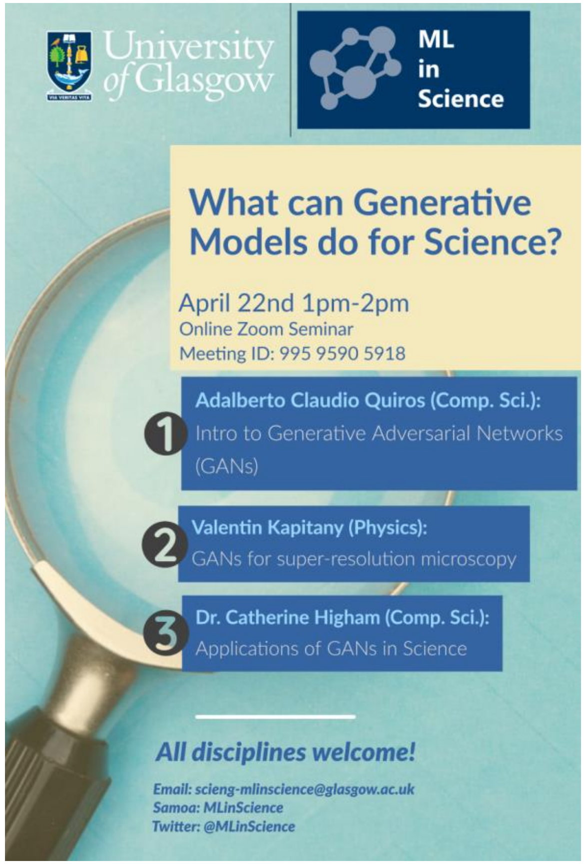


Applications of GANs in science

Catherine Higham

Wednesday 22nd April

Talk Outline

- Long Short Term Memory (LSTM)
  - Generative model used with sequence data
    - Alice in Wonderland example
    - Complexity and Learnability of language
- Conditional GANs
  - Generating images conditioned on a 'label'
    - Comparison to GAN MNIST example
    - Hybrid LiDAR – RGB and single pixel LiDAR

Long Short Term Memory LSTM

- Sequence data
- Embedding
- Input/output remember/forget
- Distribution over classes
- Applications Natural Language

Considerations

Training data

- Preparing and storing for efficiency
- labels/ground truth
- simulations

Architecture

- Type of layers: Size/number of filters/weights
- Connections between layers/networks

Training options

- Learning rates, dropout, gradient descent, regularizers

Prediction

- Classification,segmentation,end user
- Uncertainity quantification

Training Data: Natural Language

HTML code from Alice's Adventures in Wonderland by Lewis Carroll from Project Gutenberg.

Parse HTML code into short paragraphs using Text Analysis Tools

- *"In another moment down went Alice after it, never once considering how in the world she was to get out again. "*

Create a datastore that contains the data for training.

- For the predictors, this datastore converts the documents into sequences of word indices using a word encoding.
- For the responses, the datastore returns categorical sequences of the words shifted by one.

Architecture

Layers	Learnable Weights	Activations
Sequence Input		1
Word Embedding	100 x 2920	100
LSTM	4 x 100 x 100 input 4 x 100 x 100 recurrent + 4 x 100 bias	100
Dropout		100
Fully Connected	2920 x 100 + 2920 bias	2920
SoftMax		2920
Classification		

- Training Options:
  - 'adam',
  - 'MaxEpochs',300,
  - 'InitialLearnRate',0.01,
  - 'MiniBatchSize',32,
  - 'Shuffle','never',
  - 'Plots','training-progress',
  - 'Verbose',false.

Prediction

generatedText = "";

Predict the "next word" score

Network
Sequence Input (wordIndex)
Word Embedding
LSTM
Dropout
Fully Connected
SoftMax
Classification

Sample the "next word"

generatedText = generatedText + "next word"

Prediction

generatedText = "";

Predict the "next word" score

Network
Sequence Input (wordIndex)
Word Embedding
LSTM
Dropout
Fully Connected
SoftMax
Classification

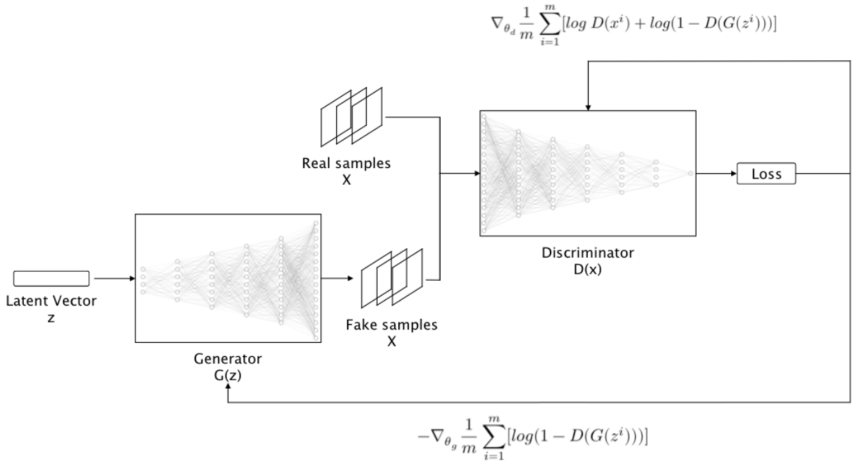
Sample the "next word"

generatedText = generatedText + "next word"

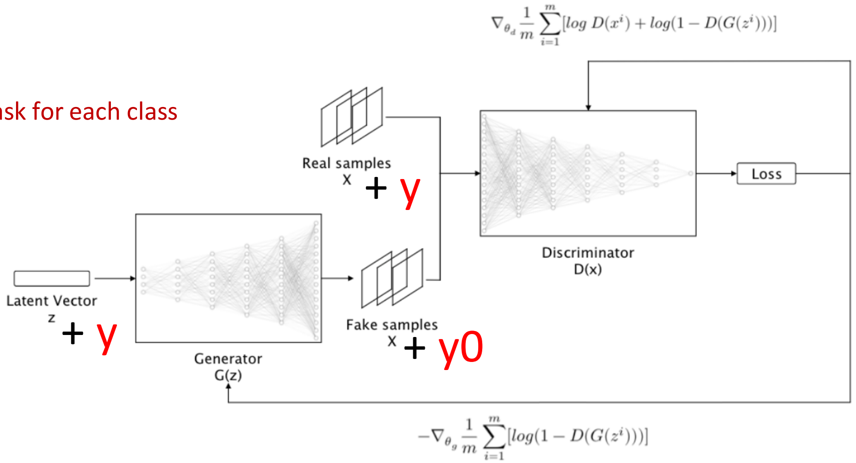
generatedText = " ‘Sure, it’s a good Turtle!’ said the Queen in a low, weak voice."

Conditional GAN

- Generator and Discriminator
- Embedding the label
- Pitfalls, mode collapse, drift
- Applications Images, Data Fusion



Embed z,x with learned mask for each class



GAN CODE x|z

```
fake_images = Generator(z);
d_output_real = Discriminator(x);
d_output_fake = Discriminator(fake_images);
% Loss due to true or not
d_loss = -mean(log(d_output_real)+log(1-d_output_fake));
g_loss = -mean(log(d_output_fake));
% For each network, calculate the gradients with respect to the loss
GradGen = dlgradient(g_loss);
GradDis = dlgradient(d_loss);
```

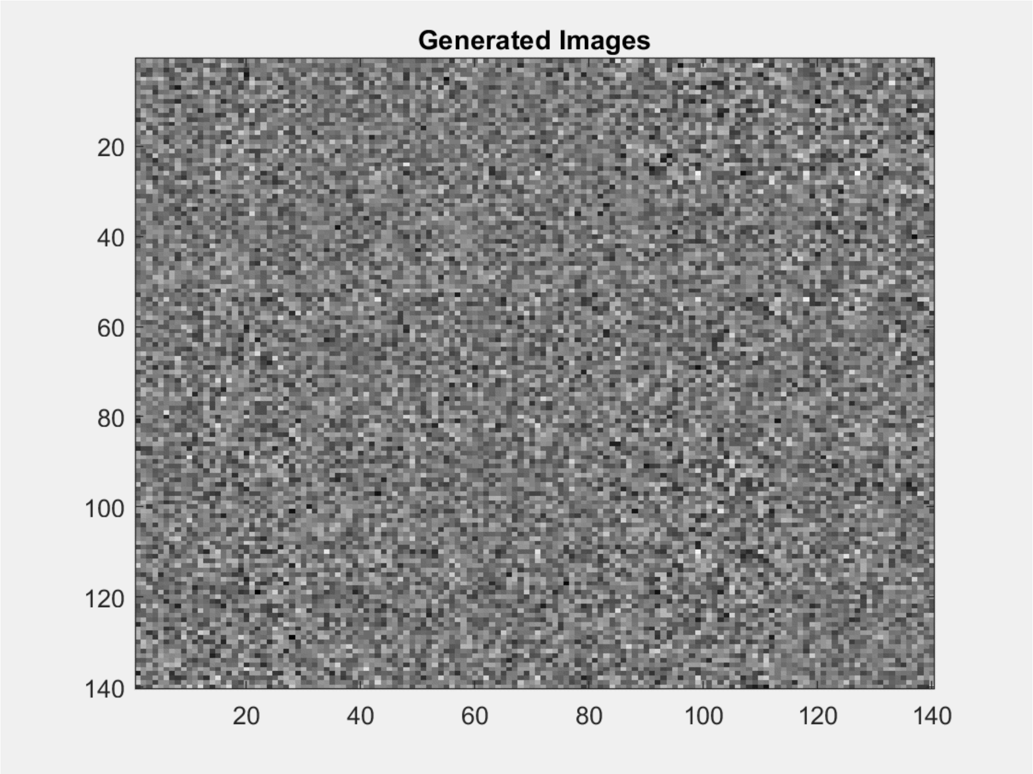
https://github.com/zcemycl/Matlab-GAN

Conditional GAN CODE x|z,y

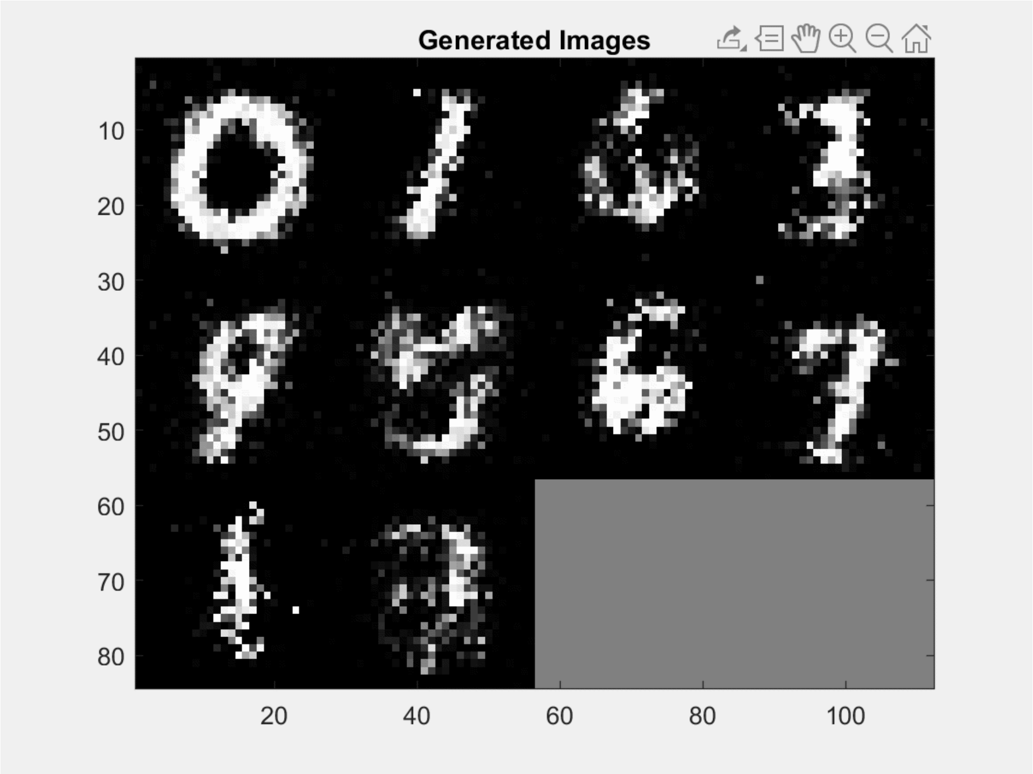
```
fake_images = Generator(z,y);
d_output_real = Discriminator(x,y);
d_output_fake = Discriminator(fake_images,y);
fake_images0 = Generator(z,y0);
y0 = randomly sampled from label classes;
d_out_fake0 = Discriminator(fake_images0,y0);
% Loss due to true or not
d_loss = -mean(log(d_output_real)+log(1-d_output_fake));
g_loss = -mean(log(d_out_fake0));
% For each network, calculate the gradients with respect to the loss
GradGen = dlgradient(g_loss);
GradDis = dlgradient(d_loss);
```

MNIST digit generation

GAN

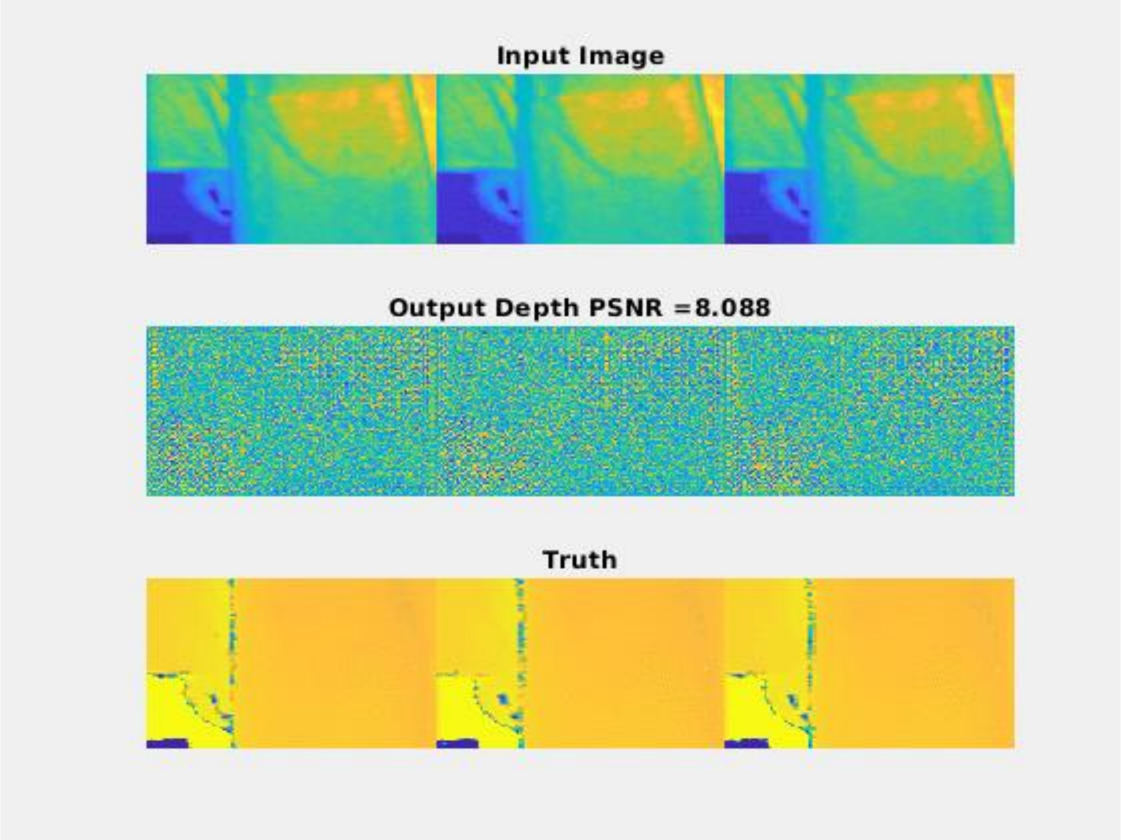


CGAN



https://github.com/zcemycl/Matlab-GAN

Hybrid LiDAR generate Depth|G (RGB)+single LiDAR signal



Joint work with Steven Johnson, Neal Radwell, Rod Murray-Smith and Miles Padgett