CS256 - Topics in Artificial Intelligence
Practical Computer Vision using Convolutional Neural Networks

Project Report

Presented to
Professor Mashhour Solh, Ph.D.
Department of Computer Science
San Jose State University

In partial fulfillment
Of the Requirements for the Class
CS256

By
Ajinkya Rajguru, Branden Lopez, Indranil Patil
Rushikesh Padia, Warada Kulkarni

ABSTRACT

Ever increasing number of multimedia applications requires in-depth research in the field of Image Generation, Domain Adaptation and Style Transfer. User reliance on video conferencing applications due to remote working trends has given rise to new innovative opportunities in these fields. The progress that has been made so far using the state of the art algorithms has been immense as compared to the classical image processing approaches. We have implemented and adapted a neural network model to synthesize a talking head upon receiving a stream of input frames based on the solution proposed in [1]. The proposed model can work on live streams, enabling its applications in video conferencing. The proposed solution takes in a source image as an input followed by a live stream of frames to drive the motion and provides us with the desired output. The trained model was optimized using static quantization to make it 1.5 times faster than before.

Key Terms - Image Generation, Domain Adaptation, Style Transfer, Static Quantization

TABLE OF CONTENTS

I. INTRODUCTION 1

II. LITERATURE SURVEY 2

III. METHODOLOGY 3

IV. IMPLEMENTATION DETAILS 4

V. RESULTS 5

VI. DISCUSSION 7

VII. CONCLUSION 8

REFERENCES 9

I. INTRODUCTION

Computer Vision (CV) is an interdisciplinary field aiming to solve computers' understanding of digital images and videos. Specific CV tasks include Motion Transfer, Image Generation, Style Transfer, and Domain Adaptation. Motion transfer tracks an object's motion over time and applies the same actions to a candidate image. Image generation aims to create new realistic images. Finally, Style transfer takes two images, a content image, and a style reference image, then blends them so that the content image has the stylization of the other. In CV, domain adaptation refers to learning features of a source data and applying those features to significantly different datasets. The above tasks often overlap and have many uses in the industry.

In the last few years educational gatherings/meetings have evolved considerably allowing for students and employees to work from any location at any given time. Due to unusual circumstances sometimes it becomes difficult to have a presentable facial appearance or even a suitable background. We have implemented a deep learning model which allows for a user to apply his head motions and facial expressions to a static image instead of worrying about his own appearance and background.

The state-of-art approaches discussed in Section II use 2-D based image translation. These approaches demonstrate excellent result qualities but can only synthesize fixed viewpoint videos which produce less immersive experiences. The 3D key point representation model proposed in [1] allows for better results for data with coarse-grained features like hair, teeth, and accessories. We have implemented a GAN (Generative adversarial network) based deep learning model which makes use of canonical keypoints, head pose and expression estimator, feature extractor and motion field estimator.

The paper is structured as follows. Section II presents the literature review of the classical approach and previous state of the art approached. Section III then describes our methodology and the required details regarding our implementation. Further depth and information regarding the implementation is provided in Section IV. Section V provides the results and the conclusion, discussion and future scope of the project is elaborated in Section VI.

II. LITERATURE SURVEY

Before the advancement of the models in image generation, style transfer, and domain adaptation, a few classical algorithms and state of the approaches were proposed. Mentioned below is our review of these approaches.

A. CLASSICAL APPROACH

P. Burt and E. Adelson [5] proposed a multiresolution spline technique that decomposes the component images - style image and content image - into a set of band-pass filtered images and then the images in each frequency are assembled into the corresponding band-pass mosaic. These band-pass mosaic images are then added to get the desired result. Working of this technique is explained using figure (LRF1).

B. STATE-OF-THE-ART APPROACHES

1. CNN

[6] Referring to the problem of style transfer as texture transfer, introduced A Neural Algorithm of Artistic Style that uses the feature space derived from the VGG network. Image generation as a combination of content of a photograph and style of an artwork is the resultant of the proposed algorithm. For this model, a good tradeoff between the style and content that needs to be maintained is difficult to synthesize.

2. Conditional GANs and CycleGANs

Image to image translation can be considered as a special case of Conditional GANs [7]. However, in this case, instead of using a single label for conditioning on the image, we are using a complete image as an input label [8]. The main drawback of this approach is that it requires “perfect” pairs of images from both domains to train models. To tackle this, CycleGAN introduced an additional loss function called cycle-consistency loss[9]. We simply complete the cycle: we translate from one domain to another and then back again. The reconstructed image must be the same as the original image. The difference between original image and reconstructed image is used to compute the cycle consistency loss.

3. VAE and Coupled GANs

Alternative to the approach mentioned above, [12] proposed an algorithm of unsupervised image-to-image translation that aims at learning the joint distribution of images in separate domains by using images from marginal distributions in individual domains. The framework is based on virtual encoders (VAEs) and coupled generative adversarial networks (GANs), where learning of translation takes place in both directions in one shot. [12]. However, this framework could be unstable due to the saddle point searching problem.

4. Style GAN

[13] introduces StyleGAN, a new generator that learns to separate different aspects of the image. The new architecture perceives an image as a collection of styles where each style controls the effects at a particular scale and adjusting the image style at every layer allows control of image features at multiple scales; allowing the style modification to only affect a certain aspect of the target image. However, the paper does not explicitly mention adopting this architecture for modifying non-synthetic images.

5. JoJoGAN

JoJoGAN [14] is a One-Shot image stylization model built on StyleGan. JoJo GAN utilizes GAN inversion to produce a paired data set from a single styling example. Finetuning StyleGAN on a new dataset [15] and performing layer swapping allows the StyleGAN to learn image to image translation with a relatively small dataset, making One-shot stylization possible.

6. Image Animation -

First Order Motion Model - In [16] a source image S is animated based on the motion of a similar object from a video D. By describing the motion as a set of keypoint displacements and local affine transformations, a GAN combines the appearance of S and the motion of D; following a self-supervised strategy inspired by Monkey-Net.

III. METHODOLOGY

Rather than training a DNN from scratch, transfer learning allows us to reuse trained model weights and apply them to our own needs. We will use a prior checkpoint, trained on the VoxCeleb dataset to view generation of former training, view its shortcomings, and determine if the model will see increased performance if it is overfitted to a single individual. Overfitting to a single individual, or fine tuning, requires data to operate on so we created a diverse video set of a single individual to tune on.

GANs, Detectors, Estimators and Extractors come in many depths and widths, and combining them for video generation takes excessive time, even on high-end computers; due to this we explore model compression techniques. Model compression concerns itself with reducing the computational and memory complexity of models, we aim to use these techniques to explore the model feasibility on edge devices.

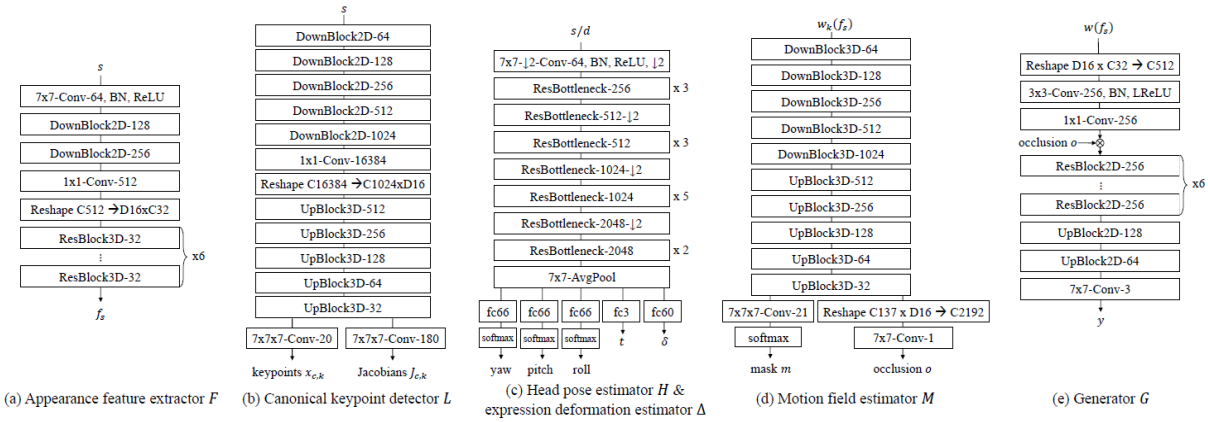
There are two major studies of model compression categories, Pruning and Quantization. Pruning removes non-important weights from a DNN, while Quantization reduces the bit precision of weights, all of which reduce memory requirements and speed up computations. There are many techniques in these bigger two which we discuss later.

On the other hand as model compression improves execution time for the model, we can use the model for generating frames for live video conferencing. OpenCV allows us to capture feed from the user's camera. This feed acts as driving video for the generator.We use

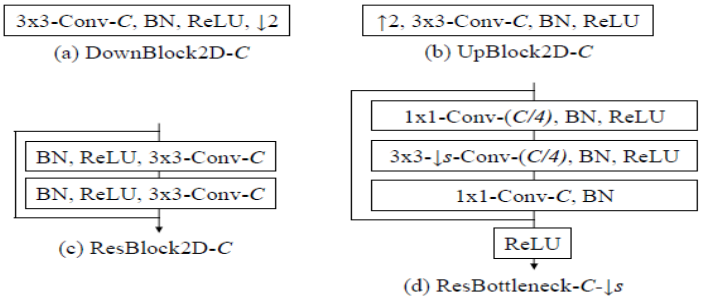
open-cv to calibrate the user's face to a bounding box from which key-points,poses and features can be extracted to drive the source image.Thus we can have photorealistic avatars for video-conferencing by changing source images.

IV. IMPLEMENTATION DETAILS

One-shot Head Motion Transfer has few implementations; the implementation we could find is in the PyTorch framework. The implementation uses five models: Appearance Feature Extractor, Canonical Keypoint Detector, Head Pose / Expression Deformation, Motion Field Estimators, and our Generator. The architecture and components are as follows:



Architecture of Individual Components



Building Blocks of Individuals Components

The model scripts are loaded to Google Colab, where we could utilize 12GB of GPU Ram for high-speed inference. We rewrote and restructured the code to run on the cloud environment despite the reference code being for local machines. Running inference on a 30-second driving video takes over four and a half minutes.

After verifying the checkpoint's ability on inference, we began the process of fine tuning. While freezing layers can significantly speed up the finetuning of generator networks [18], they do not save on memory requirements as the model's weights must be loaded into memory. Running on Colab reinforced the need for a cluster of GPUs as the provided free resource is abysmal for this topic.

Due to the lengthy duration of inference, we first explored pruning. Biological neural networks use efficient sparse connectivity and pruning attempts to mimic this by reducing model parameters with weight replacement of zeros. Unfortunately, how-to guides show that PyTorch pruning does not improve inference times due to its lack of sparse tensor support [19].

Quantization reduces the precision of network data types, and PyTorch supports three different 8-bit integer precision methods. Dynamic quantization will quantize weights during reference and dynamically quantize activation functions during inference, which is great for models whose layers take much time loading weights rather than running computations. Recurrent neural networks typically exhibit this characteristic, so we skip over this. Quantization aware training learns how to represent the quantized weights during training best and shows minor performance drops [20]. However, due to the large memory requirements of training, this method is infeasible. Lastly, Static Quantization performs post-training quantization by passing an input to learn the quantization parameters and then converting to the 8-bit integer precision.

After model compression,the duration of inference reduced significantly. We decided to apply the inference to live feed.Capturing a video stream from a web camera in python using opencv and multi-threading.The module captures and caches image frames from camera to emulate a video feed. These frames are then used to extract the key points,headpose, and motion field is estimated. These features are then used to drive the source image which is previously set. One can change the source image to get various moving avatars. The output generated can be captured using screen capture software like OBS to feed to virtual cam softwares and thus can be effectively used while video calls on conferencing softwares.

V. RESULTS

Using checkpoint from training we ran inference on colab using a 30 second driving video of a man singing and a headshot of a project member. Qualitatively the results are entertaining but they are not of real video quality.

Figure 1: Source Image

Figure 2: Left: Driving Video. Right: Driving motion on Source image.

Initially wanting to finetune the entire model to improve single face generation, we ran into cluster permission issues with the department. However, our limited resources could still run inference on the aggregated model, and we can focus model compression techniques on a subset of the generators.

PyTorch's Static Quantization performed well on Hopenet, the Head pose/expression deformation estimator. Conversion from 32-bit floating-point precision to 8-bit precision reduced the model's size by 3.91 times. Despite consistent and promising memory compression, computational complexity on an x86 CPU is volatile. Running quantized and full-precision inference 100 times and taking the average computation time shows quantized Hopenet being .9-1.5 times faster than full precision inference.

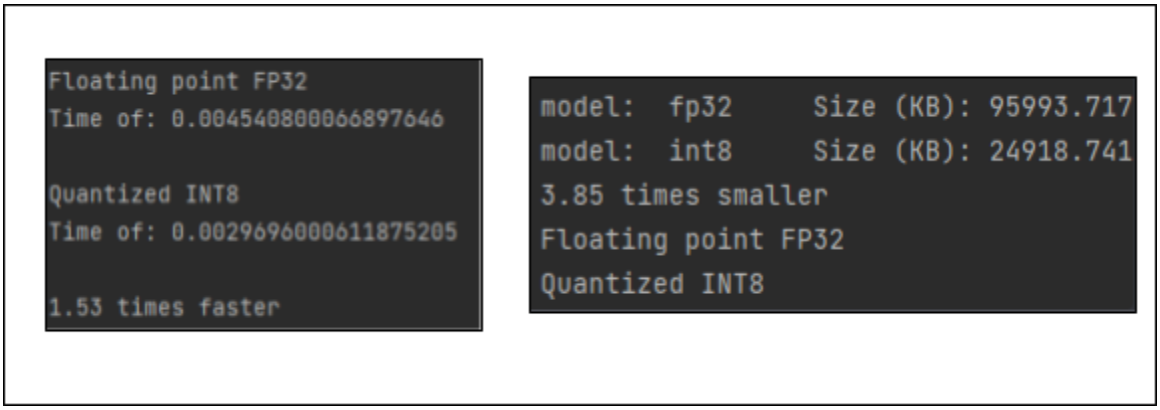


FIgure 3: Static Quantization Results

After optimizing the generator functions and model compression we were able to run inference at a faster rate.Thus we captured live feed to extract features.The redline indicates keypoints and pose for calibrating ideal pose.Blue line shows the changes in the keypoints and pose.

Figure 4: RealTime Keypoint Capture and Pose Estimation from Live Feed

Based on the feature extracted from live feed which acts as driving video we try to instill motion into a random source image.

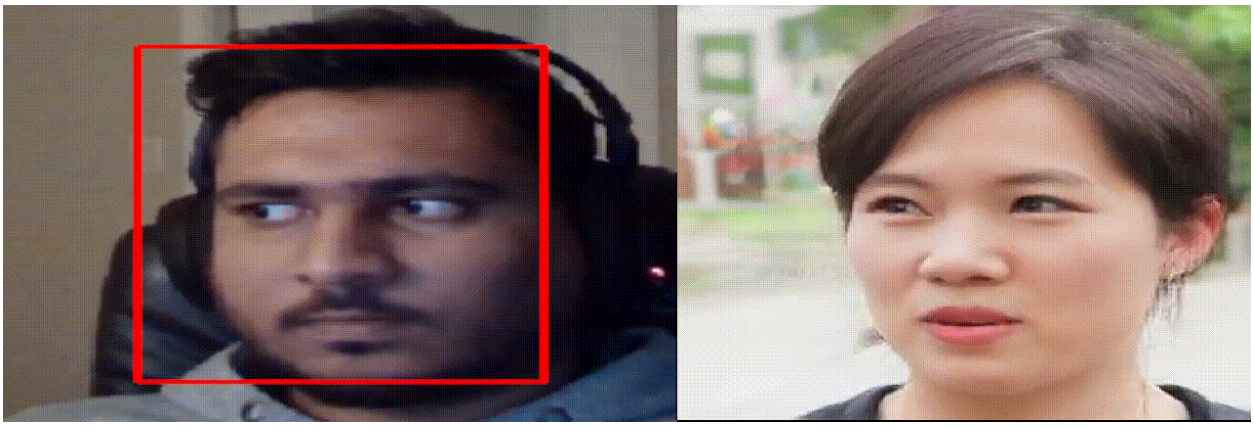


Figure 5: Left: Live Driving Video. Right: Driving motion on Source image.

VI. DISCUSSION

With few resources it's difficult to determine the feasibility of generator networks to reduce video conferencing bandwidths by replacing entire videos with a single still image animated by motion information. While Quantization revealed that model memory can be reduced by almost 4 times the original amount, and inference sped up by 1.5 times, this is still unsatisfactory and does not approach the theoretical inference speed of 2-4 times. Assuming the best possible theoretical advantage, we can change the 2GB of weights required for inference to .5GB and animate a 30 second video in 64 seconds, clearly not good enough for real time video conferencing at very high fps.Fine Tuning on individuals to increase visual results is still explorable but our issues with training have revealed a larger question to real world feasibility, how will individuals find extreme amounts of compute resources to improve still image animation? It's unlikely that such a large model can be finetuned on the edge unless it's a subset of the model that is finetuned individually of the others and that subset greatly improved visual quality.

VII. CONCLUSION

In this project, we explore the feasibility of One-Shot Head Motion Synthesis for real-time video conferencing. While neural networks could reduce the bandwidth needed for video conferencing by sending information on head movement, generator networks still have much work before they can translate movement into realistic videos in real-time. Therefore, future exploration should focus on finding efficient architecture for the models without sacrificing generation; then, a generator model can be extended to real-time video conferencing applications.

REFERENCES

[1] Wang, T.-C., Mallya, A., & Liu, M.-Y. (2021). “One-Shot Free-View neural talking-head synthesis for video conferencing”. CVPR.

[2] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. “VoxCeleb2: deep speaker recognition”. In INTERSPEECH, 2018.

[3] Martin Heusel, et al, “GANs trained by a two time-scale update rule converge to a local nash equilibrium”. In NeurIPS, 2017 References

[4] Davis E. King. “Dlib-ml: A machine learning toolkit”. JMLR, 2009

[5] P. Burt and E. Adelson, "A multiresolution spline with application to image mosaics," ACM Transactions on Graphics, vol. 2, (4), pp. 217-236, 1983. . DOI: 10.1145/245.247.

[6] L. A. Gatys, A. S. Ecker and M. Bethge, "Image style transfer using convolutional neural networks," in 2016. doi: 10.1109/CVPR.2016.265.

[7] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” arXiv preprint arXiv:1411.1784, 2014.

[8] P. Isola, et al, “Image-to-image translation with conditional adversarial networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125–1134.

[9] Langr, J., & Bok, V. (2019). “GANs in Action: deep learning with generative adversarial networks”. Manning. https://books.google.com/books?id=HojvugEACAAJ

[10] Zhu, J.-Y., et al, “Unpaired image-to-image translation using cycle-consistent adversarial networks”. Computer Vision (ICCV), 2017 IEEE International Conference On.

[11] R. Xu et al, "face transfer with generative adversarial network," 2017.

[12] M. Liu, T. Breuel and J. Kautz, "Unsupervised image-to-image translation networks," 2017.

[13] T. Karras, S. Laine, and T. Aila, “A Style-Based generator architecture for generative adversarial networks,” presented at the - 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 4396–4405, doi: 10.1109/CVPR.2019.00453.

[14] Chong. Min, and Forsyth. D.A, “JoJoGAN: one shot face stylization”, 2021 doi: arXiv:2112.11641

[15] Pinkney, J.N., Adler, D.: “Resolution dependent gan interpolation for controllable image synthesis between domains”. arXiv preprint arXiv:2010.05334 (2020).

[16] Siarohin, Aliaksandr, et al. "First order motion model for image animation." Advances in Neural Information Processing Systems 32, 2019

[17] T.-C. Wang, A. Mallya, and M.-Y. Liu, “One-Shot Free-View neural talking-head synthesis for video conferencing,” presented at the - 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 10034–10044, doi: 10.1109/CVPR46437.2021.00991.

[18] Mo S, Cho M, Shin J. Freeze the discriminator: a simple baseline for fine-tuning gans. arXiv preprint arXiv:2002.10964. 2020 Feb 25. https://arxiv.org/pdf/2002.10964.pdf Code: https://github.com/sangwoomo/FreezeD

[19] Bilogur. Aleksey, “A developer-friendly guide to model pruning in PyTorch”, spell.ml, A developer-friendly guide to model pruning in PyTorch (spell.ml) (accessed May 2022)

[20] Nair. Dinesh, Solh. Mashhour, Class Lecture, “Computer Vision and Machine Learning at The Edge”, CS-256, San Jose State University, San Jose, SJSU, April 2022.