Image Generation from Captions Using Dual-Loss Generative Adversarial Networks

Vashisht Madhavan
vashisht.madhavan@berkeley.edu

Philip Cerles
philip.cerles@berkeley.edu

Nishant Desai
nishantdesai@berkeley.edu

Abstract

Deep Convolutional Generative Adversarial Networks (DCGANs) have become popular in recent months for their ability to effectively capture image distributions and generate realistic images. Recent work has also shown that conditional information provided to Generative Adversarial Networks (GANs) allows for deterministic control of generated images simply through the regulation of the conditional vector. Although many have hinted that language models could be used as conditional information for generating images from words, there are very few attempts using GANs, much less DCGANs. In this project we explore and analyze the results of image generation by encoding captions as conditional information to our DCCGAN. We use a subset of the MSCOCO dataset to evaluate our results.

1. Introduction

In recent years, deep networks have proven to be highly effective as discriminative models for computer vision problems (e.g. classification). However, relatively little work has been done in using these networks to develop generative models of image datasets. The GAN framework attempts to bridge this gap. A generative adversarial network consists of two parts: the generator attempts to approximate the distribution of a given dataset in the feature space of the discriminator, which attempts to tell generated images apart from dataset images [13]. The network derives its name from the fact that these two components have opposing objectives. The generator attempts to maximize the discriminator's confusion, while the discriminator tries to minimize it. The discriminator's confusion is measured by the binary entropy of its prediction over a set of real and fake images.

To produce images, a vector of random noise is used as input to the generator network; this allows it to generate images that are varied in content. Recent work has attempted to control features of the generator's output by supplying it with external information alongside the noise at training time [10]. This extra input forces the model to learn distributions of the data conditioned on the external information. GANs that make use of this conditioning information are known as conditional generative adversarial networks (cGANs).

In this work, we attempt to use the cGAN framework to generate images from captions. We supply the cGAN with vector embeddings of captions from the MSCOCO dataset as conditioning information. In addition to evaluating generated images against the discriminator, we also pass each generated image through a pretrained captioning model to get a caption recovery loss, which is used to supplement training of the generator.

2. Related Work

In 2014, Goodfellow et al. [4] introduced the concept of Generative Adversarial Networks as a method of effectively modeling the distribution of image data, while leveraging the powerful feature representations of deep networks. The process involved using two neural network setups, a generator (G) and a discriminator (D), in a jointly-trained architecture. The problem is framed in a two-player minimax, or adversarial, setting, where G tries to maximize the probability of D making a mistake and D tries to minimize the domain classification error between real and generated images. Although the framework seemed promising at first, the initial results were not, as the images did not look very much like the training data and lacked fine-grained detail.

Shortly after Goodfellow et al., Mizra et al. [10] introduced the idea of conditional GANs, by passing class labels and other multi-modal information as a conditioning vector to the input of the G and D networks. This in turn showed better generation of images for MNIST and also better tagging for Flickr datasets. Although it was quite preliminary in analysis, it provided a roadmap for using conditional in-

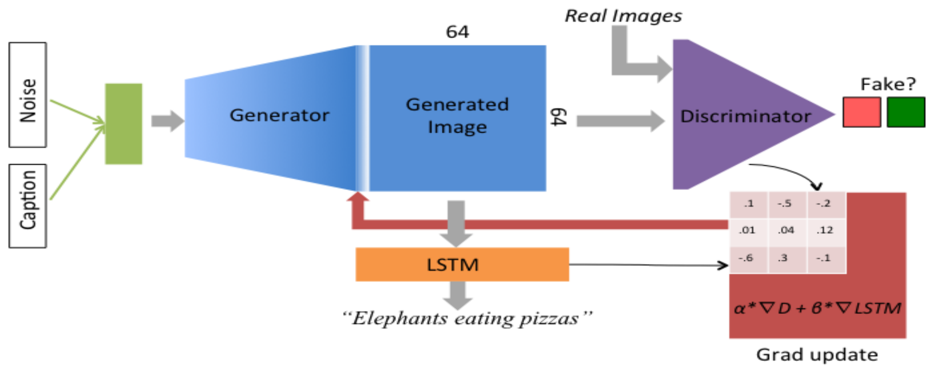


Figure 1. Architecture of cDCGAN with Caption Loss

formation as part of GANs.

In his 2015 paper, Gauthier [3] introduced a method of using conditional information in GANs to deterministically control generated images. Unlike the architecture Mizra et al. presented, a conditional vector embedding is passed before the fully connected layers of the D network. Using face attributes as the conditioning information and face images as the inputs, the architecture gives surprising results under changes to the condition embedding. With slightly changed values of the conditional vector, the network generates images that highlight different facial attributes. In our project, we attempt to see whether using sentence embeddings of image descriptions can effectively produce generated images which capture the meaning of the description.

Due to the poor results of vanilla GANs, fully convolutional network representations of GANs have become much more popular. Convolution Neural Networks (CNNs), much like the one from Krizhevsky et al. [7], have shown superior results on a number of vision tasks and have gained popularity as image feature extractors. Deeper convolutional GAN methods, such as those presented by Radford et al. [12] and Denton et al. [2]. have shown better and more stable results for generating realistic images. Since Radford's Deep Convolutional Generative Adversarial Network (DCGAN) is the more recent method, we chose to apply conditional information to this architecture and analyze the results.

3. Approach

3.1. Network Architecture

We modified a DCGAN architecture to incorporate caption loss in order to train for realistic image creation that semantically resembled the input caption. Our network is composed of two components: a generative component which optimizes likeness to the original dataset, and a semantic component which optimizes resemblance to the input caption. Other conditional GAN architectures [2, 3] provide conditional information to both the generator and the discriminator. Though this is a cleaner approach theoretically, we found that this approach did not produce recognizable images. Another issue with this approach was that we were unable to tune the loss between the semantic error (LSTM) and generative error (Discriminator). While our approach uses fixed parameters for this trade-off, future experiments might attempt to emphasize discriminator error in earlier training epochs, adding caption loss only when generated images begin to resemble those of the dataset.

The generative component of the cDCGAN resembles that of the original DCGAN paper [12], and uses the same number and size of convolutional filters. However, the input to the generative network is a concatenated 1-dimensional vector of noise and the caption embedding, while the discriminator network is unchanged from the original architecture. The semantic component of the cDCGAN is composed of a pre-trained LSTM network, which takes in a generated image, the desired caption, and returns loss based on the probability of generating this caption. Gradients from the two sources of loss are added together and used to backpropagate the generative network.

3.2. Loss Functions and Optimization

The discriminator in our model uses the binary cross entropy of its predictions as its loss function [4]. For a given batch of real and fake images, this function is given by

$$H(\mathbf{I}, \mathbf{z}, \mathbf{s}) = -\frac{1}{n}\left(\sum_{i=1}^{n} \log(D(I_i)) + \log(1 - D(G(z_i, s_i)))\right)$$

Under this loss, the discriminator tries to maximize the predicted probability of real images and minimize the predicted probability of fake images.

Our generator loss function consists of two components. The first is its loss with respect to the discriminator's output. The generator attempts to maximize the discriminator's binary cross entropy; this is equivalent to maximizing the probability that the discriminator assigns to the generated output, giving us the following criterion for a batch of generated images:

$$L_D(\mathbf{z}, \mathbf{s}) = -\frac{1}{n}\sum_{i=1}^{n} \log D(G(z_i, s_i))$$

This is the standard generator loss used for GANs [4].

In addition to the discriminator loss, we also introduce a caption recovery loss based on the output of the LSTM captioning model. The LSTM outputs a distribution over possible words at every position in the sentence. The loss associated with this output is the sum of negative log probabilities assigned to the correct word at each position [13].

$$L_{LSTM}(I, s) = -\sum_{t=1}^{N} \log p_t(s_t)$$

In the above equation, N represents the total sentence length.

We use a weighted sum of these components to get a final loss for the generator. For a batch of input, the total loss for the generator is given by

$$L_G(\mathbf{z}, \mathbf{s}) = \alpha L_D(\mathbf{z}, \mathbf{s}) + \beta \sum_{i=1}^{n} L_{LSTM}(G(z_i, s_i), s_i).$$

Our use of conditioning information, $\mathbf{s}$, here differs from the way it is incorporated into training in the standard cGAN formulation [3, 10]. Conditioning information is typically introduced directly into the last layer of the discriminator, which can learn to decide whether or not the input image matches its conditioning information. However, in our experiments, we instead pass the conditioning information to the captioning model, which is specifically trained to detect whether an image and a caption match one another.

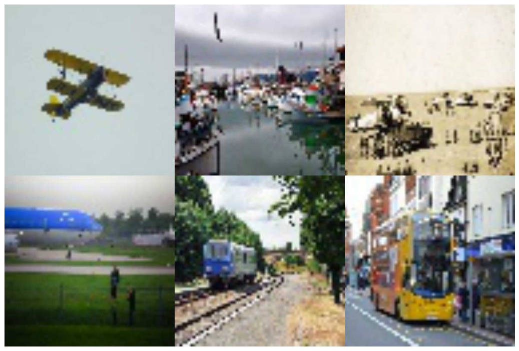


Figure 2. Example images from the training set.

4. Experiment Setup

4.1. Implementation Details

All of our models were implemented in the Torch deep learning library [1]. We used a previous DCGAN Implementation found at https://github.com/soumith/dcgan.torch. To add to this model, we introduced a conditioning vector of chosen size to pass in with the input to the G network, as Gauthier [3] suggests. We found that this method resulted in more stable training than the technique reflected in similar implementations by Radford et al. at https://github.com/Newmu/dcgan_code, where the conditioning information is passed in at every layer of the G network.

The captioning network has an architecture similar to the one proposed by Vinyals et al. [13], where a GoogLeNet is used to extract image feature representations, which are then passed into an LSTM to generate captions. We used a Torch implementation found at https://github.com/karpathy/neuraltalk2. In this implementation, a VGG-16 net is substituted in place of the original GoogLeNet. Since our network architecture did not lend itself to jointly training an LSTM and DCGAN, we used a network pre-trained on MSCOCO to our advantage. Since the neuraltalk2 repository provided us with model weights in Torch, we found it was the best option for our implementation.

4.2. Data

For these experiments we chose a subset of images from the MSCOCO 2014 training and validation sets. To make the distribution of images less noisy and easier to learn, we chose all images that correspond to the super-category "vehicles". Since Radford et al. [12] suggest that DCGANs for image generation at resolutions higher than 64x64 is inherently unstable with the baseline architecture, we resize each training image to 64x64. Some examples of the training images are presented in Figure 2

Each image has 5 associated captions, each of which we

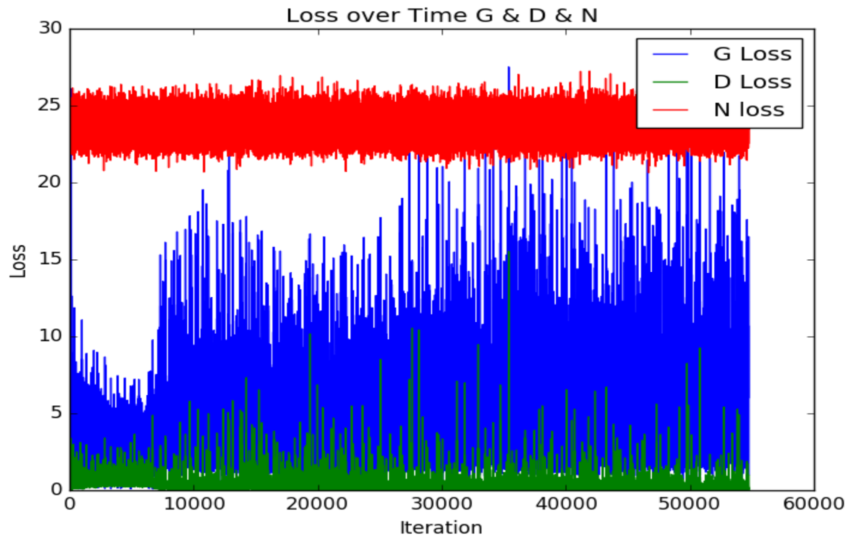


Figure 3. Dual-Loss GAN training loss over time

treat as an individual training example during training. Correspondingly, each associated image is copied 5 times in the training set for ease of integration with Batch Gradient Descent packages built into Torch. Since we had approximately 10,000 images of vehicles, our training set consisted of about 50,000 training examples.

4.3. Conditional Vector

To encode captions as a fixed size conditional vector, we converted the caption string to an embedding space using two different method: Google's *Word2Vec* tool [8, 9] and the Visual Semantic Embedding (VSE) proposed by Kiros et al. [6]. *Word2Vec* creates a 300-dimensional vector representation for a given word. Since each vector shows linear properties with respect to word content(i.e. vec(”King”) - vec(”Man”) + vec(”Woman”) $\approx$ vec(”Queen”)), we simply added the vectors. With the Visual Semantic Embedding, the method uses a deep network to map captions and their associated pictures to the same 1024-dimensional embedding space. For this part, we simply took a pre-trained embedding network and mapped each caption to an embedding vector.

In our experiments, we empirically tested both metrics but ended up going with the *Word2Vec* implementation. This decision was made partly because of the easy integration with Torch and the low dimensionality of its representation. The 1024-dimensional conditional vector made the architecture quite cumbersome to achieve stable training. Pre-trained models were only available in Theano, making it difficult to pass data to and from our Torch implementation. However, the primary reason for our decision to use *Word2Vec* was that it gave much better results than VSE. We speculate that this was due to a combination of the quality of the embedding and the higher dimensionality of VSE causing the network to over fit to conditioning information.

5. Results and Analysis

In our experiments, we compare the results from the following models:

(1) A standard cDCGAN where conditional information is added to the input of the G and D network.

(2) Our proposed network architecture as described in Section 3.1

Through this comparison, we were able to evaluate the quality of generated images from our proposed network architecture with respect to the baseline cDCGAN model. In addition, visualizing the output of the cDCGAN helped us analyze the effects of using word embeddings as conditional information.

5.1. Training Details

We used mostly the same training parameters for both experiments, barring some small changes for the new, proposed architecture. All models were trained with mini-batch stochastic gradient descent (SGD) with a mini-batch size of 64. All weights were initialized from a zero-centered Normal distribution with standard deviation 0.02. In the LeakyReLU, the slope of the leak was set to 0.2 in all models. Following the suggestions of Radford et al., we used an Adam optimizer [5] along with with a learning rate of .0002. Although this learning rate led to somewhat noisy gradient updates, the networks displayed worse performance with both higher and lower values.

The baseline parameters for the Dual-Loss model were similar in regards to optimization. However, we introduced weighting schemes to gradients from each loss function. The gradients obtained from the discriminator model were several orders of magnitude higher than those of the LSTM model, while LSTM scalar loss was around 20 times larger than discriminator loss. We experimented with weighting the LSTM gradient with β_G values of up to 500, but found little effect on the LSTM loss values. We weighted the neural talk neural loss by a β_L value of 0.005 to avoid accelerating the discriminator loss preemptively. α was kept at 1 for the discriminator network. We also experimented with running a cDCGAN without the LSTM loss, and adding the LSTM loss later on in the training schedule, but found similar results to always including the LSTM loss.

5.2. cDCGAN Results

Our cDCGAN training loss displayed some noisy convergence in the maximization of D Loss and the minimization of the G Loss. As shown in Figure 6, there is an overall descent in the D Loss in the first few epochs, from which the loss starts to oscillate around lower values. Similarly, the generator error shows a general increase in the first few epochs, but still does not reach a stable convergence point in later epochs. This behavior has been observed by both Radford et al. [12] and Goodfellow et al. [4], as their studies have shown that the minimax framework makes learning quite noisy. However, in the scope of a large number of iterations, both Radford et al. and Gauthier [3] show diverging losses, which is not seen here. These results suggest that the conditional information is affecting the adversarial learning negatively. The use of word embeddings as conditional information result in very noisy estimates of the conditional generative distribution.

Although network loss is a good guide for monitoring stable training, it can be somewhat misleading, as very noisy, low resolution images of an object can lead to incorrect predictions and thus high loss [11]. As a result, we looked at network generated images so that we could intuit what kind of images the network was learning. As shown in Figure 4, these images seem quite hazy and lack definite structure, yet some of them capture the general essence of the caption in question. This is somewhat promising, as the results suggest that the G network is able to create object representations that are reasonable, albeit very noisy.

The noisy images generated and the lack of loss divergence point to the inefficacy of the conditional vector in modeling the conditional distribution. Since *Word2Vec* representations of similar captions may be close in metric space but not close in terms of values of each entry of the conditional vector, the network has greater difficulty using

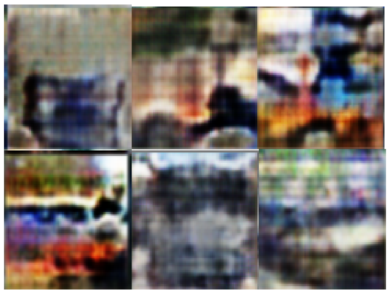
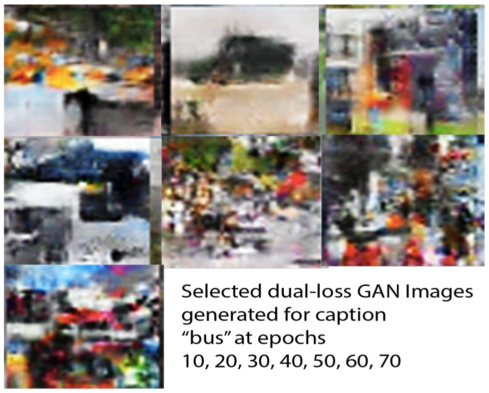


Figure 4. cDCGAN results for the caption ”bus” at epochs 25, 50, 75, 100, 125, and 150


Selected dual-loss GAN Images generated for caption ”bus” at epochs 10, 20, 30, 40, 50, 60, 70
Figure 5. Dual-Loss GAN results.

values of the vector to capture the conditional distribution. This is opposed to previous uses of conditional vectors, such as one-hot encodings of class labels, where the vectors are sparse and possess many training images per conditional vector. In our case, each of the captions is quite specific, resulting in vector representations with only few training examples per value. We believe this to be the reason for our noisy results and, more generally, the pitfall of using caption embeddings as conditional information.

5.3. Dual-Loss GAN Results

Our Dual-Loss GAN training did not result in minimization of the LSTM loss. We suspect that the LSTM and Discriminator loss functioned at odds to each other, instead of the LSTM loss guiding the GAN combination to semantic representations. We also believe that the pretrained LSTM that was used may have had too many intermediate processing steps to result in effect gradient updates to the generator network, as it had to upsample input images, generate VGG-16 net features, and backpropagate gradients through both the LSTM network and VGG-16 net. This led to both the LSTM loss not updating and the generator loss increasing, discriminator loss decreasing, instead of the desired pattern, as can be seen in Figure 3. Figure 5 displays generated images from different epochs of training. These are

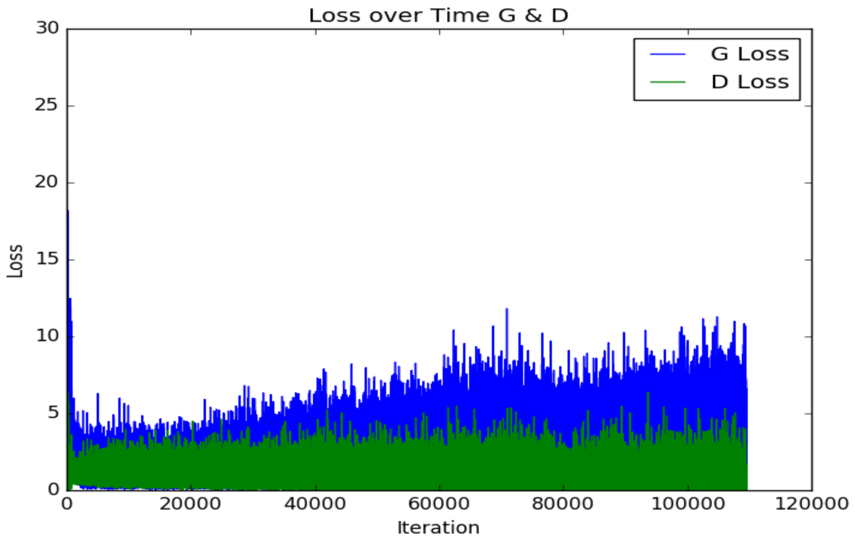


Figure 6. cDCGAN Training Loss Over Time

selected images, however, and generally, generated images tended not to be as representative of their captions as the selected images. However, we can still see that the network was learning to produce more realistic "bus"-like images at epoch 20, before tending towards more noisy images later in training.

6. Conclusion and Future Work

Generative adversarial network models are still in their infancy, and we believe there are many problems they can be adapted to in the future. There remains a great deal of work to be done in achieving stable GAN training, as our work has convinced us of the fragility of even the most stable GAN architectures. The problem turned out to be much more complex than adding conditional word embeddings, as these conditional vectors were too dense to actually estimate conditional distributions. However, we still believe that generating captioned images with GANs is a solvable problem. We also believe that our reliance on a pre-trained captioning model and lack of finetuned training schedule were the main detractors in obtaining better results, and further research can develop more appropriate methods to solve this problem.

Further experiments might want to develop a more direct link between conditioning caption information and generated loss by training a sentence-image embedding network, or might focus on stably generating larger images (which most captioning models are trained on) before adding caption loss.Since the baseline cDCGAN also showed very noisy results for the given dataset, we believe that a conditional vector which isolates individual features of the images (i.e color, shape, position, etc.) will model the conditional distribution better. Some avenues for exploration include using normalized bag-of-words vectors as conditional information or using a combination of class label one-hot encodings in conjunction with *Word2Vec* representations. One other mode of future work would be to refine the granularity of the generated images. Since our current DCGAN implementation can only generate images that are 64x64, we in turn lose information by downsampling the input COCO images, which are much larger. A reliable size increase to, say, 128x128, might make for more effective semantic image generation.

References

[1] R. Collobert, S. Bengio, and J. Marithoz. Torch: A modular machine learning software library, 2002.

[2] E. L. Denton, S. Chintala, A. Szlam, and R. Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. *CoRR*, abs/1506.05751, 2015.

[3] J. Gauthier. Conditional generative adversarial nets for convolutional face generation. 2014. Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester 2014.

[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial networks. *CoRR*, abs/1406.2661, 2014.

[5] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[6] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th*

Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States., pages 1106–1114, 2012.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[9] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[10] M. Mirza and S. Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[11] A. M. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *CoRR*, abs/1412.1897, 2014.

[12] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.

[13] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014.

7. Code and Materials

Main Folder: https://github.com/PCerles/ImageGen
DCGAN Code: https://github.com/PCerles/dcgan.torch*
Captioning Code: https://github.com/PCerles/neuralTalk2*
* These repositories are subfolders of the Main directory