**Introduction**

One of the most challenging problems in the world of Computer Vision is synthesizing high-quality images from text descriptions. No doubt, this is interesting and useful, but current AI systems are far from this goal. In recent years, powerful neural network architectures like GANs (Generative Adversarial Networks) have been found to generate good results. [1] Samples generated by existing text-to-image approaches can roughly reflect the meaning of the given descriptions, but they fail to contain necessary details and vivid object parts [2]. Through this project, we wished to explore architectures that could help us achieve our task of **generating images from given text descriptions**. Generating photo-realistic images from text has tremendous applications, including photo-editing, computer-aided design, etc.

**Example**

**Text description:**This white and yellow flower has thin white petals and a round yellow stamen.

## Generated Images:



Fig. 1.1 Example of image generated from the text description

Converting natural language text descriptions into images is an exceptional demonstration of Deep Learning. Text classification tasks such as sentiment analysis have been successful with Deep Recurrent Neural Networks that are able to learn discriminative vector representations from text. In another domain, Deep Convolutional GANs are able to synthesize images such as interiors of bedrooms from a random noise vector sampled from a normal distribution. The focus of Reed et al. [1,2] is to connect advances in Deep RNN text embeddings and image synthesis with DCGANs, inspired by the idea of Conditional-GANs.

Conditional-GANs work by inputting a one-hot class label vector as input to the generator and discriminator in addition to the randomly sampled noise vector. This results in higher training stability, more visually appealing results, as well as controllable generator outputs. The difference between traditional Conditional-GANs and the Text-to-Image model lies in the way the input is conditioned. Instead of trying to construct a sparse visual attribute descriptor to condition GANs, the GANs are conditioned on a text embedding learned with a Deep Neural Network. A sparse visual attribute descriptor might describe “a red bird with a white beak” as something like:

[ 0 0 0 1 . . . 0 0 . . . 1 . . . 0 0 0 . . . 0 0 1 . . .0 0 0]

The ones in the vector would represent attribute questions such as, red (1/0)? white (1/0)? bird (1/0)? This description is difficult to collect and doesn’t work well in practice.

Word embeddings have been the hero of natural language processing through the use of concepts such as Word2Vec. Word2Vec forms embeddings by learning to predict the context of a given word. Unfortunately, Word2Vec doesn’t quite translate to text-to-image since the context of the word doesn’t capture the visual properties as well as an embedding explicitly trained to do so does. Reed et al. [1,2] present a novel symmetric structured joint embedding of images and text descriptions to overcome this challenge.

In addition to constructing good text embeddings, traslation of image from text is a highly multi-modal task. The term ‘multi-modal’ is an important one to become familiar with in Deep Learning research. This refers to the fact that there are many different images of birds with correspond to the text description “bird”. Another example in speech is that there exist multiple different accents, that would result in different sounds corresponding to the text “bird”. Multi-modal learning is also present in image captioning (image-to-text). However, this is greatly facilitated due to the sequential structure of text such that the model can predict the next word conditioned on the image as well as the previously predicted words. Multi-modal learning is traditionally very difficult, but is made much easier with the advancement of GANs (Generative Adversarial Networks), this framework creates an adaptive loss function which is well-suited for multi-modal tasks such as text-to-image.

**GANs**

Generative Adversarial Networks, or GANs for short, are an approach to generative modeling using deep learning methods, such as convolutional neural networks.

Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that the model can be used to generate or output new examples that plausibly could have been drawn from the original dataset.

GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with two sub-models: the generator model that we train to generate new examples, and the discriminator model that tries to classify examples as either real (from the domain) or fake (generated). The two models are trained together in a zero-sum game, adversarial, until the discriminator model is fooled about half the time, meaning the generator model is generating plausible examples.

GANs are an exciting and rapidly changing field, delivering on the promise of generative models in their ability to generate realistic examples across a range of problem domains, most notably in image-to-image translation tasks such as translating photos of summer to winter or day to night, and in generating photorealistic photos of objects, scenes, and people that even humans cannot tell are fake.

### GANs and Convolutional Neural Networks

GANs typically work with image data and use Convolutional Neural Networks, or CNNs, as the generator and discriminator models.

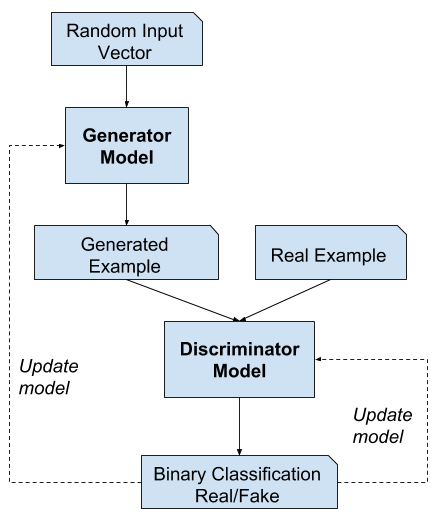
The reason for this may be both because the first description of the technique was in the field of computer vision and used CNNs and image data, and because of the remarkable progress that has been seen in recent years using CNNs more generally to achieve state-of-the-art results on a suite of computer vision tasks such as object detection and face recognition.

Modeling image data means that the latent space, the input to the generator, provides a compressed representation of the set of images or photographs used to train the model. It also means that the generator generates new images or photographs, providing an output that can be easily viewed and assessed by developers or users of the model.

It may be this fact above others, the ability to visually assess the quality of the generated output, that has both led to the focus of computer vision applications with CNNs and on the massive leaps in the capability of GANs as compared to other generative models, deep learning based or otherwise.

### Conditional GANs

An important extension to the GAN is in their use for conditionally generating an output.



The generative model can be trained to generate new examples from the input domain, where the input, the random vector from the latent space, is provided with (conditioned by) some additional input.

The additional input could be a class value, such as male or female in the generation of photographs of people, or a digit, in the case of generating images of handwritten digits.

The discriminator is also conditioned, meaning that it is provided both with an input image that is either real or fake and the additional input. In the case of a classification label type conditional input, the discriminator would then expect that the input would be of that class, in turn teaching the generator to generate examples of that class in order to fool the discriminator.

In this way, a conditional GAN can be used to generate examples from a domain of a given type.

Taken one step further, the GAN models can be conditioned on an example from the domain, such as an image. This allows for applications of GANs such as text-to-image translation, or image-to-image translation. This allows for some of the more impressive applications of GANs, such as style transfer, photo colorization, transforming photos from summer to winter or day to night, and so on.

In the case of conditional GANs for image-to-image translation, such as transforming day to night, the discriminator is provided examples of real and generated nighttime photos as well as (conditioned on) real daytime photos as input. The generator is provided with a random vector from the latent space as well as (conditioned on) real daytime photos as input.

**Reason for selecting GANs**

One of the many major advancements in the use of deep learning methods in domains such as computer vision is a technique called [data augmentation](https://machinelearningmastery.com/image-augmentation-deep-learning-keras/).

Data augmentation results in better performing models, both increasing model skill and providing a regularizing effect, reducing generalization error. It works by creating new, artificial but plausible examples from the input problem domain on which the model is trained.

The techniques are primitive in the case of image data, involving crops, flips, zooms, and other simple transforms of existing images in the training dataset.

Successful generative modeling provides an alternative and potentially more domain-specific approach for data augmentation. In fact, data augmentation is a simplified version of generative modeling.

In complex domains or domains with a limited amount of data, generative modeling provides a path towards more training for modeling. GANs have seen much success in this use case in domains such as deep reinforcement learning.

There are many research reasons why GANs are interesting, important, and require further study. Ian Goodfellow outlines a number of these in his 2016 conference keynote and associated technical report titled “NIPS 2016 Tutorial: Generative Adversarial Networks.”

Among these reasons, he highlights GANs’ successful ability to model high-dimensional data, handle missing data, and the capacity of GANs to provide multi-modal outputs or multiple plausible answers.

Perhaps the most compelling application of GANs is in conditional GANs for tasks that require the generation of new examples. Here, Goodfellow indicates three main examples:

* **Image Super-Resolution**. The ability to generate high-resolution versions of input images.
* **Creating Art**. The ability to great new and artistic images, sketches, painting, and more.
* **Image-to-Image Translation**. The ability to translate photographs across domains, such as day to night, summer to winter, and more.

Perhaps the most compelling reason that GANs are widely studied, developed, and used is because of their success. GANs have been able to generate photos so realistic that humans are unable to tell that they are of objects, scenes, and people that do not exist in real life.

**Methodology**

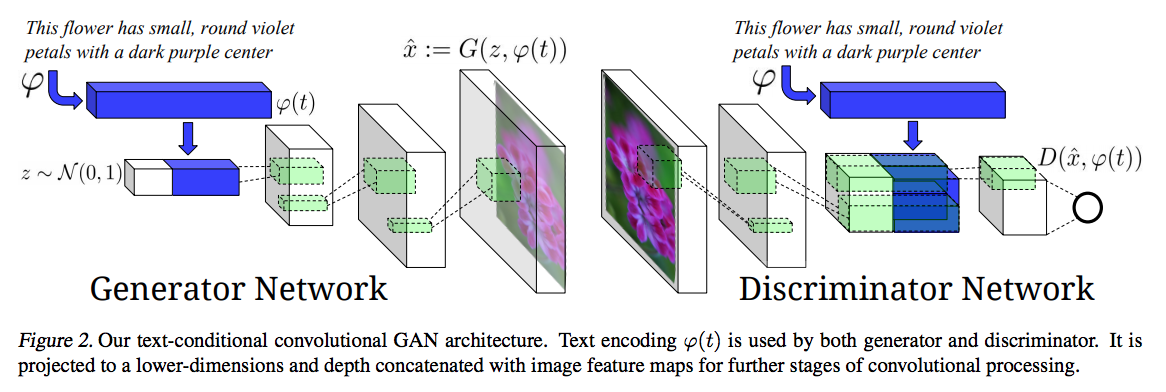


Fig. 4.1 Text conditional convolutional GAN Architecture

The above image depicts the architecture Reed et al. used to train their text-to-image GAN model. The most noteworthy takeaway from this diagram is the visualization of how the text embedding fits into the sequential processing of the model. In the Generator network, the text embedding is filtered trough a fully connected layer and concatenated with the random noise vector z. In the following scenario, the text embedding is converted from a 1024x1 vector to 128x1 and concatenated with the 100x1 random noise vector z. On the side of the discriminator network, the text-embedding is also compressed through a fully connected layer into a 128x1 vector and then reshaped into a 4x4 matrix and depth-wise concatenated with the image representation. This image representation is derived after the input image has been convolved over multiple times, reduce the spatial resolution and extracting information. The following strategy used for the embeddings for the discriminator is different from the conditional-GAN model in which the embedding is concatenated into the original image matrix and then convolved over.

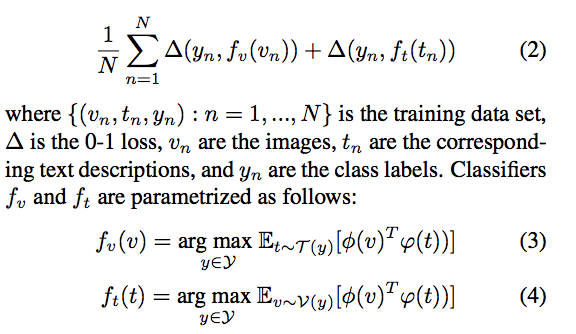
In the architecture diagram we visualize how the DCGAN upsamples vectors or low-resolution images to produce high-resolution images. You can see each de-convolutional layer increases the spatial resolution of the image. Additionally, the depth of the feature maps decreases per layer. Lastly, you can see how the convolutional layers in the discriminator network decreases the spatial resolution and increase the depth of the feature maps as it processes the image.

In the following training process, it is difficult to separate loss based on the generated image not looking realistic or loss based on the generated image not matching the text description. The authors of the paper describe the training dynamics being that initially the discriminator does not pay any attention to the text embedding, since the images created by the generator do not look real at all. Once the Generator generate images that pass the real vs. fake criteria, then we decide factoring in the text embedding.

The authors smooth out the training dynamics of this by adding pairs of real images with incorrect text descriptions which are labeled as ‘fake’. The discriminator’s sole motive is the binary task of real vs. fake and is not separately considering the image apart from the text. This is in contrast to an approach such as AC-GAN with one-hot encoded class labels. The AC-GAN discriminator outputs real vs. fake and uses an auxiliary classifier sharing the intermediate features to classify the class label of the image.

**Constructing a Text Embedding for Visual Attributes**

The most interesting component of this paper is how a unique text embedding is constructed that contains visual attributes of the image to be represented as vectors. This vector is constructed through the following process:

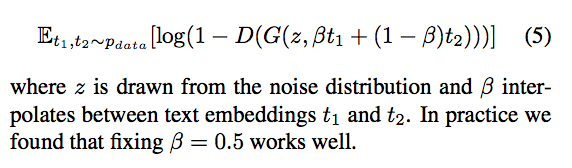


The loss function noted as equation (2) represents the overall objective of a text classifier that is optimizing the gated loss between two loss functions. These loss functions are shown in equations 3 and 4. The paper describes the intuition for this process as “A text encoding should have a higher compatibility score with images of the corresponding class compared to any other class and vice-versa”. The two terms represent an image encoder and a text encoder respectively. The image encoder is derived from the GoogLeNet image classification model. This classifier reduces the dimensionality of images upto the point where it is compressed to a 1024x1 vector. The objective function thus aims to minimize the distance between the image representation from GoogLeNet and the text representation from a character-level CNN or LSTM. Essentially, the vector encoding for the image classification is used to guide the text encodings based on similarity to similar images.

The details of this are expanded on in the following paper, “Learning Deep Representations of Fine-Grained Visual Descriptions” also from Reed et al.

**Manifold interpolation**

One of the interesting characteristics of Generative Adversarial Networks is that the latent vector z can be used to interpolate new instances, commonly referred to as “latent space addition”. An example would be to do “man with glasses” — “man without glasses” + “woman without glasses” and achieve a woman with glasses. In this paper, the authors aims to interpolate between the text embeddings. This is done with the following equation:



The discriminator has been trained to predict if image and text pairs differ or not. Therefore, images from interpolated text embeddings can fill in the gaps in the data manifold that were present during training. Using this as a regularization method for the training data space is paramount for the successful result of the model presented in this paper. This is a form of data amplification since the interpolated text embeddings can enlarge the dataset used for training the text-to-image GAN.

**Stac**kGan[3]

The aim here was to generate high-resolution images with photo-realistic details. The authors proposed an architecture where the process of generating images from text is decomposed into two stages as shown in Figure 6. The two stages are as follows:

**Stage-I GAN:**The primitive shape and basic colors of the object (con- ditioned on the given text description) and the background layout from a random noise vector are drawn, yielding a low-resolution image.

**Stage-II GAN:**The defects in the low-resolution image from Stage-I are corrected and details of the object by reading the text description again are given a finishing touch, producing a high-resolution photo-realistic image.

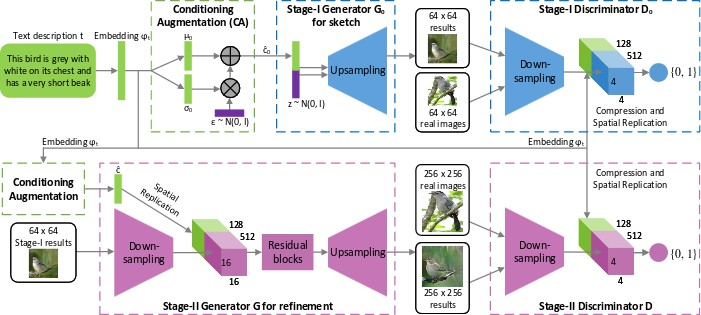


Fig. 4.2 Network Architecture of StackGAN

* **StackGAN++**[5]

This is an extended version of StackGAN discussed earlier. It is an advanced multi-stage generative adversarial network architecture consisting of multiple generators and multiple discriminators arranged in a tree-like structure. The architecture generates images at multiple scales for the same scene. Experiments demonstrate that this new proposed architecture significantly outperforms the other state-of-the-art methods in generating photo-realistic images. Fig. 4.2 shows the architecture.

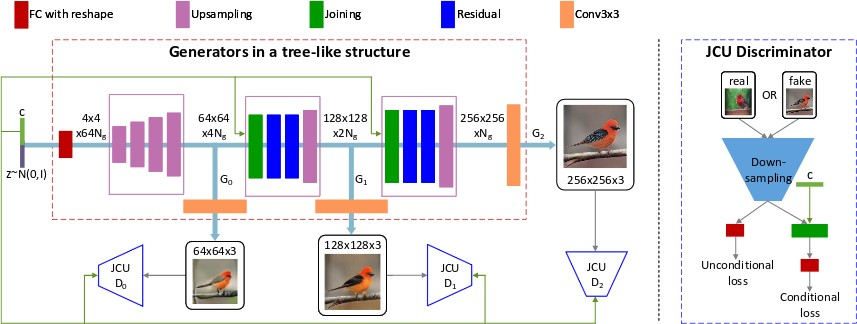


Fig. 4.3 Network Architecture of StackGAN++

# Part 1: Multi-stage Image Refinement (the AttnGAN)

The **Attentional Generative Adversarial Network** (or AttnGAN) begins with a crude, low-res image, and then improves it over multiple steps to come up with a final image.

Lets start with the first stage:

## Initial step

Like most other Text-to-Image convertors, AttnGAN starts off by generating an image from (random noise + a summation of the caption’s token-embeddings):

h(0) = F(0)(z, E)

Here, z represents the noise-input, and E represents the sum of individual word-vectors. h(0) denotes as the ‘hidden context’ — essentially, AttnGAN’s concept of what the image should look like. Based on h(0), we generate x(0) — the first image — using a GAN:

x(0) = G(0)(h(0))

(Corresponding to the Generator G(0) we also have the Discriminator D(0), which we will talk about later.)

An example of x(0) straight fromthe paper:

Caption: “This bird has a green crown black primaries and a white belly”

https://miro.medium.com/max/30/1*GBjcSqO-q1MjTLSeXjPBRQ.png?q=20



## Further iterations

One of the issues with generating an image from a combined ‘sentence’ vector (E above), is that we lose a lot of the fine-grained details hidden in individual words.

For instance, look at the example above: When you combine (green+crown+white+belly) into a ‘bag-of-words’, you are much, much less likely to understand the actual colors of the crown & belly — hence the hazy coloring in the generated image.

To remedy this, AttnGAN uses a combination of Attention & GAN at every stage, to iteratively add details to the image:

h(i) = F(i)(h(i-1), Attn([e], h(i-1)))

x(i) = G(i)(h(i))

h(1), h(2), … follow the template above.

Compare these to the initial equations:

* z gets replaced by the previous context h(i-1).
* [e] denotes the set of all [word-embeddings](https://en.wikipedia.org/wiki/Word_embedding) in the sentence. Using Attention based on h(i-1), we compute a weighted average of [e] ( Attn([e], h(i-1)) ) **to highlight words that need more detail**.
* Based on this weighted vector, F(i) alters h(i-1) to yield h(i).
* As usual, a GAN is then used to produce x(i) from h(i).

Continuing with the previous example:

Top attended words for h(1): bird, this, has, belly, white

Top attended words for h(2): black, green, white, this, bird

https://miro.medium.com/max/30/1*un6245RICK0oUXdJoEYS0Q.png?q=20



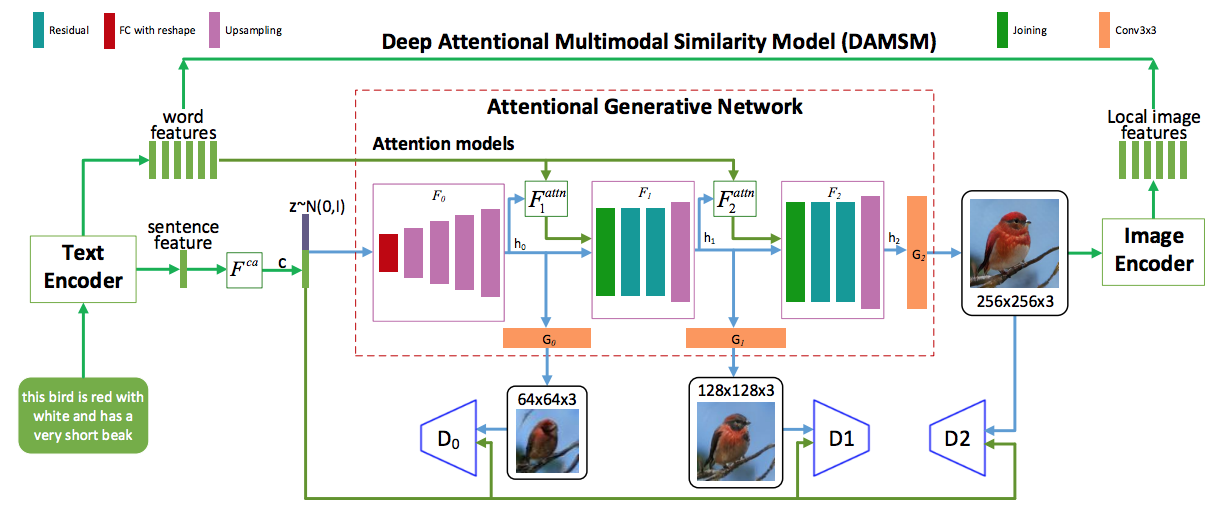
Corresponding images (x(1) & x(2))

Consider the words for h(2). You can literally see x(2) being a more colorful version of x(1).

Ofcourse the results aren’t always so pretty, but its a step in the right direction for optimizing the correct objectives. Which brings us to…

# Part 2: Multi-modal loss

At this point, it will be good for you to go through the high-level diagram of the system, given in the paper:



Lets consider the parts we haven’t touched upon as yet.

## The Discriminators

Looking at the equations for h & x, it is natural to wonder why we need the x’s at all, except at the last step. For example, x(0) does not appear in the equations for h(1) & x(1)!

The reason is — training. In the learning phase, the D’s are trained with scaled-down versions of real image-caption examples (from a dataset like [COCO](http://cocodataset.org/#explore)). This makes the G’s better at generating x’s from the h’s. By rules of back-propagation, this makes the F functions better at generating the hidden contexts — thereby ensuring that each stage adds something meaningful to the image.

## The Deep Attentional Multimodal Similarity Model (DAMSM)

After the concept of multi-stage image refinement, I think this is the second key feature of this framework.

While the individual discriminators do make the system better, we do not yet have an objective that checks if every single word in the caption is appropriately represented in the actual image (the discriminators are trained on the overall caption E & scaled-down image pairs).

To encode this task effectively, we first train an ‘expert’ of sorts — the DAMSM. DAMSM takes as input an image and the set [e], and provides feedback on how well the two ‘fit’ together. It does this as follows:

* Using a standard Convolutional Neural Network, the image is converted into a set of feature maps. Each feature map essentially signifies some concept/sub-region in the image.
* The dimensionality of the feature maps is made equal to that of the word embeddings, so that they can be treated as equivalent entities.
* Based on each token in the caption, Attention is applied over the feature maps, to compute a weighted average of them. This attention-vector essentially represents the image’s abstraction of the token.
* Finally, DAMSM is trained to minimize the difference between the above attention-vector (visual portrayal of the word) & the word-embedding (textual meaning of the word). You are basically trying to make the ‘green’ part of the image as ‘green’ as possible.

The reason DAMSM is called ‘**multimodal**’, is because it defines an objective that **combines two modes of understanding — visual & textual**.

Once DAMSM has been sufficiently trained on a dataset, it can then be used in conjunction with the step-wise discriminators, to provide a rich target for AttnGAN to optimize.

**References**

[1] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee. Generative Adversarial Text to Image Synthesis. 2016.

[2] Scott Reed, Zeynep Akata, Bernt Shiele, Honglak Lee. Learning Deep Representations of Fine-grained Visual Descriptions. 2016.

[3] Zhang, Han, et al. ”Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks.” arXiv preprint (2017).

[4] Goodfellow, Ian, et al. ”Generative adversarial nets.” Advances in neural information processing systems. 2014.

[5] Zhang, Han, et al. ”Stackgan++: Realistic image synthesis with stacked generative adversarial networks.” arXiv preprint arXiv:1710.10916 (2017).

[6] Nilsback, Maria-Elena, and Andrew Zisserman. ”Automated flower classifi- cation over a large number of classes.” Computer Vision, Graphics & Image Processing, 2008. ICVGIP’08. Sixth Indian Conference on. IEEE, 2008.