Research Article

Reconstruction of Generative Adversarial Networks in Cross Modal Image Generation with Canonical Polyadic Decomposition

Ruixin Ma, Junying Lou, Peng Li, and Jing Gao

School of Software, Dalian University of Technology, 116024, China

Correspondence should be addressed to Ruixin Ma; maruixin@dlut.edu.cn

Received 8 September 2020; Revised 10 October 2020; Accepted 8 March 2021; Published 9 April 2021

Academic Editor: Xiaojie Wang

Copyright © 2021 Ruixin Ma et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Generating pictures from text is an interesting, classic, and challenging task. Benefited from the development of generative adversarial networks (GAN), the generation quality of this task has been greatly improved. Many excellent cross modal GAN models have been put forward. These models add extensive layers and constraints to get impressive generation pictures. However, complexity and computation of existing cross modal GANs are too high to be deployed in mobile terminal. To solve this problem, this paper designs a compact cross modal GAN based on canonical polyadic decomposition. We replace an original convolution layer with three small convolution layers and use an autoencoder to stabilize and speed up training. The experimental results show that our model achieves 20% times of compression in both parameters and FLOPs without loss of quality on generated images.

1. Introduction

Generating images according to the corresponding text is an important, challenging, and interesting task in computer vision. Compared with text, images are direct and easy to understand. Cross modal image generation attracts many researchers due to its great potential and value in application of computer vision, such as cross modal search, art creation, and image editing. It is conducive to reducing storage space and operating cost. Generating synthetic images from text application in art creation and criminal image calls for fast reaction and compact models. For illustrations for stories and painting for album covers, compact cross modal image generation models can instantly visualize thoughts in the mind by a few descriptive sentences. Text-to-image GANs can activate visualization application so as to promote artistic creation greatly.

In the past few years, most of generative models have applied the Markov chain learning mechanism, Monte Carlo estimation, and sequence data to learn joint distribution. These models involve too much computation and are not suitable for large-scale image generation. The Variational Autoencoder (VAE), Recurrent Neural Network (RNN), and Convolutional Neural Networks (CNN) are used to generate natural pictures according to a conditional distribution [1–3]. These models can generate pictures only by labels or feature information generated by other networks. However, images generated by these models were unreal. Driven by the proposal of the generative model of GANs, images generated from text tasks got a significant development. Reed et al. [4] firstly applied GAN to synthesize impressive and compelling pictures from character level to pixel level. More and more researchers have committed to improving the quality of generated images by adding modules and constraints. Many excellent models have been proposed, such as StackGAN++ [5], AttnGAN [6], and HDGAN [7]. These models can generate high-pixel pictures. But existing text-to-image GANs are so complex that it is hard to deploy them on the mobile end.

Low computation and response in real time are critical for cross modal search and criminal image generating tasks. With the emerging of 5G technology [8–12], the demand for mobile terminal deployment is increasing. However, existing text-to-image GAN models have a large number of
parameters and huge computation for low-end devices within the Internet of things. In order to compress and speed up text-to-image GAN, we propose a compact architecture based on canonical polyadic decomposition.

Rank decomposition has been widely applied in model compression and acceleration. Rank decomposition can represent a complex matrix as multiplication of small submatrices. It means that a few submatrices can be used to reconstruct the weight matrix. These submatrices maintain important properties of the matrix. For cross modal-generated image task, there are too many parameters and high computation in existing models. Therefore, we can use rank decomposition to reduce parameters and computation. There are two methods to apply rank decomposition: decomposing the complex matrix and replacing [13–16] and designing low-rank separable network structures [17, 18]. Canonical polyadic decomposition is an efficient and standard rank decomposition method. It has been effectively applied to compress and accelerate networks [13, 15]. So we use CP decomposition to compress text-to-image GAN.

There are three problems to decompose the complex model. First, implement rank decomposition in the original model because decomposition operations involve high computational cost. Second, text-to-image GANs are more complex than CNN. Because the training process of GAN is using zero-sum two-person game to learn the distribution of real data, the training is unstable and the decomposed model is not easy to converge. The third problem is that cross modal image generation applications have high requirements on the authenticity, clarity, diversity, and resolution of the generated images. It is hard to compress the model as much as possible under the premise of ensuring the image quality.

To solve the first problem, we use CP decomposition to reconstruct text-to-image GAN. It reduces a large number of redundant parameters and decomposition operation cost. Then, we use autoencoders to pretrain to stabilize the decomposed model. For the last problem, we explored a large number of the experiments to find the appropriate rank to guarantee the generated pictures’ quality. Experimental results on representative cross modal image generation datasets show that our scheme can efficiently reduce computation complexity by CP decomposition. More importantly, our model is slightly better than the original model in FID and achieves 20% compression in FLOPs and parameters.

The contributions of this paper can be summarized as follows:

(i) To the best of our knowledge, this is the first paper to use CP decomposition to reconstruct cross modal GAN

(ii) We design a compact text-to-image GAN based on CP decomposition and use autoencoders to pretrain, reducing high computational cost

The rest of the paper is organized as follows: Section 2 presents the preliminaries related to this paper. In Section 3, the reconstruction process of compact cross modal GAN architecture is illustrated. Section 4 evaluates our proposed compact model, and Section 5 summarizes our work.

2. Related Work

The aim of this paper is to reconstruct a compact architecture for text-to-image GAN from scratch. In this section, we present the relevant research in text-to-image GAN and compressing deep neural networks by rank decomposition.

2.1. GAN in Cross Modal Image Generation. The text-to-image task extracts features from human-written descriptions to generate images, which turn low-dimensional and low-rank data into comparatively high-dimensional pictures. It is challenging to use GAN to generate high-resolution images according to text because of GAN’s training instability. Reed et al. [4] first successfully used GAN to generate 64 × 64 high-quality images by modifying DCGAN; then, they put forward GAWWN [19] to generate high-quality 128 × 128 images by using text description and object location as conditions.

StackGAN [20] used stacked conditional GAN to generate 256 × 256 pictures for the first time. In subsequent work, StackGAN++ [5] used tree structure and multiple generators to generate images of different scales. In addition to conditional loss, it introduced unconditional loss and colour regulation. These additional conditions improved stability of the training process and quality of generated images. The third work of the team was to introduce the attention mechanism [6], which synthesized fine-grained details of different sub-areas of images by focusing on the relevant words in the natural language description. It was the first time to indicate that layered attention GAN can automatically select word level conditions to generate different parts of images. TAC-GAN [21] also used condition GAN to synthesize 128 × 128 resolution images with text. Compared with StackGAN [20], its inception score had improved by 7.8%, but resolution was not as high as that of StackGAN. Johnson et al. [22] proposed to use a scene map as an intermediate medium to generate pictures. The model of Johnson et al. [22] solved the outstanding problem of StackGAN which could not deal with complex text. HDGAN [7] designed a pyramid hierarchy structure to solve the problem that images do not match the text in StackGAN [20].

ObjGAN [23] could generate complex scenes according to text. This paper solved the problem of how to make AI understand the relationship between multiple objects in the scene. The generator in ObjGAN could use fine-grained words and object-level information to gradually refine synthetic images. StoryGAN [24] could draw stories based on the sequence condition of the GAN framework. Given a multisentence paragraph, StoryGAN could generate a series of images and each image corresponded to a sentence, completely visualizing the whole story. In order to get vivid generated images, the network had been getting deep and complex. Existing models are hard to be deployed on the mobile end. Therefore, it is necessary to compress these models.

2.2. Rank Decomposition. Rank decomposition is to extract important features of a matrix, such as Singular Value Decomposition (SVD), canonical polyadic decomposition.
(CP decomposition), Tucker decomposition, and tensor train decomposition (TT decomposition). It reduces redundant parameters using small and simple submatrices to represent a complex matrix. Tucker decomposition has a core tensor. Compared with Tucker, CP decomposition is a special Tucker decomposition, which is simpler and more efficient for compressing parameters. TT decomposition is suitable for sequence data and model. Therefore, this paper uses CP decomposition to compress the model.

Rigamonti et al. [25] used SVD and CPD to get a couple of separable filters to approximate an original convolutional layer. It proved validity of separable convolution. Thus, many researchers paid attention to using low-rank decomposition to accelerate network. Some decomposed pretrained networks by tensor decomposition and then replaced by the original network layer [13–16, 26–29]. Some directly designed low-rank separable network structures [17, 18, 30, 31]. Lin et al. [16] decomposed CNN by GSVD and used backpropagation to decrease global reconstruction error. Based on Lin et al. [16] which only performed spatial decomposition, Jaderberg et al. [14] explored both cross channel and spatial decomposition. Then, Denton et al. [13] and Lebedev et al. [15] used CP decomposition to compress and speeded up CNN. Novikov et al. [31] used TT decomposition to compress the model. Based on the separability of convolution, compact networks were designed and trained from scratch [17, 18, 32].

It is feasible and necessary to compress models. There are a few works to compress GANs [33, 34]. Li et al. [33] and Shu et al. [34] used a pretrained network to prune to compress the model. Due to extra high computational cost of decomposing a pretrained network, we design a compact network architecture. It is the first time to use CP decomposition for text-to-image GANs. We train a compact model from scratch so as to reduce cost of decomposition computation. The reconstructed model overcomes unstable training of GANs as the model deepens. Finally, the reconstructed model achieves 20% compression while ensuring the quality of generation.

3. Method

The architecture of our model is shown in Figure 1. Description embedding is concatenated to a noise vector, and then, it is fed forward through the decomposed generator G. Generated images and real images coupled with description embedding are fed to discriminator D. During the training, D learns to distinguish whether pictures are real pictures and pair up with text. Overall, our method has three steps: the first step is to take a convolutional layer and reconstruct it using CP decomposition, the second step is to pretrain a decomposed network layer by layer, and the third step is to select an appropriate learning rate and train the network using backpropagation.

3.1. Canonical Polyadic Decomposition. Canonical polyadic decomposition was proposed by Hitchcock in 1927 [35]. An N-order tensor can be decomposed into a sum of a finite number of rank-one tensors. The finite number of components is the tensor rank R. For example, a second-order kernel tensor \( \mathbf{X} \in \mathbb{R}^{S \times T} \) with rank R is given by the following form:

\[
\mathbf{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r,
\]

where \( \circ \) is the vector outer product, \( \mathbf{a} \in \mathbb{R}^S \), and \( \mathbf{b} \in \mathbb{R}^T \). GAN consisted of a discriminator and a generator generally. The discriminator and generator in GAN-int-clfs are convolutional neural networks. The most time-consuming operation in CNNs is convolution, which maps an input tensor \( \mathbf{X}(i, j, s) \) of size \( X \times Y \times S \) into an output tensor \( \mathbf{Y}(x, y, t) \) of size \( (X - D + 1) \times (Y - D + 1) \times T \). The convolution can be represented as

\[
\mathbf{Y}(x, y, t) = \sum_{i-x}^{x+\delta} \sum_{j-y}^{y+\delta} \sum_{s}^{S} \mathbf{X}(i-x+\delta, j-y+\delta, s) \mathbf{X}(i, j, s),
\]
where $\mathcal{H}(i - x + \delta, j - y + \delta, s, t)$ is a 4D kernel tensor of size $D \times D \times S \times T$ with the first two dimensions corresponding to spatial dimensions, the third dimension corresponding to input channels, and the fourth dimension corresponding to output channels. The $\delta$ denotes half-width $(D - 1)/2$. As shown in Figure 2, the convolution procedure consists of $T$ convolutions of $D \times D \times S$.

In order to compress GAN, we use CP decomposition to reconstruct convolutional layers in a generator. Spatial dimension in the convolutional layer does not need decomposition as it is relatively small (e.g., $3 \times 3$ or $4 \times 4$).

$$\mathcal{H}(i - x + \delta, j - y + \delta, s, t) = \sum_{r=1}^{g} \mathcal{H}_{x,y}^{(1)} \mathcal{H}_{x,y}^{(2)} \mathcal{H}_{x,y}^{(3)}, \quad (3)$$

where $\mathcal{H}_{x,y}^{(1)}$, $\mathcal{H}_{x,y}^{(2)}$, and $\mathcal{H}_{x,y}^{(3)}$ are the three components of sizes $R \times S, R \times D \times D$, and $T \times R$, respectively.

Substituting Equation (3) into Equation (2) and performing simple manipulations give Equation (4). Equation (4) can approximate the convolution (Equation (2)) from the input tensor $X'$ into the output tensor $Y'$.

$$Y(x, y, t) = \sum_{r=1}^{R} \mathcal{H}_{x,y}^{(3)} \sum_{j=1}^{D} \sum_{i=1}^{D} \mathcal{H}_{x}^{(2)} \left( \sum_{r=1}^{R} \mathcal{H}_{x,y}^{(1)} \mathcal{X}(i, j, s) \right) \quad (4)$$

Based on Equation (4), replacing the original convolution with a sequence of three convolutions can reduce convolutional layers’ parameters. For the convenience of understanding, we call these three layers as first, second, and third:

$$G^{(1)}(x, y) = \sum_{i=1}^{S} \mathcal{H}_{x}^{(1)} \mathcal{X}(i, j, s), \quad (5)$$

$$G^{(2)}(x, y) = \sum_{j=1}^{D} \sum_{i=1}^{D} \mathcal{H}_{x}^{(2)} \mathcal{G}^{(1)}(i, j, r) \quad (6)$$

$$Y(x, y, t) = \sum_{r=1}^{R} \mathcal{H}_{x,y}^{(3)} \mathcal{G}^{(2)}(x, y, r), \quad (7)$$

where $G^{(1)}(x, y, r)$ and $G^{(2)}(x, y, r)$ are intermediate tensors of sizes $R \times X \times Y$ and $R \times (X - D + 1) \times (Y - D + 1)$, respectively. The target tensor is computed by three convolutions (see Figure 3).

3.2. Layer-Wise Pretraining. In this paper, we design a new architecture based on canonical polypadic decomposition which decomposes a layer into three layers. Because the network is deeper than the original model and the training process of GAN is unstable, it is necessary to conduct layer by layer pretraining for the model. He et al. [36] proposed that the effect of random initialization was not worse than that of pretraining, but the convergence time was slower. We adopt the autoencoder to pretrain the model layer by layer.

An autoencoder consists of an encoder and decoder. The encoder is to turn input into a hidden spatial representation. It can be represented by a function $h = f(x)$. The decoder is aimed at reconstructing input from a representation of hidden space by function $x' = g(h)$. As a whole, autoencoder can be described by function $g(f(x)) = x'$, where $x'$ is close to original input $x$. Autoencoder learns valuable information from original input by reconstruction.

The training process is that $n$ autoencoders are trained in sequence. After the first autoencoder training, output of the first encoder is taken as the input of the second autoencoder. And third autoencoder takes the output of the second encoder as the input. The structure of the encoder is the same as that of decomposed layers. After training, the encoder replaces a decomposed layer. Taking the training of these three layers first, second, and third as an example in Figure 4, we train the first autoencoder taking the first’s input and replace parameters of the first with those of the encoder in the first autoencoder. Then, after taking the output of the first as the second autoencoder’s input to train the second autoencoder, we replace parameters of second with the encoder’s in the second autoencoder. So is the third’s training.

The training algorithm is shown in Algorithm 1.
3.3. Selection of the Learning Rate. The learning rate determines whether objective function converges to the local minimum and when it converges to the minimum. A suitable learning rate can make objective function converge to the local minimum in a suitable time. Due to the decomposed model getting deeper, the learning rate should be adjusted appropriately. The learning rate determines the step size of weight iteration, so it is a very sensitive parameter. Its influence on the model performance is reflected in two aspects: the first is the initial learning rate and the second is the transformation scheme of the learning rate.

Smith [37] put forward an excellent way to find the initial learning rate which was called the LR range test. The method is very simple and useful to set the learning rate. We use this method to choose the appropriate range of the learning rate. The accuracy or loss curve is obtained by using different learning rates. And we can set two inflection points of increasing and decreasing precision as the upper bound and the lower bound.

Figure 5 is the curve of the increasing learning rate and the curve of loss corresponding to the increase in the number of iterations within CUB200-2011 for the reconstructed architecture. The method LR range test has three hyperparameters: iteration, max learning rate, and min learning rate. In this paper, we set three hyperparameters as 40, 0.001, and 0, respectively. We change the learning rate once every 40 iterations. The learning rate changes according to the following formula:

\[ lr = lr \times 10^{1/20}. \] (8)

Figure 5 shows that loss has a minimum value when the learning rate is around 0.0002 and loss decreases sharply when learning rate is around 0.00017 and 0.00014.
4. Experiment

4.1. Setups

4.1.1. Model. We conduct experiments on a classic and basic model in text-to-image GAN to demonstrate the generality and effectiveness of our method. Reed et al. [4] was the first to successfully apply generative adversarial networks to cross modal image generation which converted a descriptive text into images directly. The colour information obtained by GAN and GAN-cls is correct, but images look unreal. Images generated by GAN-int-cls are more reasonable, so we choose GAN-int-cls.

4.1.2. Dataset. We evaluate our decomposed architecture on the following dataset:

(i) Caltech-UCSD Birds-200-2011. There are 11788 bird images in the data set, including 200 bird subclasses, 5994 images in the training dataset, and 5794 images in the test set. Each image provides image class information, bird bounding box, key part information of the bird, and attribute information of the bird.

4.1.3. Implementation Details. For the reconstructed model, the initial learning rate is 0.00017 for both the generator and discriminator during training. MultistepLR is a learning rate attenuation method in Pytorch. And we adjust learning rate by attenuation coefficient 0.85 in MultistepLR. The batch size on Caltech-UCSD Birds-200-2011 is 64 followed by the setting in the original paper [4] and trained for 1000 epochs. ADAM [38] solver with beta1 0.5 is used for all models. For the sake of comparison, we handle the dataset the same as StackGAN++ [5]. We split CUB into class-disjoint training and test sets and use char-CNN-RNN [19] to obtain text embedding of given description according to images.

4.1.4. Evaluation Metrics. We use an inception score (IS) and Fréchet inception distance (FID) to evaluate generated images quantitatively. IS is commonly used as an evaluation index of GAN. It evaluates the performance of generative models to use entropy and KL divergence by feeding a large number of generated pictures to Inception V3. The large IS score means high quality of the generated images. FID represents the distance between the feature vector of generated images and that of real images. Small FID score means small distance of images distribution, which means that generated images have high definition and rich diversity. We compute IS and FID on 30k samples randomly generated for the test set the same as StackGAN++ [5].

4.2. Results. In the dataset Caltech-UCSD Birds-200-2011, our model is slightly better than the original structure in the Caltech-UCSD Birds-200-2011 in FID and is similar to the original model in IS. The FID and IS of the original model are 66.92 and 2.56 ± 0.03, while those of our model were 65.05 and 2.38 ± 0.01. There are many redundant parameters in the model. Our model got 19% and 23% reduction in parameters and FLOPs, respectively. Our model reduces 6.8 × 10^6 FLOPs and 1.1 × 10^6 parameters compared to the original model (see Table 1).

It is a classic topic to balance the performance and the compression ratio. Trade-off in rank decomposed GAN is more difficult to achieve because GAN is unstable and rank selection is NP-hard in rank decomposition. In rank decomposition, rank represents the compression ratio. As shown in Table 2, we do a large number of experiments to find the balance. We explored different ranks, which are the ratios in Table 2. The ratio is the rank ratio, where 1.0 is the full-rank decomposition and 0.9 means about 0.9 times of original model’s rank. Table 2 shows that with increasing rank, FLOPs and parameters grow. FID is getting smaller and smaller, while IS only has a little change. Maybe it is because FID is more sensitive to model collapse and IS is little unstable. Compared with IS, FID has better robustness. When the rank ratio is 1.0, FID and IS get the best value which is similar to the original model. Although rank is 1.0, the model is compressed by about 20%. It is effective to use CP decomposition to design a compact GAN network. Results on the Caltech-UCSD Birds-200-2011 dataset can be seen in Figure 6. Rank decomposition can reconstruct a model with less parameters from scratch without loss of generating quality.

The reconstructed model proves that there are redundant parameters in the original model at the current optimal effect.
point. The full-rank decomposition result which is little better than the original model in FID may be because the model is small so that it is easier to find the area where the global optimization point is located. In this paper, we also do quantities of comparison experiments to find the global optimization. As shown in Table 3, we adopt three schemes to explore the optimization point. The LR range test proves that 0.00017 and 0.00014 maybe is the better learning rates. We used three transformation schemes of learning rate which is fixed learning rate, CosineAnnealing with Warm-Restarts [39], and MultistepLR. The initial learning rates are around 0.00017 and 0.00014. The result proves that the MultistepLR transformation scheme helps to find the global optimization.

### 5. Conclusion

Cross modal GAN have a wide range of applications in computer vision. However, these models have too high computation and many parameters to be deployed on the mobile end. In this paper, we developed a compact model for text-to-image GAN based on CP decomposition. We replace a complex convolution layer with three small convolutions. Due to unstable training of GAN and uncontrollable generating, we pretrained the decomposed network layer by layer and explored a considerable amount of experiments to select an appropriate learning rate. We demonstrated that cross modal GAN can be reconstructed with less parameters without quality falling. GAN-int-clt is the most classic and basic model of cross modal GAN. CP decomposition is a standard and efficient tensor decomposition method. Our method has proven that CP decomposition is an efficient decomposition method for GAN in the general evaluation index FID and IS. It is applicable for other cross modal GAN to use CP decomposition. In future work, we aim to further study a more compact and stable network architecture of cross modal GAN.
Data Availability
The datasets used in this paper are public datasets which can be accessed through the following website: http://www.vision.caltech.edu/visipedia/CUB-200-2011.html.

Conflicts of Interest
The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments
This work was supported by the Fundamental Research Funds for the Central Universities (No. DUT20LAB136).

References
[1] E. Mansimov, E. Parisotto, J. L. Ba, and R. Salakhutdinov, “Generating images from captions with attention,” in Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 2016.

[2] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, “Pixel recurrent neural networks,” in Proceedings of the 33rd International Conference on International Conference on Machine Learning, vol. 48, pp. 1747–1756, NY, USA, 2016.

[3] A. van den Oord, N. Kalchbrenner, L. Espeholt, A. Graves, and K. Kavukcuoglu, “Conditional image generation with PixelCNN decoders,” Advances in Neural Information Processing Systems, vol. 29, pp. 4790–4798, 2016.

[4] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, “Generative adversarial text to image synthesis,” Proceedings of Machine Learning Research, vol. 48, pp. 1060–1069, 2016.

[5] H. Zhang, T. Xu, H. Li et al., “Stackgan++: realistic image synthesis with stacked generative adversarial networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 8, pp. 1947–1962, 2019.

[6] T. Xu, P. Zhang, Q. Huang et al., “AttnGAN: fine-grained text to image generation with attentional generative adversarial networks,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1316–1324, Salt Lake City, UT, USA, June 2018.

[7] Z. Zhang, Y. Xie, and L. Yang, “Photographic text-to-image synthesis with a hierarchically-nested adversarial network,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6199–6208, Salt Lake City, UT, USA, June 2018.

[8] P. Li, Z. Chen, L. T. Yang, J. Gao, Q. Zhang, and M. J. Deen, “An incremental deep convolutional computation model for feature learning on industrial big data,” IEEE Transactions on Industrial Informatics, vol. 15, no. 3, pp. 1341–1349, 2019.

[9] J. Gao, P. Li, Z. Chen, and J. Zhang, “A survey on deep learning for multimodal data fusion,” Neural Computation, vol. 32, no. 5, pp. 829–864, 2020.

[10] J. Gao, P. Li, and Z. Chen, “A canonical polyadic deep convolutional computation model for big data feature learning in Internet of things,” Future Generation Computer Systems, vol. 99, pp. 508–516, 2019.

[11] P. Li, Z. Chen, L. T. Yang, Q. Zhang, and M. J. Deen, “Deep convolutional computation model for feature learning on big data in Internet of things,” IEEE Transactions on Industrial Informatics, vol. 14, no. 2, pp. 790–798, 2018.

[12] H. Huang, M. Lin, L. T. Yang, and Q. Zhang, “Autonomous power management with double-Q reinforcement learning method,” IEEE Transactions on Industrial Informatics, vol. 16, no. 3, pp. 1938–1946, 2020.

[13] E. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, “Exploiting linear structure within convolutional networks for efficient evaluation,” Advances in Neural Information Processing Systems, vol. 27, pp. 1269–1277, 2014.

[14] M. Jaderberg, A. Vedaldi, and A. Zisserman, “Speeding up convolutional neural networks with low rank expansions,” in Proceedings of the British Machine Vision Conference 2014, Nottingham, UK, 2014.

[15] V. Lebedev, Y. Ganin, M. Rakhabu, I. Oseledets, and V. Lempitsky, “Speeding-up convolutional neural networks using fine-tuned CP-decomposition,” 2014, https://arxiv.org/abs/1412.6553.

[16] S. Lin, R. Ji, X. Guo, and X. Li, “Towards convolutional neural networks compression via global error reconstruction,” in International Joint Conference on Artificial Intelligence, pp. 1753–1759, NY, USA, 2016.

[17] F. Chollet, “Xception: deep learning with depthwise separable convolutions,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1800–1807, 2017.

[18] A. Howard, M. Zhu, B. Chen et al., “Mobilenets: efficient convolutional neural networks for mobile vision applications,” 2017, https://arxiv.org/abs/1704.04861.

[19] S. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee, “Learning what and where to draw,” Advances in Neural Information Processing Systems, vol. 29, pp. 217–225, 2016.

[20] H. Zhang, T. Xu, H. Li et al., “StackGAN: text to photo-realistic image synthesis with stacked generative adversarial networks,” in Proceedings of the IEEE International Conference on Computer Vision, pp. 5907–5915, Venice, Italy, 2017.

[21] A. Dash, J. C. Gamboa, S. Ahmed, M. Liwicki, and M. Z. Afzal, “TAC-GAN - text conditioned auxiliary classifier generative adversarial network,” 2017, https://arxiv.org/abs/1703.06412.

[22] J. Johnson, A. Gupta, and L. Fei-Fei, “Image generation from scene graphs,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1219–1228, 2018.

[23] W. Li, P. Zhang, L. Zhang et al., “Object-driven text-to-image synthesis via adversarial training,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 12174–12182, 2019.

[24] Y. Li, Z. Gan, Y. Shen et al., “StoryGAN: a sequential conditional GAN for story visualization,” in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6329–6338, Long Beach, CA, USA, June 2019.

[25] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua, “Learning separable filters,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2754–2761, 2013.

[26] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, and N. de Freitas, “Predicting parameters in deep learning,” Advances in Neural Information Processing Systems, vol. 26, pp. 2148–2156, 2013.

[27] B. Liu, M. Wang, H. Foroosh, M. Tappen, and M. Penksy, “Sparse convolutional neural networks,” in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 806–814, Boston, MA, USA, June 2015.

[28] T. N. Sainath, B. Kingsbury, S. Sindhwani, E. Arisoy, and B. Ramabhadran, “Low-rank matrix factorization for deep
neural network training with high-dimensional output targets,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6655–6659, Vancouver, Canada, May 2013.

[29] J. Xue, J. Li, and Y. Gong, “Restructuring of deep neural network acoustic models with singular value decomposition,” *Interspeech*, pp. 2365–2369, 2013.

[30] T. Garipov, D. Podoprikhin, A. Novikov, and D. Vetrov, “Ultimate tensorization: compressing convolutional and FC layers alike,” 2016, https://arxiv.org/abs/1611.03214.

[31] A. Novikov, D. Podoprikhin, A. Osokin, and D. Vetrov, “Tensorizing neural networks,” *Neural Information Processing Systems*, vol. 28, pp. 442–450, 2015.

[32] Y. Ioannou, D. Robertson, J. Shotton, and R. Cipolla, “Training CNNs with low-rank filters for efficient image classification,” *Journal of Asian Studies*, vol. 62, no. 3, pp. 952–953, 2015.

[33] M. Li, J. Lin, Y. Ding, Z. Liu, J.-Y. Zhu, and S. Han, “GAN compression: efficient architectures for interactive conditional GANs,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5283–5293, Seattle, WA, USA, June 2020.

[34] H. Shu, Y. Wang, X. Jia et al., “Co-evolutionary compression for unpaired image translation,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3234–3243, Seoul, Korea (South), October 2019.

[35] F. L. Hitchcock, “The expression of a tensor or a polyadic as a sum of products,” *Journal of Mathematics and Physics*, vol. 6, no. 1-4, pp. 164–189, 1927.

[36] K. He, R. Girshick, and P. Dollár, “Rethinking imagenet pre-training,” in *Proceedings of the IEEE international conference on computer vision*, pp. 4918–4927, Seoul, Korea, 2019.

[37] L. N. Smith, “Cyclical learning rates for training neural networks,” in *Proceedings of the 2017 IEEE Winter Conference On Applications Of Computer Vision (WACV)*, pp. 464–472, Santa Rosa, CA, USA, 2017.

[38] D. P. Kingma and J. Ba, “Adam: a method for stochastic optimization,” in *Proceedings of the International Conference On Learning Representations*, San Diego, CA, USA, 2015.

[39] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *7th International Conference on Learning Representations*, New York, LA, USA, 2019.