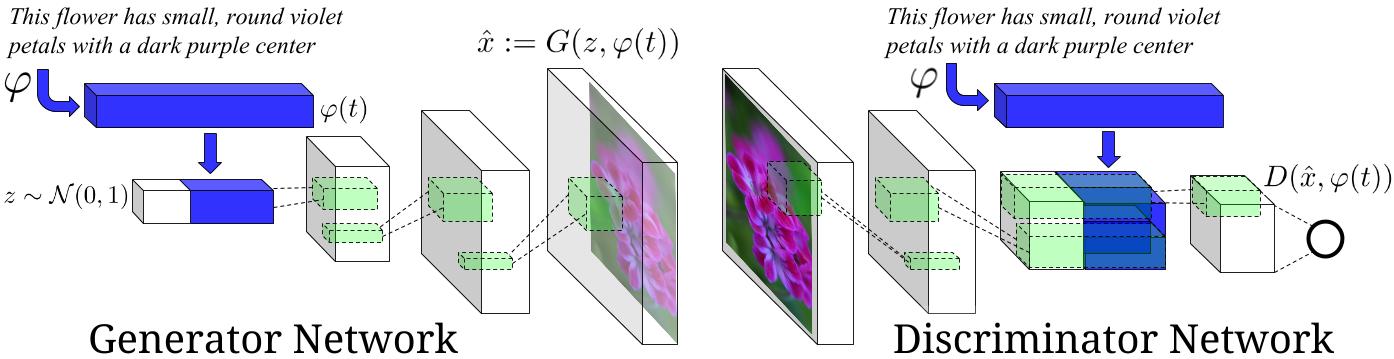
**TEXT-TO-IMAGE GENERATION USING DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORK**

**ARCHITECTURE:**

****

**PRE-PROCESSED AND PREPARED DATA:**

The data is prepared, pre-processed and stored into ./text-to-image directory as .npy files.

1. train\_captions.npy
2. train\_images.npy

**Dependencies:**

* python 3.5
* TensorFlow 1.1
* [Optional] [Torch](http://torch.ch/docs/getting-started.html#_) is needed, if using the pre-trained char-CNN-RNN text encoder.
* The word to image caption and caption to the word mapping are done and stored into ./dictionary directory as:

1. id2Word.npy
2. word2ID.npy
3. vocab.npy

**EXECUTION:**

* Run ./Text-to-Image-GAN/train.py to train the DCGAN model on the Oxford 102-Flowers dataset using the pre-trained Word-ID embedding Numpy array model.
* The results will be stored to ./Text-to-Image-GAN/train\_samples/.
* If you want to try your own datasets, we encourage to try different hyper-parameters and architectures, especially for more complex datasets like MS-COCO datasets.
* Inorder to improve the performance of the DCGAN, you can also use other techniques as follows:

1. Normalize the inputs
2. A modified loss function
3. Batch normalization
4. Avoid sparse gradients: ReLU, MaxPool

**RESULT:**

**AFTER 600 EPOCH, (**We assume better results can be achieved by playing with the hyper-parameters)

**INPUT SENTENCES:**

* The flower shown has yellow anther red pistil and bright red petals.
* This flower has petals that are yellow, white and purple and has dark lines
* The petals on this flower are white with a yellow center
* This flower has a lot of small round pink petals.
* This flower is orange in color, and has petals that are ruffled and rounded.
* The flower has yellow petals and the center of it is brown
* This flower has petals that are blue and white.
* These white flowers have petals that start off white in color and end in a white towards the tips.

****