	0.654	0.972	0.121	0.373	0.994	0.027	**0.239**	**0.998**	**0.003**
EMNIST	0.821	0.883	0.374	0.787	0.895	0.358	**0.752**	**0.912**	**0.313**
grayCIFAR-10	0.492	0.996	0.003	0.105	**1.000**	**0.000**	**0.101**	**1.000**	**0.000**
Noise	0.463	0.998	0.000	**0.100**	**1.000**	**0.000**	**0.100**	**1.000**	**0.000**
Adv. Noise	1.000	0.031	1.000	**0.102**	**0.998**	**0.002**	0.162	0.992	0.042
Adv. Samples	0.999	0.358	0.992	0.987	0.549	0.953	**0.854**	**0.692**	**0.782**
Trained on **SVHN**	Plain (TE: **3.53%**)			CEDA (TE: 3.50%)			ACET (TE: 3.52%)		
	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
SVHN	**0.980**	–	–	0.977	–	–	0.978	–	–
CIFAR-10	0.732	0.938	0.348	0.551	0.960	0.209	**0.435**	**0.973**	**0.140**
CIFAR-100	0.730	0.935	0.350	0.527	0.959	0.205	**0.414**	**0.971**	**0.139**
LSUN CR	0.722	0.945	0.324	0.364	0.984	0.084	**0.148**	**0.997**	**0.012**
Imagenet-	0.725	0.939	0.340	0.574	0.955	0.232	**0.368**	**0.977**	**0.113**
Noise	0.720	0.943	0.325	**0.100**	**1.000**	**0.000**	**0.100**	**1.000**	**0.000**
Adv. Noise	1.000	0.004	1.000	0.946	0.062	0.940	**0.101**	**1.000**	**0.000**
Adv. Samples	1.000	0.004	1.000	0.995	0.009	0.994	**0.369**	**0.778**	**0.279**
Trained on **CIFAR-10**	Plain (TE: 8.87%)			CEDA (TE: 8.87%)			ACET (TE: **8.44%**)		
	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
CIFAR-10	**0.949**	–	–	0.946	–	–	0.948	–	–
SVHN	0.800	0.850	0.783	0.327	0.978	0.146	**0.263**	**0.981**	**0.118**
CIFAR-100	0.764	0.856	0.715	**0.761**	**0.850**	0.720	0.764	0.852	**0.711**
LSUN CR	0.738	**0.872**	**0.667**	**0.735**	0.864	0.680	0.745	0.858	0.677
Imagenet-	0.757	0.858	0.698	0.749	0.853	0.704	**0.744**	**0.859**	**0.678**
Noise	0.825	0.827	0.818	**0.100**	**1.000**	**0.000**	**0.100**	**1.000**	**0.000**
Adv. Noise	1.000	0.035	1.000	0.985	0.032	0.983	**0.112**	**0.999**	**0.008**
Adv. Samples	1.000	0.034	1.000	1.000	0.014	1.000	**0.633**	**0.512**	**0.590**
Trained on **CIFAR-100**	Plain (TE: **31.97%**)			CEDA (TE: 32.74%)			ACET (TE: 32.24%)		
	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
CIFAR-100	**0.751**	–	–	0.734	–	–	0.728	–	–
SVHN	0.570	0.710	0.865	0.290	0.874	0.410	**0.234**	**0.912**	**0.345**
CIFAR-10	0.560	0.718	0.856	0.547	0.711	0.855	**0.530**	**0.720**	**0.860**
LSUN CR	0.592	0.690	0.887	0.581	0.678	0.887	**0.554**	**0.698**	**0.881**
Imagenet-	0.531	0.744	0.827	0.504	0.749	0.808	**0.492**	**0.752**	**0.819**
Noise	0.614	0.672	0.928	0.010	1.000	0.000	**0.010**	**1.000**	**0.000**
Adv. Noise	1.000	0.000	1.000	0.985	0.015	0.985	**0.013**	**0.998**	**0.003**
Adv. Samples	0.999	0.010	1.000	0.999	0.012	1.000	**0.863**	**0.267**	**0.975**

Summary

- ReLU networks always make high confidence predictions far away from training data.
- Temperature rescaling and reject option does not help.
- Using modified training similar to adversarial training, we can reduce high confidence problem at out-of-distribution data.