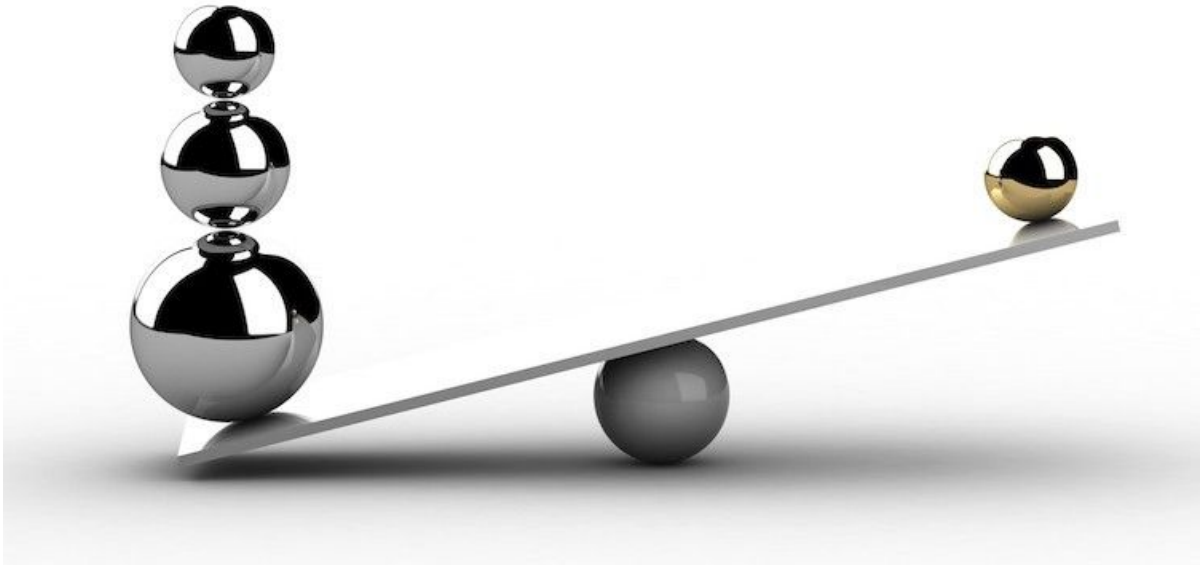
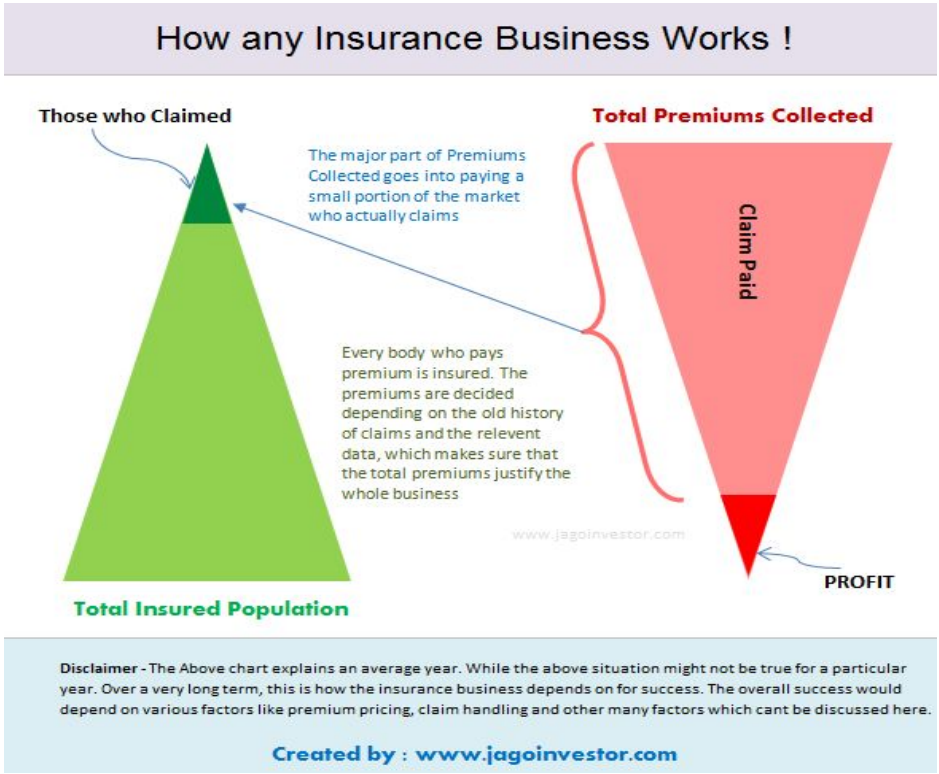
Generative Adversarial Networks (GANs) for data imbalance

Armando Vieira & Javier Rodriguez Zaurin

Data imbalance in insurance – why it matters?



Data imbalance in insurance - why it matters?



1.

Myths Around Insurance modeling

Statistics are there for a reason...

- Data follows **Poisson** / overdispersed Poisson / Gamma distributions
- There are **well defined quantities** we can always compute
  - mean
  - standard deviation
  - skewness
- Data is **stationary** (i.e. The rate at which events occur is constant)
- Events are **independent** (i.e. The occurrence of one event does not affect the probability that a second event will occur
- **Pooling** solves the issues

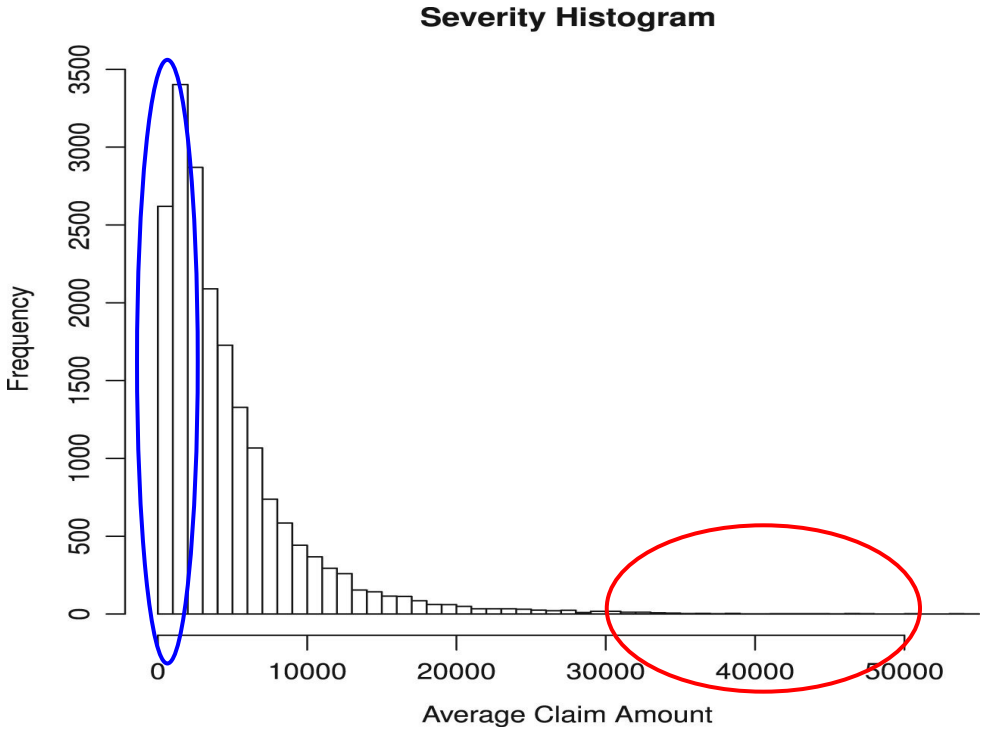
2.

The mean is meaningless

In Insurance only two types of clients matter

- ❏ Clients that **never** claims
- ❏ Clients with **large** claims

By focusing on the extremes we can better model the important factors



3.

Imbalance Datasets

Techniques to handle imbalance data

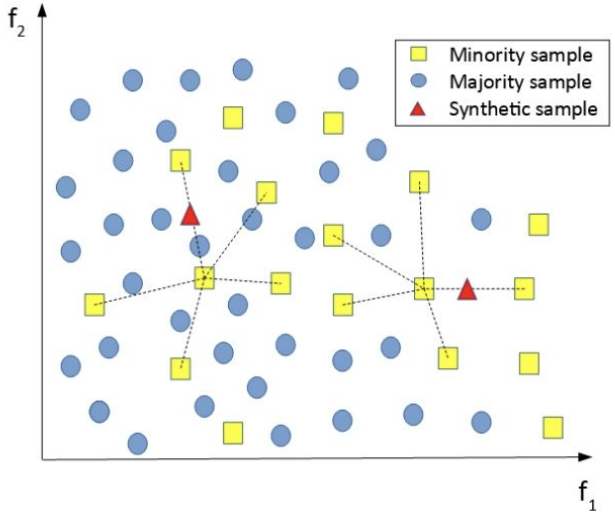
(a) SMOTE

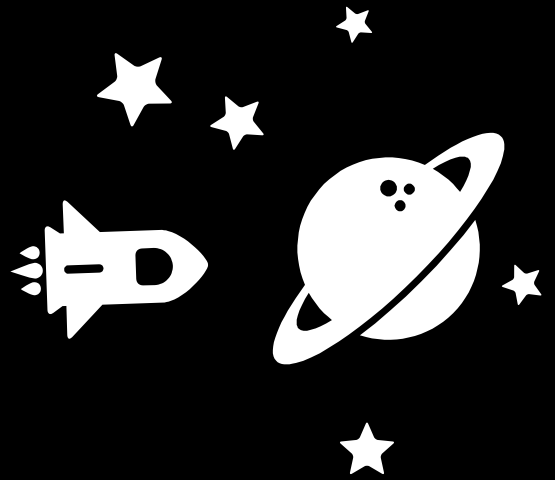
(b) ADASYN

(c) VAE based Method

(d) GAN based Method

SMOTE + Tomek

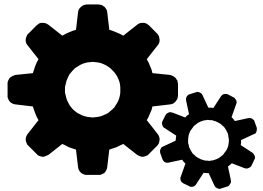




GANs

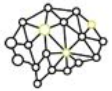
Generative Adversarial Networks

GANs



3. Deep Learning

Deep Learning Achieves Photorealistic Image Generation



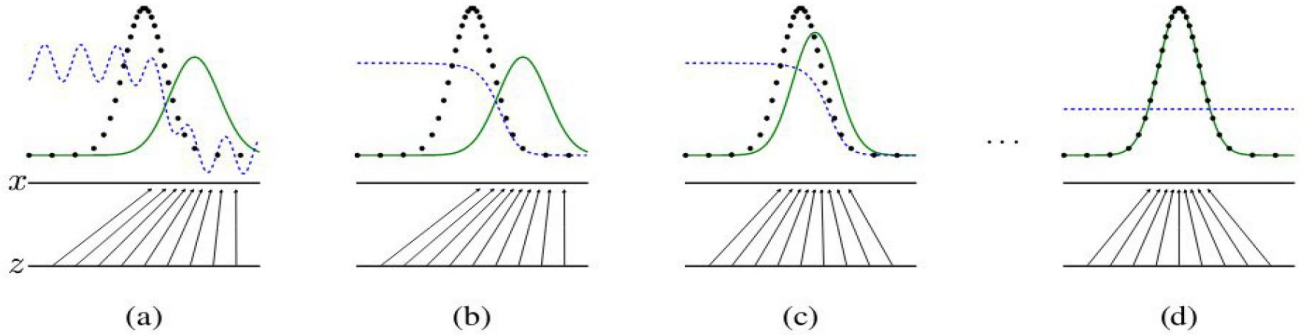
Deep learning can recognize and generate images. Early results were blurry and unconvincing, as seen on the left. The latest results approach photorealism, as seen on the right.

Fake Images Generated Using Deep Learning

2016

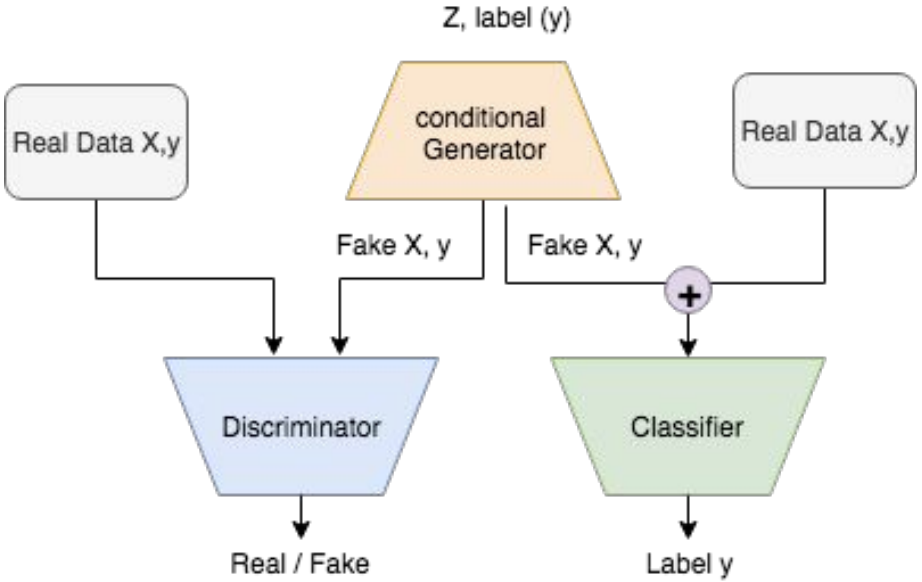
2017

Generative Adversarial Nets

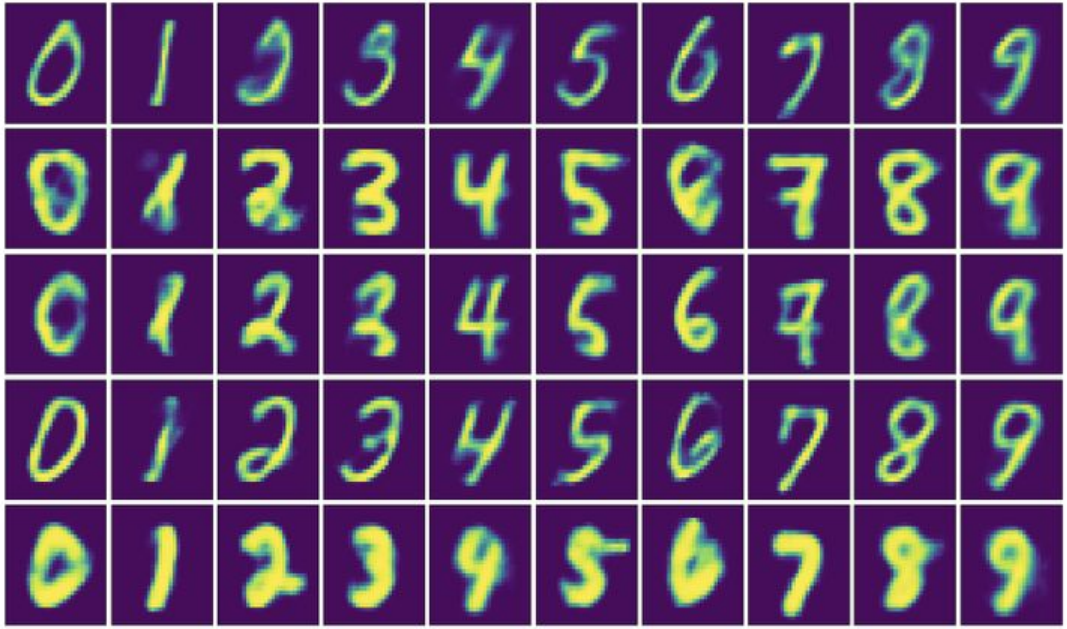


Training stages of a GAN
Black dotted: True data
Green solid: Generated data
Blue dotted: Discriminator loss

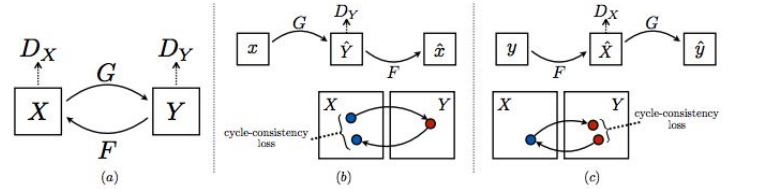
Conditional GAN



Conditional GAN



CycleGAN

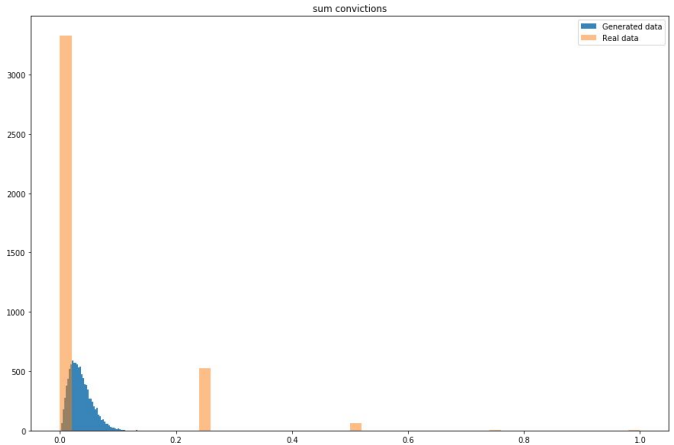


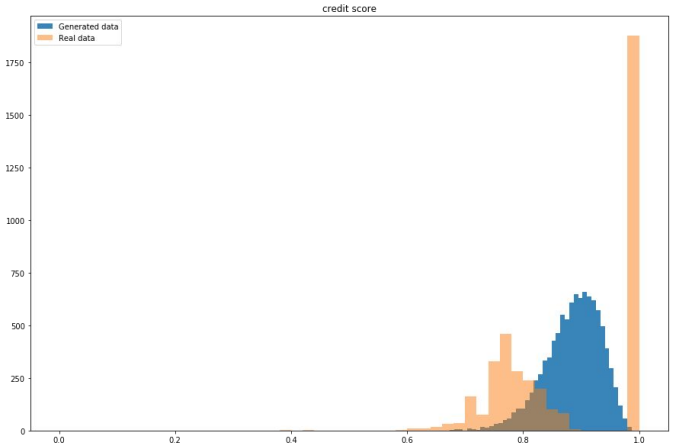
Horse → Zebra

or ...

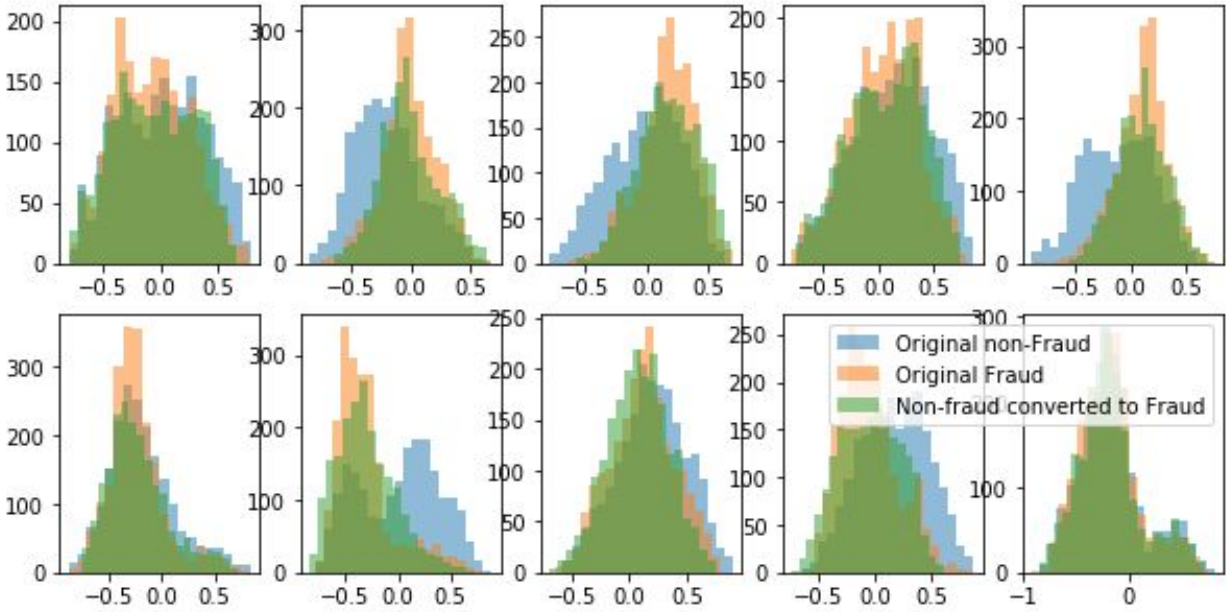
No Claim → Claim

GANs were very effective for images, but how about tabular data?

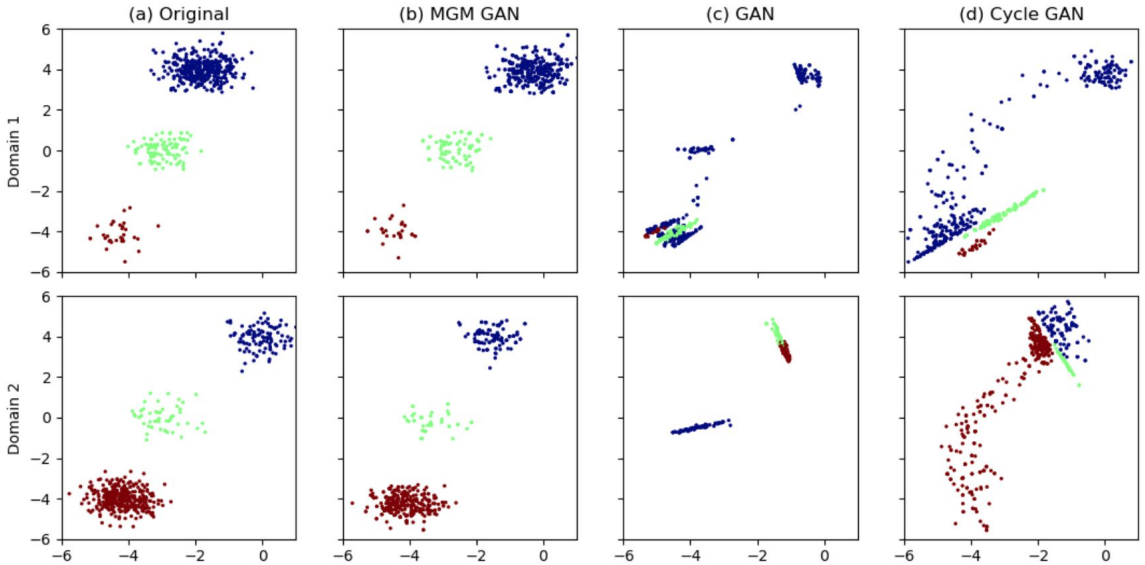




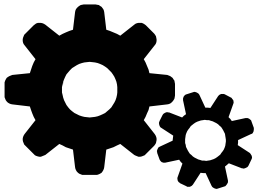
CycleGAN for fraud detection + sparse Autoencoders



Topology preserving CycleGAN



Results

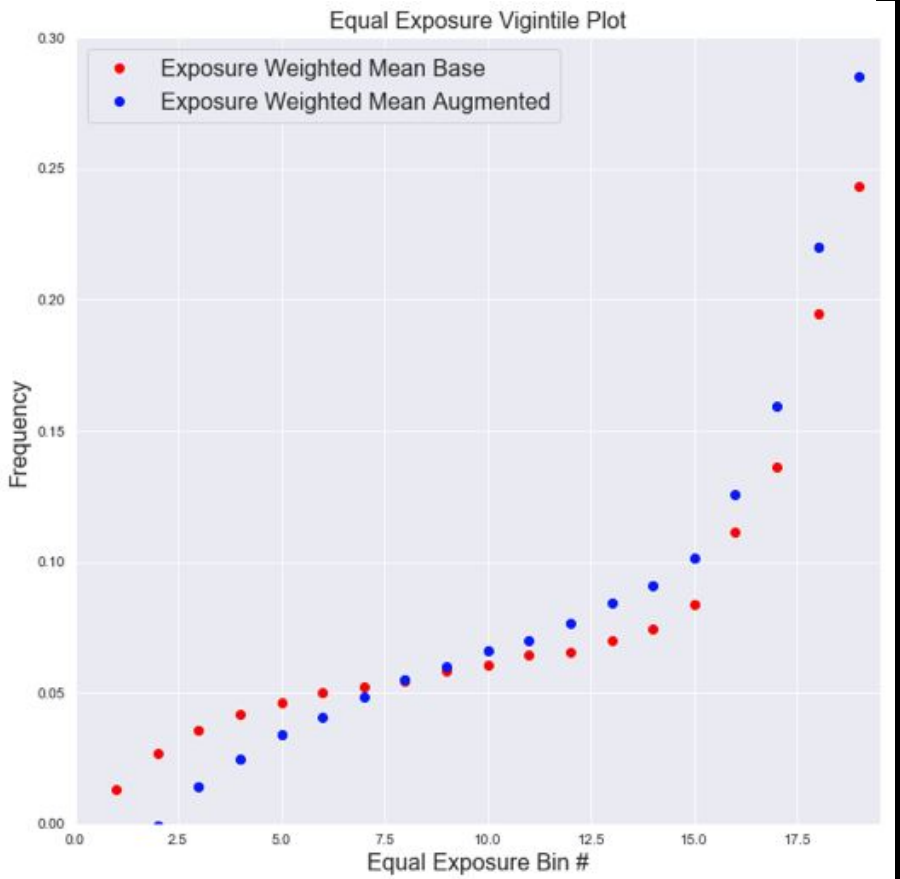


	Precision	Recall	F1-score
No augmentation	0.88	0.77	0.79
SMOTE	0.94	0.79	0.85
ADASYNE	0.79	0.76	0.77
cGAN	0.90	0.85	0.87
cycleGAN	0.92	0.83	0.87

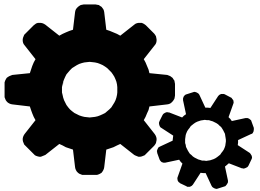
N_s	No. of misclassifications for MNIST. Test error rate for others.				
	Supervised	DCGAN	W-DCGAN	DCGAN-10	BayesGAN
MNIST	N=50k, $D = (28, 28)$	14	15	114	32
20	—	1823 ± 412	1687 ± 387	$\mathbf{1087 \pm 564}$	1432 ± 487
50	—	453 ± 110	490 ± 170	$\mathbf{189 \pm 103}$	332 ± 172
100	2134 ± 525	128 ± 11	156 ± 17	97 ± 8.2	$\mathbf{79 \pm 5.8}$
200	1389 ± 438	95 ± 3.2	91 ± 5.2	78 ± 2.8	$\mathbf{74 \pm 1.4}$
CIFAR-10	N=50k, $D = (32, 32, 3)$	18	19	146	68
1000	63.4 ± 2.6	58.2 ± 2.8	57.1 ± 2.4	$\mathbf{31.1 \pm 2.5}$	32.7 ± 5.2
2000	56.1 ± 2.1	47.5 ± 4.1	49.8 ± 3.1	29.2 ± 1.2	$\mathbf{26.2 \pm 4.8}$
4000	51.4 ± 2.9	40.1 ± 3.3	38.1 ± 2.9	27.4 ± 3.2	$\mathbf{23.4 \pm 3.7}$
8000	47.2 ± 2.2	29.3 ± 2.8	27.4 ± 2.5	25.5 ± 2.4	$\mathbf{21.1 \pm 2.5}$
SVHN	N=75k, $D = (32, 32, 3)$	29	31	217	81
500	53.5 ± 2.5	31.2 ± 1.8	29.4 ± 1.8	27.1 ± 2.2	$\mathbf{22.5 \pm 3.2}$
1000	37.3 ± 3.1	25.5 ± 3.3	25.1 ± 2.6	18.3 ± 1.7	$\mathbf{12.9 \pm 2.5}$
2000	26.3 ± 2.1	22.4 ± 1.8	23.3 ± 1.2	16.7 ± 1.8	$\mathbf{11.3 \pm 2.4}$
4000	20.8 ± 1.8	20.4 ± 1.2	19.4 ± 0.9	14.0 ± 1.4	$\mathbf{8.7 \pm 1.8}$
CelebA	N=100k, $D = (50, 50, 3)$	103	98	649	329
1000	53.8 ± 4.2	52.3 ± 4.2	51.2 ± 5.4	47.3 ± 3.5	$\mathbf{33.4 \pm 4.7}$
2000	36.7 ± 3.2	37.8 ± 3.4	39.6 ± 3.5	31.2 ± 1.8	$\mathbf{31.8 \pm 4.3}$
4000	34.3 ± 3.8	31.5 ± 3.2	30.1 ± 2.8	$\mathbf{29.3 \pm 1.5}$	29.4 ± 3.4
8000	31.1 ± 4.2	29.5 ± 2.8	27.6 ± 4.2	26.4 ± 1.1	$\mathbf{25.3 \pm 2.4}$

Insurance

Double lift chart with data Augmentation + extreme modeling

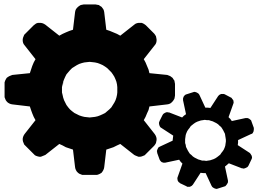


Implementation Details



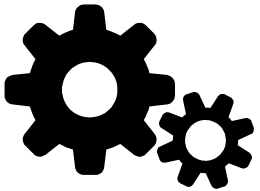
- Data has to be converted into a dense representation
- Domains have to be consistently related - retain same semantic features.
- Loads of tricks (weight clipping, regularizers, training schedule)
- Consistency over time (i.e. stationary)
- Avoid mode collapsing
- Diversity enforcing with conditional batch normalization and noise

How to train GANs



- Small (decreasing) learning rate
- Different optimizers for encoder and generator
- Train generator multiple times for each discriminator
- Batch normalization and Instance Batch Normalization

Conclusions:



- Focus on the **extremes** (less data, better results)
- Use **augmentation** to alleviate imbalance data
- **GANS can be effective** under some conditions