

Exploiting synthetic images for real-world image recognition

Max Maton

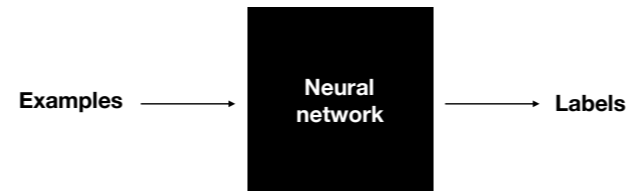


All used images with the exception of MNIST samples are CC0

Hi, I'm Max. Thank you all for being here today.

Last two months I've been doing research on using rendered images for use in machine learning.

What is machine learning?



What is machine learning?

Basically machine learning is the field of creating black boxes called neural networks. You feed your black box examples together with the desired label and you let it train on that data. After that training, if you're lucky, the black box learns to give the right labels to examples it hasn't seen before.

Let's walk through the creation of a simple neural network called a classifier:

Dataset creation



Daisy



Dandelion

It all starts with collecting data. Let's say we want to create a classifier for flowers. We collect photographs of daisies and dandelions and train the classifier with them. We then take an image we haven't shown the classifier yet and ask it what it thinks the label is.

Validation



Daisy (70%)

Unfortunately we haven't given our classifier enough data so it misclassified our validation photo. In this case our training samples for daisies all had a green background and our dandelions had other background colours. This caused it to predict the wrong label.

The solution to this problem is to collect more photographs in a lot of different situations.

Unfortunately, photographers are expensive so people are starting to use rendered data to augment their real datasets.

Rendered data



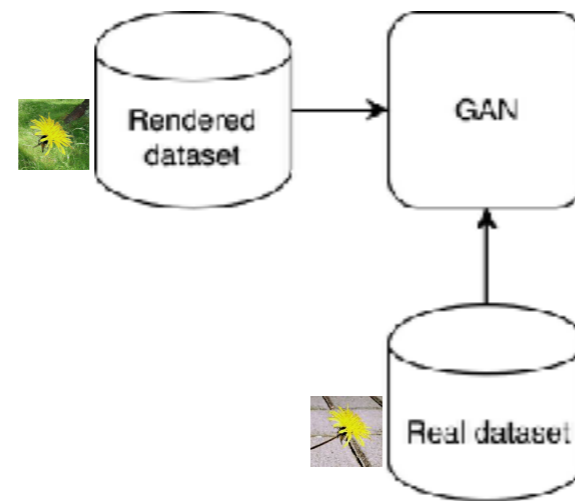
An example of how we could use rendered data for this dataset is by carefully removing the flower itself in one of the images and pasting it on top of a different photo without a flower. This creates a new example of a picture with a dandelion in it.

Distribution gap



When looking at the resulting image in more detail it becomes apparent that we introduced some artefacts. Most rendered images have small unnatural differences from real images. In this case it's the bad shadows, the cutout artefacts and the grey border. These differences are known as a distribution gap between the real dataset and the rendered dataset. Training of neural networks on rendered datasets therefore often doesn't work as well as expected.

Are GANs able to fix this?



7

The research focussed on figuring out if it's possible to inflate small real datasets by training a neural network to convert rendered images to images that look like they are real. This network is called a GAN.

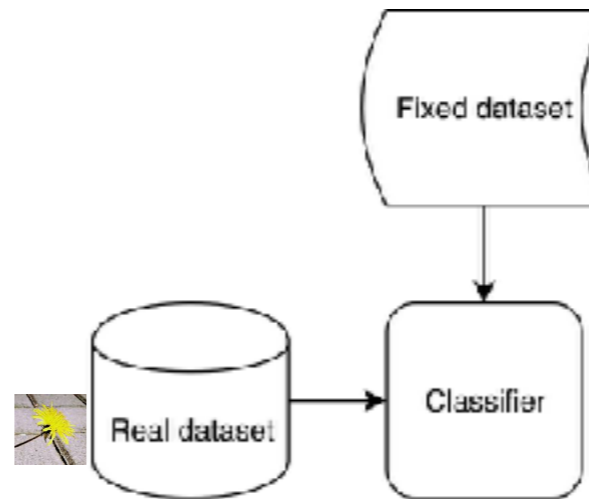
First we train the GAN using our real and rendered dataset.

Are GANs able to fix this?



We then convert the rendered dataset to a fixed dataset that looks real.

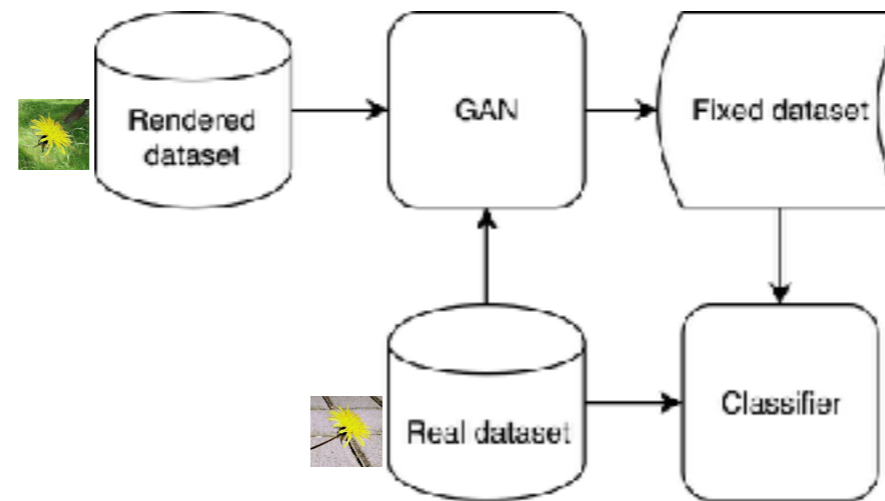
Are GANs able to fix this?



9

Finally we train the classifier we want on the combined images from the real and fixed dataset.

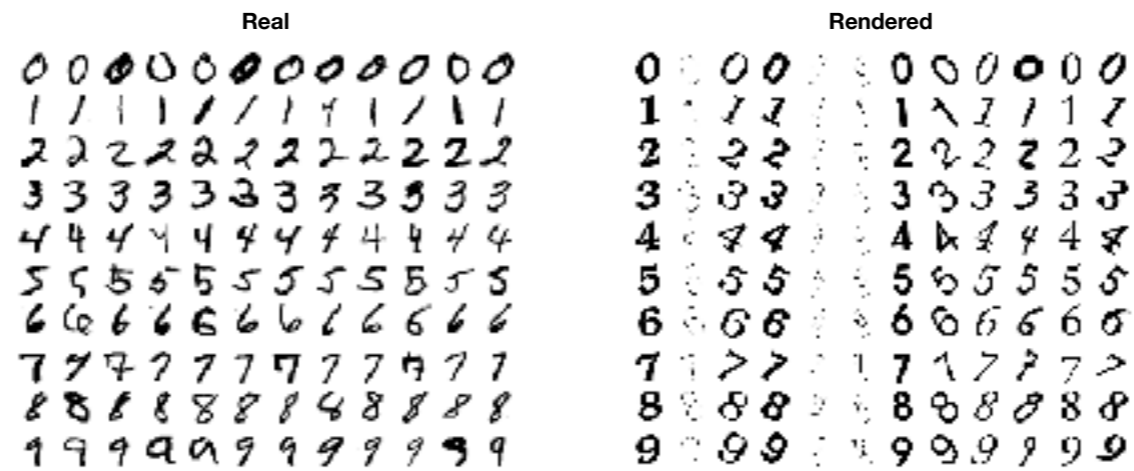
Are GANs able to fix this?



10

Our goal is to get a better accuracy on the new classifier than we had before.

Using MNIST as testcase



11

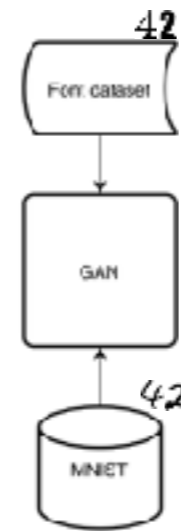
During the research we used something a little bit less computationally intensive. We tried to classify handwritten digits from the MNIST dataset which you can see on the left. We used rendered fonts as the rendered dataset as you can see on the right.

Using MNIST as testcase



Here you can see how we created the font dataset. We used open source fonts to create a dataset of images that look like the numbers on the top right.

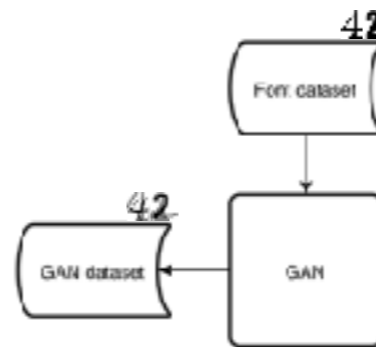
Using MNIST as testcase



13

We trained a GAN on the real MNIST images and on our rendered images.

Using MNIST as testcase



The trained GAN is then used to create a new dataset that looks like the real images.

Using MNIST as testcase



15

Again we then use the GAN dataset and the real dataset to train a classifier.

Using MNIST as testcase

```

graph TD
    A[(Dataset)] --> B[Normalization]
    B --> C[(Form. dataset)]
    C --> D[GAN]
    D --> E[(GAN dataset)]
    E --> F[Classifier]
    G[(MNIST)] --> F
    D --> G
  
```

The flowchart illustrates the process of using the MNIST dataset as a test case for a GAN-based classifier. The process begins with a 'Dataset' (cylinder) which is processed by 'Normalization' (rectangle) to form a 'Form. dataset' (cylinder). This dataset is then used to train a 'GAN' (rectangle). The GAN generates a 'GAN dataset' (cylinder), which is then used by a 'Classifier' (rectangle). The 'Classifier' also receives input from the 'MNIST' dataset (cylinder). The 'GAN' is also connected to the 'MNIST' dataset, suggesting a comparison or evaluation process. Handwritten '42' annotations are present next to the 'Form. dataset', 'GAN dataset', and 'MNIST' cylinders.

We ran this experiment with smaller and smaller amounts of real MNIST data to see what would happen.

Output

55 images

in	0	1	2	3	4	5	6	7	8	9
out	2	8	2	8	9	9	5	9	7	5

17

Here you can see some of the images that were generated by the GAN. In this case for 55 images from MNIST which means that the GAN had on average 5,5 examples per digit.











As you can see it doesn't really work.

Output 385 images

in	0	1	2	3	4	5	6	7	8	9
out	7	2	2	0	7	2	2	2	3	7

When we go to 385 images the output still doesn't seem that much better.

Output 5,500 images

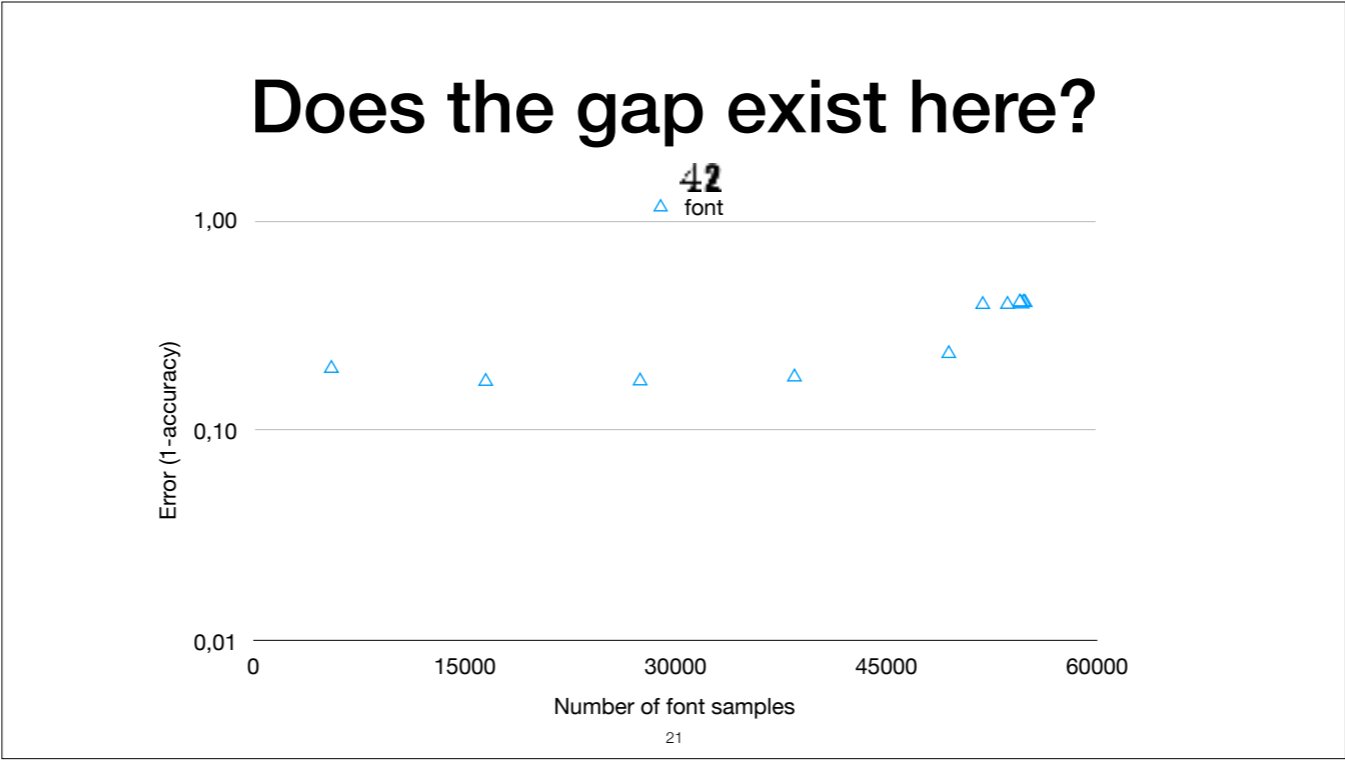
in	0	1	2	3	4	5	6	7	8	9
out										

At 5,500 images however we can see pretty accurate results. The generated digits start to look like they are real handwritten digits.

Output 38 , 500 images

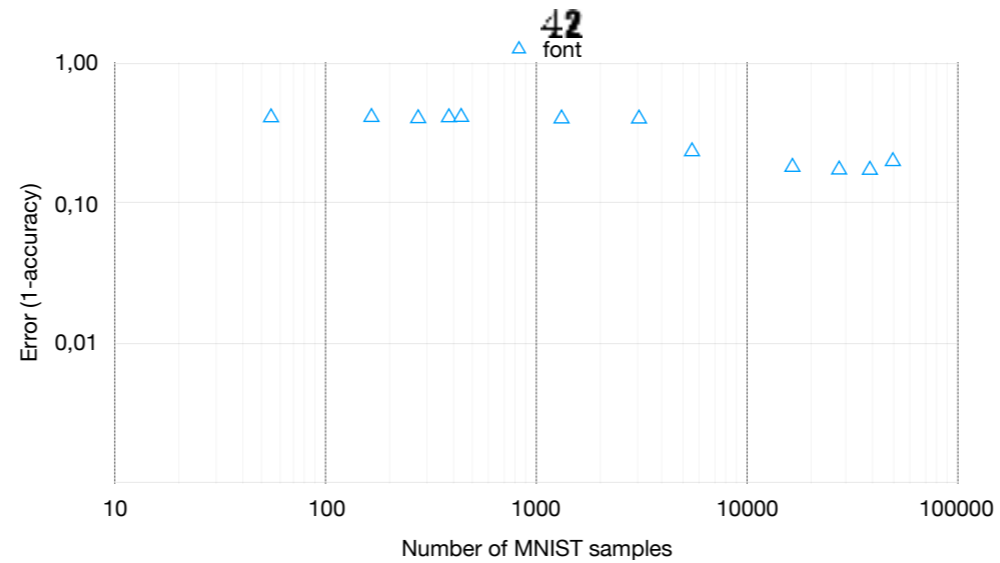
in	0	1	2	3	4	5	6	7	8	9
out	0	1	2	3	4	5	6	7	8	9

When we use 38,500 images they look like they could be real.



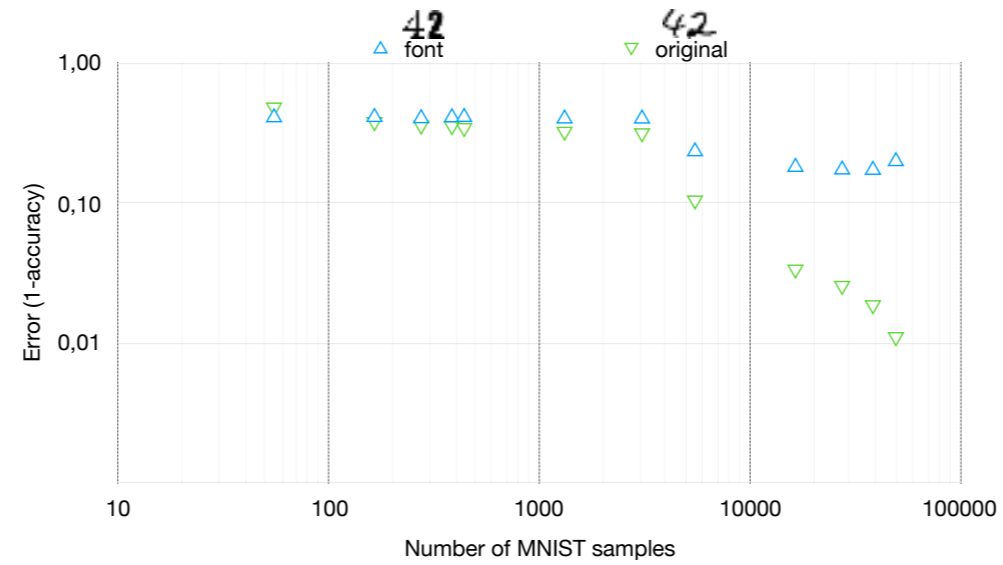
We mentioned that there is usually a distribution gap between rendered and real datasets. To validate wether we didn't accidentally create a rendered dataset that was too accurate we tested the error of a classifier trained on the font dataset on MNIST. In this case we want the error to be as low as possible. You can see the error going up when we train with more font images. This indicates that there is a distribution gap between the rendered images and the real MNIST images.

Does the GAN help?



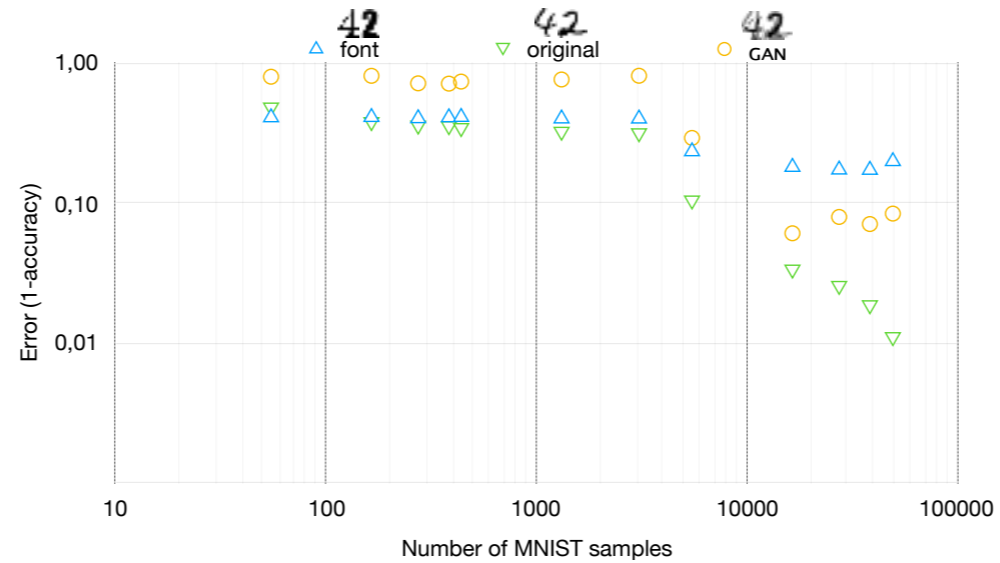
We wanted to know whether the GAN was actually helping so we plotted the result of classifiers trained with varying amounts of data together. You can see the previous graph again, but flipped horizontally. This is because we use less font images if we use more MNIST images.

Does the GAN help?



Now we add the results from the classifiers that are only trained on the real data. This is our baseline.

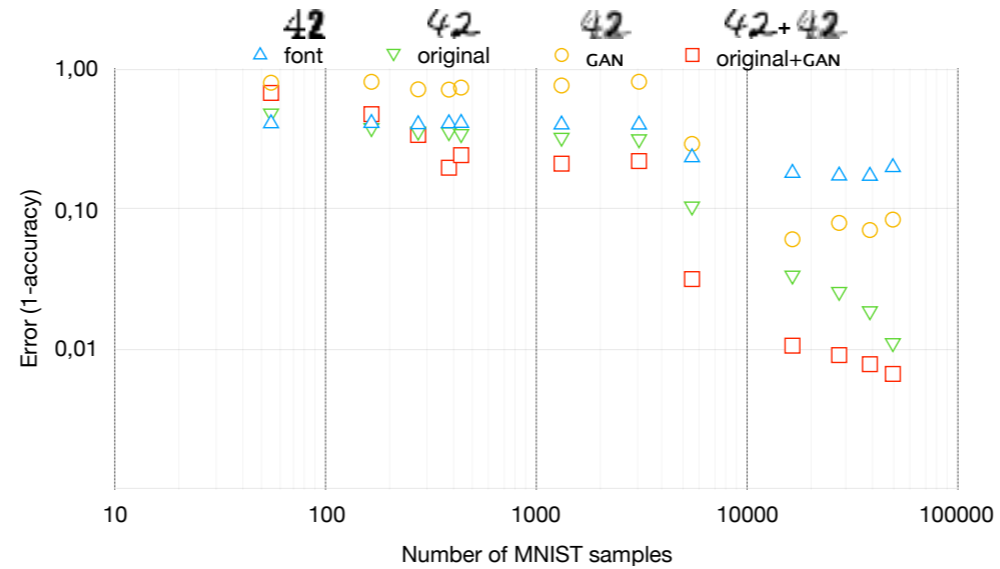
Does the GAN help?



24

We then plot the error of the classifiers trained on only the GAN data. From these results it looks like the GAN data is always worse than the baseline and sometimes even worse than the font dataset.

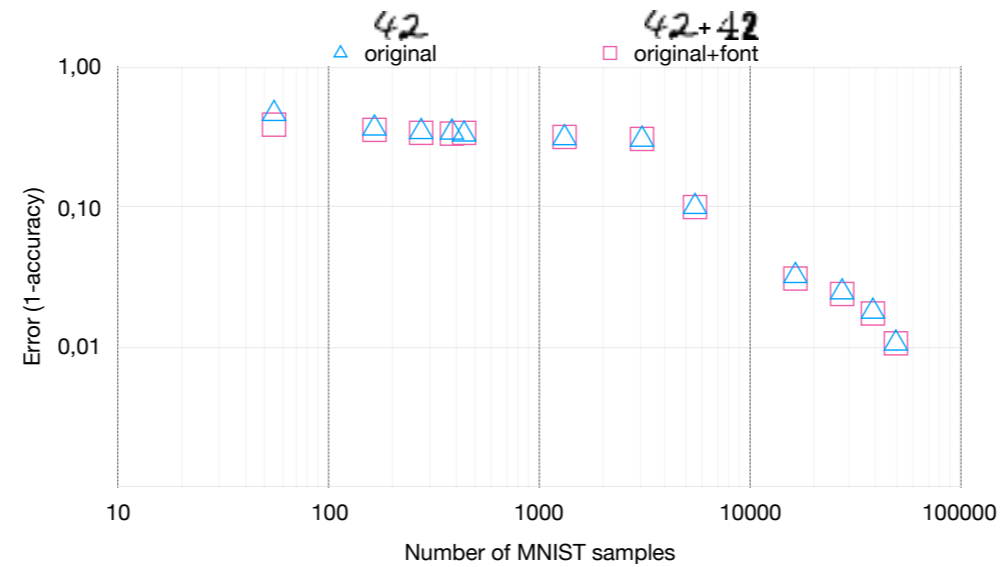
Does the GAN help?



When looking at the result of classifiers trained on original+GAN however we see that that classifier starts performing better than our baseline. This seems to start happening from 385 images.

All in all the GAN certainly seems to help.

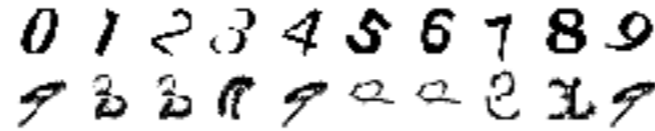
Isn't original+font enough?



To make sure we couldn't achieve the same result by training on original+font we plotted that as well. We noticed that it decreased the error slightly but not nearly as much as training together with the GAN data.

Further research

- GAN loses labels while still working



When looking at the classifier results graph earlier we noticed that original+GAN started to be better at 385 images. If we look at the output of the GAN at that point the digits still look very unnatural. Due to time constraints we haven't been able to figure out why it still works.

Conclusion

- This works great on MNIST
- Even with very little real training data
- GANs are very useful for inflating datasets

28

In short, This works great on MNIST
even with very little training data.
And GANs seem very useful for inflating small datasets.

<https://aiir.maxmaton.nl>

Thank you

Questions?

Exploiting synthetic images for real-world image recognition

Max Maton



All used images with the exception of MNIST samples are CC0

How much better is it?

