IMAGE GENERATION FROM CAPTION

Mahima Pandya and Prof. Sonal Rami

[1]CSPIT, Charotar University of Science and Technology, Changa, India

ABSTRACT

Generating images from a text description is as challenging as it is interesting. The Adversarial network performs in a competitive fashion where the networks are the rivalry of each other. With the introduction of Generative Adversarial Network, lots of development is happening in the field of Computer Vision. With generative adversarial networks as the baseline model, studied Stack GAN consisting of two-stage GANS step-by-step in this paper that could be easily understood. This paper presents visual comparative study of other models attempting to generate image conditioned on the text description. One sentence can be related to many images. And to achieve this multi-modal characteristic, conditioning augmentation is also performed. The performance of Stack-GAN is better in generating images from captions due to its unique architecture. As it consists of two GANS instead of one, it first draws a rough sketch and then corrects the defects yielding a high-resolution image.

KEYWORDS

Adversarial Networks, Stack-GAN.

1.INTRODUCTION

Artificial intelligence is the development of computer systems to be able to perform tasks that humans can do by applying intelligence such as computer-vision, speech-recognition, Natural Language processing, reasoning, Knowledge representations and so-on. So, the main concept is to make the machines do what humans are capable of doing in their daily lives such as imitating handwritings, classifying images, identifying emotions of a person. Such as that, painting is another ability of humans. The Painting has been an important and unique skill of humans. Since ancient time, different and more complex paintings have evolved. A human can easily draw an image or replicate it. She can draw based on her imaginations with a description in mind. As still, computers do not have their own thinking capacity, text description shall be provided by the user. The input shall be a sentence i.e. text description of the image. The generated image shall be the output of the algorithm. To achieve this goal, certain issues have to be considered. The system has to understand the relationship between the objects. For example, if the description is `a cat sitting on a chair'. The computer algorithm has to take care of the semantics of the sentence such as the prepositions and plurals or singulars.

DOI :10.5121/ijscai.2018.7201

Figure 1. Example of images conditioned on the caption: 'This bird is completely red with black wings and pointy beak'

Considering all those factors, an image is generated with the best resolution.

2.RELATED WORK

Recently, in last few years a lot of advancements have taken place in automatic synthesis of Images from human-given descriptions of objects to be generated. Many researchers have contributed to achieving this objective. The Generative Adversarial Nets proposed by Ian Goodfellow and other researchers [**19**] demonstrated the capacity of this framework consisting of two networks namely Generator G and Discriminator D. The former learns the data distribution and the latter, D determines the probability that the sample belongs to the training set rather than generated from the Generator. The model is trained in an adversarial manner where G tries to maximize the probability of D giving the wrong output and vice-a-versa like min-max players. This mechanism remains the base of some of the other works such as Stack-GAN. In a multimodal learning study by ReedScot [**4**], Generative Adversarial text to image synthesis entirely based on GAN structure for generating images conditioned on a text description. Deep symmetric structured joint embedding to convert text into vector representations using convolutional and recurrent neural networks [**5**]. They used bird and flower images to train the model to prove the effectiveness of deep text-conditional Convolutional GAN architecture. It effectively produced 64×64 resolution image.

PixelBrush, a tool proposed by Zhi Jiale [**10**] for art generation from a text description. It uses Generative adversarial networks to produce artistic images conditioned on human-written descriptions. Skip-thoughts are used for text embedding of 4800 dimension and conditional generator and discriminator network for generating images. It carried out a comparative analysis of different generator network by using inception score. According to the published research paper, the resolution is limited to 64x64. Recurrent Adversarial networks [**8**] have also proved to be efficient in mapping natural language input to visual images inspired by the model proposed by Gatys [**12**]. The recurrent model learns to optimize itself without using any coarse-to-fine structure. In Another proposed solution [**13**], introduced a conditional alignDRAW model, extension of DRAW-Deep Recurrent Attention Writer [**14**] that used recurrent neural network instead of Convolutions to generate images using soft attention mechanism.

Han Zhang [**1**] put forward Stacked Generative adversarial network. As the name suggests it comprises two-stage GAN. To produce diversified images, it includes a novel approach for applying conditioning augmentation for producing more training samples for better results. Stage-I GAN produces a sketch of basic shapes, colors and other characteristics with low-resolution satisfying the text description. Stage-II GAN corrects the defects of Stage-I produced image. It also provides finishing to the features of the objects synthesizing a more photo-realistic image. Moreover, it compared the resulted images with other architectures such as GAN-INT-CLS [**7**] that generated 64x64 resolution images and GAWWN [**6**] 128x128 resolution images. In

comparison to other models, Stack-GAN architecture generated more photo-realistic images of up to 256x256 resolution.

3.BACKGROUND THEORY

The following section provides an overview of the concepts used in this model architecture.

3.1. Skip-thought Vector

An unsupervised approach [**15**] that consists of encoder and decoder model which can be RNN-RNN [**20**], ConvNet-RNN [**23**] or LSTM-LSTM [**21**]. It is the modification of skip-gram [**16**,**18**] model where instead of determining the neighboring words, it predicts the surrounding sentences of a given sentence. Given a tuple (S_{i-1}, S_i ,S_{i+1}), it reconstructs the previous and the next sentence of S_i. As mentioned in [**15**], RNN with GRU [**22**] activations is used as encoder and RNN with conditional GRU as a decoder.

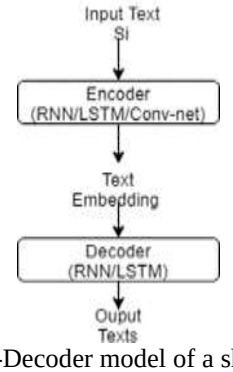

Figure 2. Encoder-Decoder model of a skip-thought vector.

3.1.1 Encoder

Given, S_i: a sentence, N: #words in S_i, w_i^1,….., w_i^N: words in S_i, h_i^t: hidden state produced by the encoder at each time stamp, representation of w_i^1,…..,w_i^t, h_i^N: hidden state produced by the encoder, representation of w_i^1,….., w_i^N.
The encoder iterates through these equations to map an English sentence to text embedding.

$$r^t = \sigma(W_r x^t + U_r h^{t-1}) \quad (1)$$
$$z^t = \sigma(W_z x^t + U_z h^{t-1}) \quad (2)$$
$$\vec{h}^t = tanh(Wx^t + U(r^t \odot h^{t-1})) \quad (3)$$
$$h^t = (1-z^t) \odot h^{t-1} + z^t \odot \vec{h}^t \quad (4)$$

Notations: r^t: reset gate, Θ: component-wise product, z^t: update gate, $\bar{h}^t$: proposed state update at time t

3.1.2 Decoder

The computations of RNN decoder with conditional GRU are similar to that of the encoder. It takes the output of the encoder h_i and adds the matrices C_z, C_r, and C as bias to update gate, reset gate and proposed state update respectively. One decoder is used for the computation of previous sentence S_{i-1} and another decoder for the next sentence S_{i+1}. The decoder iterates through these equations to reconstruct the surrounding equation.

$$r^t = \sigma(W_r^d x^{t-1} + U_r^d h^{t-1} + C_r h_i) \quad (5)$$

$$z^t = \sigma(W_z^d x^{t-1} + U_r^d h^{t-1} + C_z h_i) \quad (6)$$

$$\vec{h}^t = tanh(W^d x^{t-1} + U^d(r^t \odot h^{t-1}) + Ch_i) \quad (7)$$

$$h^t_{i+1} = (1-z^t) \odot h^{t-1} + z^t \odot \vec{h}^t \quad (8)$$

$$P(w^t_{i+1} | w^{<t}_{i+1}, h_i) \propto exp(v_{w^t_{i+1}} | h^t_{i+1}) \quad (9)$$

Eq. (9), specifies the encoder vector given $v_{w^t_{i+1}}$: a row of V corresponding to the word of w^t_{i+1}. The above equations can be modified for the computations of S_{i-1}.

3.2 Adversarial Nets

Adversarial nets as the name suggest comprises networks acting as adversaries to each other in order to make itself better.

3.2.1 Generative Adversarial Nets

Generative Adversarial Nets are deep neural networks consisting of two networks competing with each other analogous to a counterfeiter and investigator. The counterfeiter tries to replicate by learning from the original sample and the investigator determines whether the sample was generated by the counterfeiter or the original maker. Similarly, generator learns from the training set and replicates the sample. Discriminator like the investigator determines the originality of the sample.

Generator G takes input from random noise z and generates data that has a similar distribution as training data x. Discriminator D takes x and the resultant image from G. It determines the probability of a resultant image belonging to training set rather than generated by G. G tries to increase the error rate of Discriminator D and in turn D tries to correctly assign labels for both training samples and generated samples. Networks G and D compete with each other like min-max players with value function V(G, D):

$$\min_G \max_D V(D, G) = E_{x \sim pdata(x)}[\log D(x)] + E_{z \sim pz(z)}[\log(1-D(G(z)))] \quad (10)$$

3.2.2 Stack-GAN

The architecture proposed by Zhang, Han, 2017 [**1**] has proved to generate high-resolution images up to 256x256. The model is decomposed into two-stages, each stage consisting of a generator and a discriminator. Stage-I GAN draws the outline of the object in the scene and applies basic colors to an image producing a low-resolution image. Stage II GAN corrects defects in the Stage-I generated image, provides finishing to the outline of the objects in the image. It refines the quality yielding an image with much higher resolution as compared to the other approaches mentioned in [**4**,**8**,**12**,**13**,**14**].

Moreover, a novel approach was introduced known as conditioning augmentation producing another conditioning variable ĉ as a solution to discontinuity in the latent data manifold due to the constrained amount of data. ĉ is randomly sampled from the Gaussian Distribution $N(\mu(\varphi_t), \Sigma(\varphi_t))$. Mean and diagonal covariance matrix are the functions of text embedding φ_t. This leads to more diversified text-image pairs inducing robustness into the algorithm. To avoid the problem of high variance, regularization is applied with Kullback-Leibler divergence.

3.2.3 Stage-I GAN

As mentioned in earlier related work [**4**,**8**,**12**,**13**,**14**], their model directly yielded image conditioned on the text description. Input t is mapped into vector representation producing text embedding φ_t using skip-thought vector [**15**]. Vector is then fed for conditioning augmentation to obtain more training samples which help to achieve the multimodal objective. The conditioning variable $\hat{c}_0$ is sampled from the uniform Gaussian distribution $N(\mu(\varphi_t), \Sigma(\varphi_t))$ where μ and Σ are the functions of text embedding.

Stage-I Generator G_0: The dimension conditioning variable $\hat{c}_0$ is concatenated with noise vector. The resulting vector is then fed into series of up-sampling blocks generating an image. Stage-I Discriminator D_0: G_0 generated image and the real image are passed through generator comprising down-sampling blocks. The text embedding φ_t undergoes compression following spatial replication and the image tensor is concatenated with text tensor. The resultant tensor is fed into a 1x1 convolutional layer and then to a fully connected layer to output the probability of the sample belonging to the training set rather than generated from the generator. Stage-I GAN G_0 and D_0 are trained using the Eq. (**11**) and Eq. (**12**) where L_{D0} is maximized and L_{G0} minimized respectively.

$$L_{D0} = E_{(I0,t)\sim pdata}[\log D_0(I0, \varphi_t)]+E_{z\sim pz,t\sim pdata}[\log(1-D_0(G_0(z,\hat{c}_0),\varphi_t))] \quad (11)$$

$$L_{G0} = E_{z\sim pz,t\sim pdata}[\log(1-D_0(G_0(z,\hat{c}_0),\varphi_t))]+\lambda(D_{KL}(N(\mu(\varphi_t),\Sigma(\varphi_t))||N(0,I)) \quad (12)$$

Where I_0: real image, t: input sentence, φ_t: text embedding, I_0 and t are sampled from data distribution p_{data}, z: Noise vector sample from Gaussian distribution p_z, λ: Regularization parameter.

3.2.4 Stage-II GAN

Another set of Discriminator and Generator are used for completing the details of the object previously ignored as well as correcting the defects of the image generated by the Stage-I GAN. Again conditioning augmentation is applied on the text embedding which gives another conditioning variable $\hat{c}$ different from $\hat{c}_0$.

Stage-II generator G: Meanwhile, Stage-I result is fed into down-sampling blocks. The tensor $\hat{c}$ is then spatially replicated and concatenated with the image tensor from down-sampling block. To learn multi-modal representations across image and text features, the encoded image and the test features are fed into residual blocks. Up-sampling block (acts as a decoder) generate high-resolution images. Stage-II Discriminator D: The computation steps are similar to Stage-I D_0. The stage-II generated image and the real image are fed into down-sampling block. Compression and spatial replication are performed on text embedding which is then fed into networks as similar to stage-I D_0 and generates a decision score. Discriminator D and Generator G are trained by alternatively maximizing LD and minimizing LG in Eq. (13) and Eq. (14) respectively.

$$L_{D0} = E_{(I0,t)\sim pdata}[\log D_0(I0, \varphi_t)]+ E_{z\sim pz,t\sim pdata}[\log(1-D_0(G_0(s_0,\hat{c}),\varphi_t))] \quad (13)$$

$$L_{G0} = E_{z\sim pz,t\sim pdata}[\log(1-D_0(G_0(s_0,\hat{c}),\varphi_t))]+\lambda(D_{KL}(N(\mu(\varphi_t),\Sigma(\varphi_t))||N(0,I)) \quad (14)$$

Where $s_0 = G_0(z,\hat{c}_0)$.

4. MODEL ARCHITECTURE

4.1 Text Embedding:

Input text t, is converted into text embedding using skip-thought vector mechanism. Recurrent Encoder with GRU activation is implemented yielding 4800 dimension vector. Text embedding is denoted by φ_t.

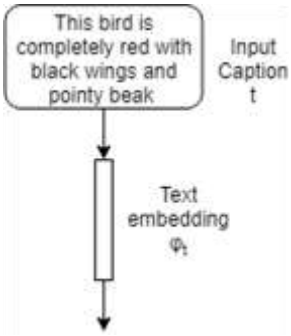

Figure 3: Mapping input text into text embedding.

4.2 Conditioning Augmentation for Stage-I:

Text embedding is fed into a fully connected layer to generate μ_0 and σ_0 (diagonal of the Σ_0 matrix). $\hat{c}_0$, the conditioning variable is sampled from the Gaussian distribution stated as $N(\mu_0(\varphi_t), \Sigma_0(\varphi_t))$. This can be elaborated as σ_0 is element-wise multiplied with distribution $\varepsilon \sim N(0, I)$. The resultant vector is then added to μ_0 which gives conditioning variable $\hat{c}_0$ in Eq. **(15)**

$$\hat{c}_0 = \mu_0 + \sigma_0 \odot \varepsilon \qquad (15)$$

4.3 Stage-I Generator G_0:

Conditioning variable $\hat{c}_0$ is concatenated with noise vector N_z (100) sampled from distribution $z \sim \mathcal{N}(0, I)$. Noise is added to satisfy multi-modal characteristics. The concatenated vector is then fed into series of up-sampling layers yielding an image of $W_0 \times H_0$ (64×64) dimensions.

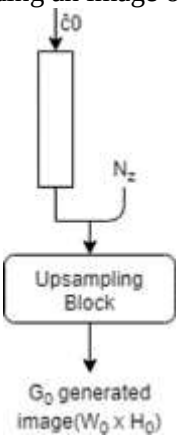

Figure 4. Generator G_0 architecture

4.4 Stage-I Discriminator D_0:

In D_0, the sample is fed into down-sampling block to convert it into $M_d \times M_d$ (4×4) dimension tensor. Moreover, φ_t undergoes compression to achieve N_d (128) dimension the only to replicate to form $M_d \times M_d \times N_d$ (4×4×128). Both the image tensors are concatenated to along with channel dimension and fed into a 1×1 convolutional layer and then to a fully connected layer to produce the probability of the sample.

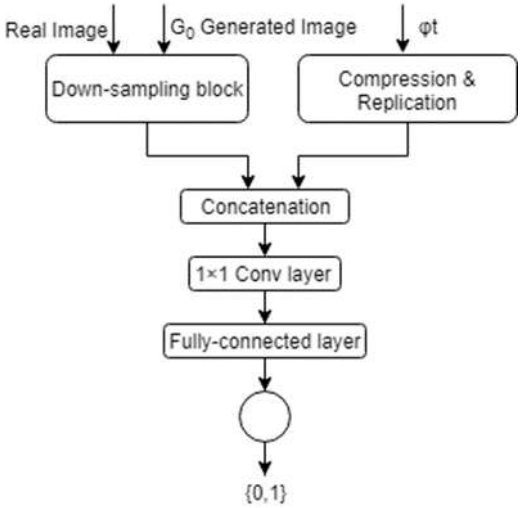

Figure 5. Discriminator D_0 architecture

4.5 Conditioning augmentation for stage-II

Similar to stage-I conditioning augmentation, a conditioning variable ĉ generated of dimension N_g (128) from φ_t.

4.6 Stage-II Generator G:

Stage-II Generator consists of down-sampling blocks, residual blocks and upsampling blocks. The image generated from G_0 network is fed into down-sampling block to convert it into $M_g \times M_g$ (16×16) dimension tensor. The resultant image tensor along with spatially replicated tensor of ĉ are concatenated. The encoded image and text features are fed into two residual blocks [**24**] for generating 128×128 resolution image. The output from the residual blocks is fed into up-sampling blocks to generate an image conditioned on the text description of dimension W×H (128×128).

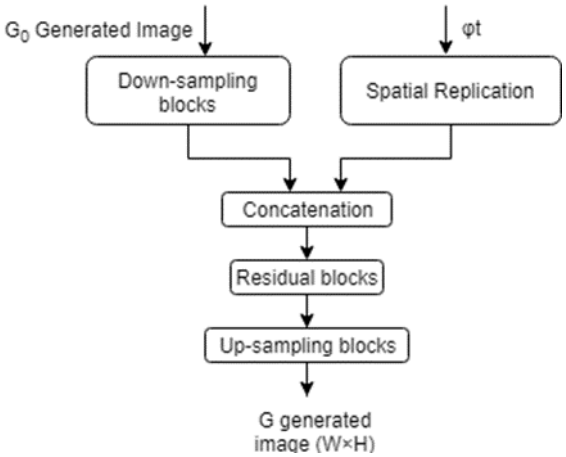

Figure 6. Generator G architecture.

4.7 Stage-II Discriminator D:

The image (W×H) from the Generator is fed into the Discriminator that output the probability of the sample of the training set rather than generated by the Generator. Its architecture and computations are same as stage-I Discriminator D_0 that is it consists of down sampling layers. Compression and replication, 1×1 convolution layer and fully connected layers. Here, number of down sampling blocks (i.e. encoder) is used as the resolution of the image is increased.

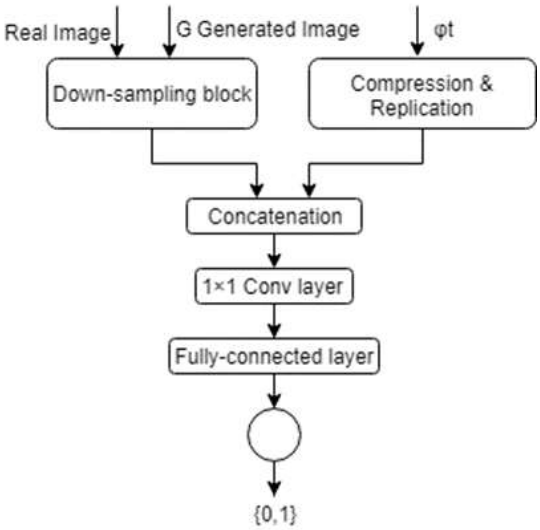

Figure 7. Discriminator D architecture

5. IMPLEMENTATION

5.1 Training Details

Stage-I has been trained for 600 epochs and Stage-II for another 600 epochs. The following values have been considered. The dimensions of each notation mentioned in brackets. Optimization algorithm: Adam (Adaptive Gradient Algorithm). Initial learning rate: 0.0002. Mini-batch size: 64. Activation function after each layer except the last in up-sampling block and residual blocks after every 3×3 stride 1 convolution is Relu. Batch Normalization is applied to the all the layers except the last in both the blocks. This was carried out using the implementation code provided by [**1**], modifying according to the current dependencies.

5.2 Dataset

Implementing for the first time, Caltech-UCSD Birds-200-2011 **[25]** dataset has been used. It contains 200 images of birds all of the different sizes. With 11,788 number of images, 15 part locations, 312 binary attributes, and 1 bounding box are given as annotations per image.

5.3 Image preprocessing

As an image processing step, all the images are cropped to maintain greater-than-0.75 object-image size ratio. Total 11,788 images are divided into 8855 for training and 2933 for testing and loaded into pickle files.

6. EXPERIMENTAL ANALYSIS

Till now, lots of different architectures have been proposed to achieve this objective of mapping input text to visual images. GAN [**19**], DCGAN [**17**], WGAN [**2**], LSGAN [**3**] only to name some. Figure 8. represents visual comparisons between the images generated by three different architectures. Stack-GAN proves to be promising while achieving the objective of generating high-resolution images. According to what is specified in [**9**] mentions many ways to train GANS. To avoid overtraining of the discriminator, feature matching should be implemented. Mini-batch discrimination is to feed the discriminator with multiple examples instead in isolation that prevents collapsing of the generator.

Figure 8. Bird images generated by GAN [19,4], DCGAN [17], Stack GAN[1].

7. CONCLUSION

For implementing stack-GAN, I had only used birds' dataset as the initial dataset. We showed step-by-step how stack-GAN work and visual comparison between other approaches. The limitations of the previous models are effectively overcome by this approach producing better results. As it generates images of high-resolution, the model can be modified to be applied to a more real-time application.

REFERENCES

[1] Zhang, Han, et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks." *IEEE Int. Conf. Comput. Vision (ICCV)*. 2017.

[2] Arjovsky, Martin, Soumith Chintala, and Léon Bottou. "Wasserstein gan." *arXiv preprint arXiv:1701.07875* (2017).

[3] Mao, Xudong, et al. "Least squares generative adversarial networks." *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017.

[4] Reed, Scott, et al. "Generative adversarial text to image synthesis." arXiv preprint arXiv:1605.05396 (2016).

[5] Reed, Scott, et al. "Learning deep representations of fine-grained visual descriptions." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[6] Reed, Scott E., et al. "Learning what and where to draw." *Advances in Neural Information Processing Systems*. 2016.
[7] Reed, Scott, et al. "Generating interpretable images with controllable structure." (2016).
[8] Im, Daniel Jiwoong, et al. "Generating images with recurrent adversarial networks." *arXiv preprint arXiv:1602.05110* (2016).
[9] Salimans, Tim, et al. "Improved techniques for training gans." *Advances in Neural Information Processing Systems*. 2016.
[10] Zhi, Jiale. "PixelBrush: Art Generation from text with GANs."
[11] yf Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
[12] Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "A neural algorithm of artistic style." *arXiv preprint arXiv:1508.06576* (2015).
[13] Mansimov, Elman, et al. "Generating images from captions with attention." *arXiv preprint arXiv:1511.02793* (2015).
[14] Gregor, Karol, et al. "DRAW: A recurrent neural network for image generation." *arXiv preprint arXiv:1502.04623* (2015).
[15] Kiros, Ryan, et al. "Skip-thought vectors." Advances in neural information processing systems. 2015.
[16] Lazaridou, Angeliki, Nghia The Pham, and Marco Baroni. "Combining language and vision with a multimodal skip-gram model." *arXiv preprint arXiv:1501.02598* (2015)
[17] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).
[18] Zhu, Yukun, et al. "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books." *Proceedings of the IEEE international conference on computer vision*. 2015.
[19] Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014
[20] Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv:1406.1078* (2014).
[21] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.
[22] Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." *arXiv preprint arXiv:1412.3555* (2014).
[23] Kalchbrenner, Nal, and Phil Blunsom. "Recurrent continuous translation models." *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 2013
[24] Sun, Shijun. "Methods and systems for block-based residual upsampling." U.S. Patent No. 8,023,569. 20 Sep. 2011.
[25] Wah, Catherine, et al. "The caltech-ucsd birds-200-2011 dataset." (2011).

Authors

Mahima Pandya
A student, currently pursuing Bachelors in Information Technology from CSPIT, Charotar University of Science and Technology, Changa, India. Area of interests include Artificial Intelligence, Machine Learning, Deep learning and statistics

Prof. Sonal Rami
An Assistant Professor at CSPIT, Charotar University of Science and Technology, Changa, India. Area of interests include Data Mining, Operating System, Artificial Intelligence, Design & Analysis of Algorithm and Data Structure