Joint Caption Detection and Inpainting using Generative Network

July 21, 2018

1 Team details

- Team name - Inception
- Team leader name - Anubha Pandey
- Team leader address, phone number and email-

  Address - Room No.651, Tunga Hostel, IIT Madras, Chennai, India, Pin Code 600036
  Phone Number - 7024024065
  Email - anubhap93@gmail.com
- Rest of the team members - Vismay Patel
- Team website URL (if any) `https://github.com/vismay93/VideoDecaptioningChalearn`
- Affiliation - Computer Vision Lab, Indian Institute of Technology, Madras

2 Contribution details

- Title of the contribution :

  Joint Caption Detection and Inpainting using Generative Network
- Final score :

  PSNR: 32.0020521872
  MSE: 0.00120915453247
  DSSIM: 0.0499458078688

- General method description :

Input

Inputs to the network are the videos from the training dataset and the ground truth masks extracted for each frame of the input videos around the captions. The masks are extracted by taking the difference between the corresponding frames of the ground truth videos and input videos to be inpainted and it contain ones in places of captions and zeros everywhere else.

Frame level Inpainting generator with skip-connections

Our network has the following main modules:-
1. Generator Module: This module is used to generate inpainted images and masks corresponding to the frames of the input videos. It is an encoder-decoder based CNN model. In the encoder part initially we use series of convolution layers. Each convolution layer is followed by a batch normalization and a RELU layer. We use dilated convolutions in the later stages of the encoder. Dilated convolutions are also followed by a batch normalization and a RELU layer. First three convolution layers are shared between both the image and mask generation tasks and later layers are divided into two branches and are trained independently. The decoder for both the branches use a series of deconvolution followed by convolution layers. We also add skip connections from the encoder module to the decoder module in the image generation branch. The decoder tries to generate an image closer to the ground truth frame and a mask of the size same as the input frame, containing holes(consists of ones) around the captions and zeros everywhere else. The input to the Generator module is 128*128*3 sized tensor which corresponds to the frame of input video to be inpainted. The generator architecture is shown in Figure 2.
Our network is inspired by the work of [1].
2. Discriminator Module: This module is used while applying GAN [2] loss to the generator. It is responsible to distinguish between Real ground truth frames and the fake inpainted images generated corresponding to the input frames from the generator. Our discriminator module consists of 5 convolution layers and a fully connected layer as can be seen from the Figure 2b. The discriminator receives 2 types of inputs- The real ground truth frames that comes from the ground truth videos of the training set and the fake images are generated by the generator module. The discriminator outputs probability of the image being real. Thus the output value should be close to 0 for fake images and it should be close to 1 for real ground truth images.

Loss Functions

Following loss functions have been used to train the network:-

1.Reconstruction Loss - To train the generator in order to generate images and masks similar to the ground truth frames and masks respectively, we try to reduce absolute error between them.

$L_r = \frac{1}{K}\sum_{i=1}^{K}|I_x^i - I_{imitation}^i| + \alpha * \frac{1}{K}\sum_{i=1}^{K}(I_{Mask}^i - O_{Mask}^i)^2$
where, K is the batch size and alpha = 0.000001(hyper parameter)

2.Adversarial Loss - To the train the GAN [**?**] network end to end, adversarial loss is used. It is the combination of reconstruction loss and discriminator loss. Discriminator Loss - In order to maximize the probabilities for real images and minimize the probabilities for fake images, the total discriminator loss is combination of 2 partial losses.
$L_{real} = -log(p)$
$L_{fake} = -log(1-p)$
$L_d = l_{real} + 6 * L_{fake}$
$L_{adv} = L_r + \beta * \frac{1}{K}\sum_{i=1}^{K} L_{fake}$
$where, \beta = 0.01(hyperparameter)$

3.Perceptual Loss [3] - We have used pretrained vgg network as loss function between ground truth images and the generated images.

$L_p = \frac{1}{K}\sum_{i=1}^{K}(\phi(I_y) - \phi(I_{imitation}))^2$
where, ϕ represents VGG16 network pretrained on Microsoft COCO dataset.

Training

We have train our network using Adam Optimizer with learning rate 0.006 and batch size 20. For first 8 epochs we have trained only the generator module of the network by minimizing reconstruction loss and perceptual loss and for the next 12 epochs we have trained the entire GAN network end to end on the adversarial and perceptual loss.

• References

[1] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (TOG)*, 36(4):107, 2017.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[3] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

- Representative image / diagram of the method

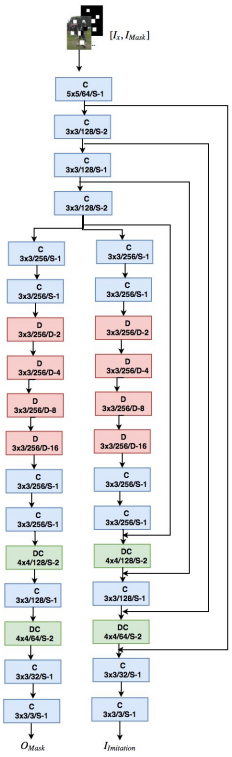
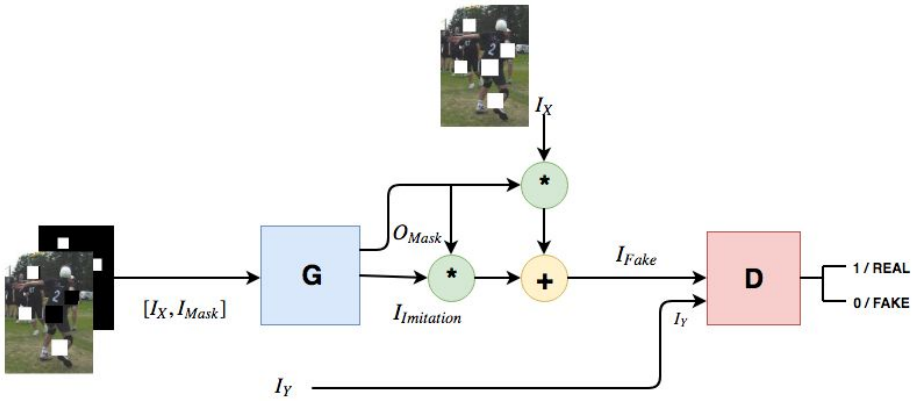
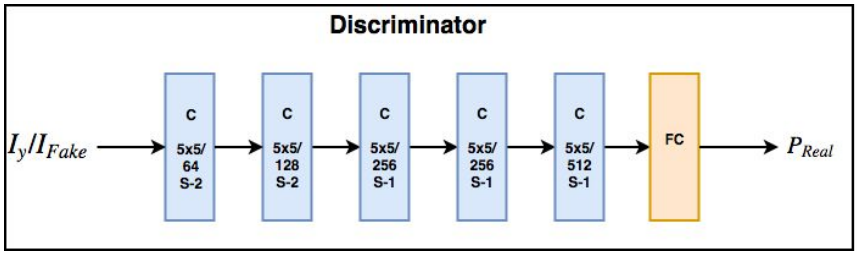


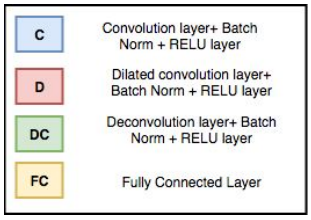
Figure 1: Architecture of the generator module of the inpainting network. Each building block is described in Figure 2c.



(a) Full pipeline of the Image Inpainting network.



(b) Architecture of the discriminator module of the inpainting network. Each building block is described in Figure 2c.



(c) Building blocks of the network. Each block in the diagram also contains the filter size/ number of output channels and stride(S) or dilation(D) of the respective layer.

Figure 2: Network Architecture.

- Describe data preprocessing techniques applied (if any):

  We extract masks corresponding to each frames of the input video and feed it to the network as ground truth mask along with the input videos from the training dataset. The masks contain ones in places of captions and zeros everywhere else.

3 Method description

3.1 Features / Data representation

Describe features used or data representation model (if any)

-None-

3.1.1 Dimensionality reduction

Dimensionality reduction technique applied (if any)

-None-

3.1.2 Compositional model

Compositional model used, i.e. pictorial structure (if any)

-None-

3.1.3 Learning strategy

Learning strategy applied (if any)

We use an encoder-decoder based CNN model to train the network to generate both inpainted frames and masks corresponding to the input videos. The network has two branches each for the image generation and the mask generation tasks, and these branches share the parameters upto three convolution layers thereafter, are trained independently. Each branch uses a combination of regular and dilated convolutions followed by batch normalization and ReLU to encode the partial image, the decoder uses combination of deconvolution and convolutions to generate the full image. The decoder of the image generation branch uses skip connections from the encoder. We use a combination of Reconstruction loss, perception loss and Adversarial loss to train our model using Adam Optimization Algorithm.

Table 1: Generator architecture. conv:2d convolution layer, bn: batch normalization layer, dilconv: dilated convolution layer, deconv: deconvolution layer, relu: RELU activation layer, tanh: Tanh activation layer, sigm: Sigmoid activation layer, ic13 and mc13 are output to the network.

Layer name	layer type	input	filter size	filter channels	stride/dilation
c1	conv+bn+relu	frame	5x5	64	1
c2	conv+bn+relu	c1	3x3	128	2
c3	conv+bn+relu	c2	3x3	128	1
c4	conv+bn+relu	c3	3x3	128	2
ic1	conv+bn+relu	c4	3x3	256	1
ic2	conv+bn+relu	ic1	3x3	256	1
ic3	dilconv+bn+relu	ic2	3x3	256	2
ic4	dilconv+bn+relu	ic3	3x3	256	4
ic5	dilconv+bn+relu	ic4	3x3	256	8
ic6	dilconv+bn+relu	ic5	3x3	256	16
ic7	conv+bn+relu	ic6	3x3	256	1
ic8	conv+bn+relu	ic7	3x3	256	1
ic9	deconv+bn+relu	ic8	4x4	128	2
ic10	conv+bn+relu	ic9	3x3	128	1
ic11	deconv+bn+relu	ic10	4x4	64	2
ic12	conv+bn+relu	ic11	3x3	32	1
ic13	conv+tanh	ic12	3x3	3	1
mc1	conv+bn+relu	c4	3x3	256	1
mc2	conv+bn+relu	mc1	3x3	256	1
mc3	dilconv+bn+relu	mc2	3x3	256	2
mc4	dilconv+bn+relu	mc3	3x3	256	4
mc5	dilconv+bn+relu	mc4	3x3	256	8
mc6	dilconv+bn+relu	mc5	3x3	256	16
mc7	conv+bn+relu	mc6	3x3	256	1
mc8	conv+bn+relu	mc7	3x3	256	1
mc9	deconv+bn+relu	mc8	4x4	128	2
mc10	conv+bn+relu	mc9	3x3	128	1
mc11	deconv+bn+relu	mc10	4x4	64	2
mc12	conv+bn+relu	mc11	3x3	32	1
mc13	conv+sigm	mc12	3x3	3	1

3.1.4 Other techniques

Other technique/strategy used not included in previous items (if any)

-None-

3.1.5 Method complexity

There are 31686529 parameters to be learned in the network.

3.2 Data Fusion Strategies

List data fusion strategies (how different feature descriptions are combined) for learning the model / network: Single frame, early, slow, late. (if any)

Single frame is used for learning the model/network.

3.3 Global Method Description

- Which pre-trained or external methods have been used (for any stage, if any)

  -None-

- Which additional data has been used in addition to the provided ChaLearn training and validation data (at any stage, if any)

  -None-

- Qualitative advantages of the proposed solution

  The perceptual loss fill the missing patches with the priors from the images of the same class, while the GAN loss allows us to generate realistic looking images. The skip-connections allows us to pass fine details to the coarse layers in order to generate details in the images. The caption detection allows us to copy

- Results of the comparison to other approaches (if any)

- Novelty degree of the solution and if is has been previously published

  We have tried frame level decaptioning of the videos. We train the generator to generate masks around the captions in the frame and inpaint the frame simultaneously. Most of our work is inspired from previous work on image generation and image inpainting. We have used several techniques proposed in the previous literature and combined them together in our solution.

4 Other details

- Language and implementation details (including platform, memory, parallelization requirements)

  tensorflow-gpu 1.6.0 on top of python 3.6.4 is used for implementation
  Platform: Ubuntu 14.04
  GPU : GeForce GTX 1080
  CUDA : 9.0

- Human effort required for implementation, training and validation?

  15 person days of coding (implementation)

- Training/testing expended time?

  Training requires 2 days to train the network upto 20 epochs for 75000 videos.
  Testing requires 2:30 hours for 5000 videos.

- General comments and impressions of the challenge? what do you expect from a new challenge in face and looking at people analysis?

  The videos were low resolution and taken from different domains, making it difficult to do better inpainting. We expect more active participation in the new challenge in face and looking at people analysis.