How Real is Real?

Quantitative and Qualitative comparison of GANs and supervised-learning classifiers.

Riccardo Verzeni (rverzeni@stanford.edu), Jacqueline Yau (jyau@stanford.edu)

Predictions

- **Motivation**
  - Investigate how well supervised learning classifiers, trained on real MNIST images, generalize on GANs synthetic images.
  - Investigate how well a modified GANs semi-supervised learning classifier would perform over real MNIST images.
- **Approach**
  - We built four supervised learning classifiers of increasing complexity; we trained them over real MNIST images and compare their results against real and synthetic test datasets.
  - We built a semi-supervised learning classifier modifying a GANs discriminator and trained it using a combination of unlabeled synthetic and labeled real MNIST images.[1]
- **Results**
  - The various supervised classifier seemed to generalize reasonably well on GANs synthetic MNIST images.
  - The semi-supervised learning classifier appeared to perform worse when training on an equally split labeled real / unlabeled synthetic MNIST images dataset than when training on a fully labeled real MNIST images dataset (given an identical number of labeled real samples in both datasets).

Data

MNIST grayscale images (28 pixels x 28 pixels, 1 channel)

- **The real MNIST images**
  The dataset has been downloaded using Keras APIs (60000 training examples, 10000 test examples)[2].
- **The synthetic MNIST images**
  The dataset (6336 unlabeled examples) has been generated using a Deep Convolutional Generative Adversarial Networks[3], which has been previously trained separately.
  - We manually labeled 1000 of the synthetic images acting as ground truth creating a synthetic test dataset for the quantitative accuracy comparison.

Features

- Input features are the pixels that make up the MNIST grayscale image.
- The neural networks derive new features in the hidden layers, and the convolutional ones extract additional features in the convolutions.
- Attention visualization shows where the CNN focused.

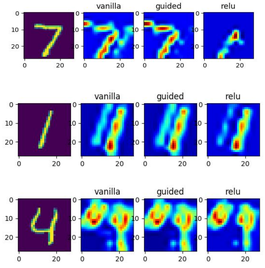

Fig.1 Class Activation attention for cnn-5l.

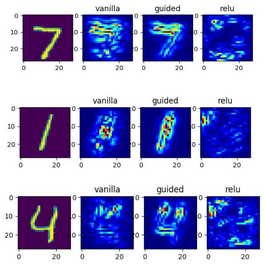

Fig.2 Saliency attention for cnn-5l.

Results

epochs

model	5		50		100	
	Real	Synthetic	Real	Synthetic	Real	Synthetic
linear softmax	0.923 / 0.919	**n/a** / 0.923	0.9359 / 0.921	**n/a** / 0.921	0.9377 / 0.916	**n/a** / 0.913
fcnn	0.989 / 0.978	**n/a** / 0.957	0.999 / 0.979	**n/a** / 0.969	0.999 / 0.983	**n/a** / 0.962
cnn-4l	0.995 / 0.986	**n/a** / 0.971	1.000 / 0.989	**n/a** / 0.970	1.000 / 0.989	**n/a** / 0.974
cnn-5l	0.996 / 0.994	**n/a** / 0.972	0.998 / 0.988	**n/a** / 0.973	0.998 / 0.985	**n/a** / 0.969
semi-gan*	1.000 / 1.000; 0.9954 / 0.973	**n/a**	1.000 / 1.000 ; 1.000 / 0.979	**n/a**	1.000 / 1.000; 1.000 / 0.977	**n/a**
semi-gan**	0.966 / 0.889; 0.974 / 0.857	**n/a**	1.000 / 0.962 ; 1.000 / 0.932	**n/a**	1.000 / 0.968; 1.000 / 0.933	**n/a**

- Training accuracy / test accuracy results against 54000 sample training dataset (6000 validation) and **1000 real** and **1000 synthetic** MNIST images **test datasets**.
- *Training accuracy / test accuracy (discriminator layer ; label classifier layer) results against the combination of 5400 labeled real and 0 unlabeled synthetic sample training dataset and **1000 real** MNIST images **test dataset**.
- **Training accuracy / test accuracy (discriminator layer ; label classifier layer) results against the combination of 5400 labeled real and 5400 unlabeled synthetic sample training dataset and **1000 real** MNIST images **test dataset**.

Discussion

- The test accuracy obtained by the various supervised learning classifiers over synthetic images was overall slightly worse but comparable with the real images, where the most complex convolutional NN (cnn-5l) performed best.
- The test accuracy obtained by training the semi-supervised learning classifier on the combined labeled real / unlabeled synthetic dataset was worse than the one obtained by training it on the fully labeled real dataset.
- We expected the CNN to perform better than any other classifier since it is considered state-of-the-art for the MNIST digit classification problem[4]. That has been indeed the case.
- The CNN performed well because the convolution was able to extract meaningful features that were effective in identifying and then classifying the number in the MNIST image, and the 5-layer CNN performed better than the 4-layer one since the extra convolution layer with 64 filters was able to find additional features that the first 32 filter convolution layer could not.
- For the semi-supervised learning classifier, we expected that the addition of the unlabeled synthetic data would have improved the label classifier accuracy on real MNIST images.
- Contrary to expectation, the semi-supervised learning classifier did not perform well. This is probably because, due to time constraints, we used previously generated synthetic images as unlabeled data, instead of building also a generator and training it along with the modified discriminator as shown by Tim S. and All[1]. As a result the features extracted by the unlabeled data might not have been as relevant as they would have been if coming from the same distribution of the real data that the generator would have reproduced.

Model

- **Linear Softmax classifier**
- **Fully connected Neural Network (fcnn)**
  - Flatten and ReLU (256 neurons)
  - Softmax output (10 neurons)
- **Convolutional Neural Network 4l (cnn-4l)**
  - Convolution layer with 32 filters with kernel size 5
  - Max Pooling with pool size 2 x 2
  - Flatten and ReLU (128 neurons)
  - Softmax output (10 neurons)
- **Convolutional Neural Network 5l (cnn-5l)**
  - Same as cnn-4l but with an additional convolution layer (64 filters)
- **Semi-Supervised modified GANs discriminator (semi-gan)**
  - Same as (cnn-5l) but with two output layers: (1 neuron Sigmoid), (10+1 neurons Softmax)

Softmax

$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}}$$

Cross entropy loss

$$\mathcal{H}(p,q) = -\sum_{x} p(x) \log q(x)$$

Sigmoid

$$g(x) = \frac{1}{1+e^{-x}}$$

Future work

- Since overfitting seems to occur for the CNN, one thing to do would be to add some regularization, such as a dropout layer.
- Another idea we would like to try is implementing a ResNet to see if it could perform even better than CNN.
- It would be interesting to build a complete semi-supervised learning GANs classifier, such as the one suggested[1] and see if that would improve the results obtained in our partial attempt.

References

[1] Tim S., Ian G., Wojciech Z., Vicki C., Alec R., and Xi C. Improved techniques for training gans. 2016. https://arxiv.org/pdf/1606.03498.pdf .
[2] Tensorflow keras high-level apis. https://www.tensorflow.org/guide/keras .
[3] A tensorflow implementation of “Deep Convolutional Generative Adversarial Networks” https://github.com/carpedm20/DCGAN-tensorflow .
[4]Xuan Y., Jing P. MDig: Multi-digit Recognition using Convolutional Neural Network on Mobile.

Original table without digits being normalised for Reference

	epochs					
	5		50		100	
	Real	Synthetic	Real	Synthetic	Real	Synthetic
linear softmax	0.923 / 0.919	**n/a** / 0.923	0.9359 / 0.921	**n/a** / 0.921	0.9377 / 0.916	**n/a** / 0.913
fcnn	0.9891 / 0.978	**n/a** / 0.957	0.9991 / 0.979	**n/a** / 0.969	0.9996 / 0.983	**n/a** / 0.962
cnn-4l	0.9959 / 0.986	**n/a** / 0.971	1.0 / 0.989	**n/a** / 0.970	1.0 / 0.989	**n/a** / 0.974
cnn-5l	0.9963 / 0.994	**n/a** / 0.972	0.9984 / 0.988	**n/a** / 0.973	0.9984 / 0.985	**n/a** / 0.969
semi-gan*	1.0 / 1.0 ; 0.9954 / 0.973	**n/a**	1.0 / 1.0 ; 1.0 / 0.979	**n/a**	1.0 / 1.0 ; 1.0 / 0.977	**n/a**
semi-gan**	0.9666 / 0.889 ; 0.9743 / 0.857	**n/a**	1.0 / 0.962 ; 1.0 / 0.932	**n/a**	1.0 / 0.968 ; 1.0 / 0.933	**n/a**