FIANCEE: Faster Inference of Adversarial Networks via Conditional Early Exits

Karpikova Polina [*1,2], Radionova Ekaterina [*1], Yaschenko Anastasia [*1,2], Spiridonov Andrei [*1]
Kostyushko Leonid [3], Fabbricatore Riccardo [1], Ivakhnenko Aleksei [1†]

[1]Samsung AI Center - Moscow, Russia   [2]Higher School of Economics - Moscow, Russia
[3]Lomonosov Moscow State University - Russia

Abstract

Generative DNNs are a powerful tool for image synthesis, but they are limited by their computational load. On the other hand, given a trained model and a task, e.g. faces generation within a range of characteristics, the output image quality will be unevenly distributed among images with different characteristics. It follows, that we might restrain the model's complexity on some instances, maintaining a high quality. We propose a method for diminishing computations by adding so-called early exit branches to the original architecture, and dynamically switching the computational path depending on how difficult it will be to render the output. We apply our method on two different SOTA models performing generative tasks: generation from a semantic map, and cross-reenactment of face expressions; showing it is able to output images with custom lower-quality thresholds. For a threshold of LPIPS $\leq$ 0.1, we diminish their computations by up to a half. This is especially relevant for real-time applications such as synthesis of faces, when quality loss needs to be contained, but most of the inputs need fewer computations than the complex instances.

1. Introduction

Image synthesis by generative adversarial networks (GANs) received great attention in the last years [64, 71], its applications span from image-to-image translation [35] to text-to-image rendering [21], neural head avatars generation [17] and many more. However, this approach suffers from heavy computational burdens when challenged with producing photo-realistic images. Our work stems from the observation that deep neural networks (DNNs) output images with different but consistent quality when conditioned on certain parameters. Since their expressivity is uneven within the set of possibly generated images, it follows that for some examples, a simpler DNN may suffice in generat-

ing an output with the required quality.

On the other hand, approaches aimed at easing the heavy computational load of DNNs have been applied with great results, significantly decreasing redundant computations [2, 13]. While strategies such as pruning [46, 53, 65] or knowledge distillation [4, 9, 26] generate a DNN with fewer parameters, early exit (EE) [42, 78] is a setup that allows for dynamic variation of the computational burden, and therefore presents itself as an ideal candidate for an image generation strategy aimed at outputting pictures of consistent quality, while avoiding excessive computation due to their irregular rendering difficulty.

Despite this, implementing EE strategies has remained out of the scope of studies on generative models. This is perhaps due to the fact that EE processes logits of intermediate layers, thus restricting their field of application to tasks where the latter are meaningful (*e.g.* in classification), while excluding pipelines in which a meaningful output is given only at the last layer (*e.g.* generative convolutional networks).

We propose a method that employs an EE strategy for image synthesis, dynamically routing the computational flow towards the needed exit in accordance to pictures' complexity, therefore reducing computational redundancy while maintaining consistent quality. To accomplish this, we employ three main elements, which constitute the novel contributions of our work.

First, we attach *exit branches* to the original DNN (referred as the backbone), as portrayed in Fig. 1. These branches are built of lightweight version of the modules constituting the backbone architecture, their complexity can be tuned in accordance with the desired quality-cost relation. Their depth (*i.e.* number of modules) varies in accordance to the number of backbone modules left after the point they get attached to. In this way, intermediate backbone logits are fairly processed.

In second place, we make use of a small *database* of features, from which guiding examples are selected and used to condition image generation by concatenating them to the input of each branch. These features are obtained by process-

---

*These authors contributed equally to this work
†aleksei.ivakhnenko@mail.ru

ing a selection of images by the first layers of the backbone. Its presence yields a quality gain for earlier exits, at the expense of a small amount of memory and computations, thus harmonizing exits' output quality. This is extremely handy for settings where real-time rendering is needed and guiding examples can be readily provided, such as neural avatar generation.

Lastly, the third component of our workflow is a *predictor*, namely a DNN trained on the outputs of our branches, and capable of indicating the exit needed for outputting an image of a given quality. This element is fundamental for ensuring a consistent lower-quality threshold, as we will see.

Our method is applicable to already trained models, but requires additional training for the newly introduced components. We report its application to two distinct tasks of the image synthesis family, namely generation from a semantic map, and cross-reenactment of face expressions. Our main result may be summarized in this way: the method is easily applicable to already existing and trained generative models, it is capable of outputting images with custom lower-quality threshold by routing easier images to shorter computational paths, and the mean gain in terms of saved computations per quality loss is, respectively, 1.2×10^3, and 1.3×10^3 GFLOPs/LPIPS for the two applications.

2. Related work

2.1. Conditional generative adversarial networks

Generative adversarial networks (GANs) are a class of generative frameworks based on the competition between two neural networks, namely a *generator* and a *discriminator* [23–25]. While the latter performs a classification task (decides whether a generated image is real or not), the former synthesises an image from a target distribution.

Conditional GANs are a variation of the original framework [52]. Their architecture allows for the input of additional information, which is used to restrict the target space according to it. In this way, the network may be conditioned, for instance, by mask [35], label [55], or text [60].

2.2. Neural head avatars

Recent years have seen the rise of neural head avatars as a practical method for creating head models. They allow to reenact a face with given expression and pose. Such models could be divided into two groups – the ones with latent geometry [5, 15, 17, 80, 81] and those with 3d prior, e.g. head mesh [16, 20, 28, 43, 43, 48, 84]. Additionally, there is a set of papers, targeting the whole human body, including the head and face, which could be divided by input data requirements. Some of them take only few images [1], others require a video [7, 22, 31, 33, 40, 41, 58, 79, 85]. In this work we refer to [17], as the state-of-the-art method for one-shot,

high-resolution neural head reenactment.

2.3. Early exits

Early exits are a computational-saving strategy employed mainly in classification tasks [45, 62]. They are characterized by the addition of outputs to the DNN, from which an approximation of the final result can be obtained at a lower computational cost. They were rediscovered through the years as a standalone approach, despite being natively implemented in architectures such as Inception [67] as a countermeasure to overfitting. Seldomly this approach has also been called cascade learning [47, 51, 73], adaptive neural network [3] or simply branching [61]. Proposed implementations differ on three design choices: exits' architecture, *i.e.* what type of layers to use for processing the backbone's logits; where to append exits in order to spread evenly computations among them; and how to choose the computational path. The latter issue is often solved by implementing a confidence mechanism and selecting a single exit [42, 68, 86] or reusing predictions for further computations [76, 78]. To a lesser extent, learnable exit policies have been proposed as well [8, 12, 61].

2.4. Predictor

Changing computational path on a per-input basis has been proposed as a way for efficiently utilizing a single exit during inference [49, 54]. Our approach is inspired by a technique pioneered in the field of neural architecture search: the use of a so-called *predictor* to speed up the performance estimation of a given architecture [6, 75], as well as in natural language processing [18, 77], and has been applied to inference through early exits for resource-constrained edge AI. [14].

2.5. Database use

Early image synthesis methods were based on the retrieval of examples from large image datasets [30, 34, 39, 44]. This is in contrast with contemporary DNN techniques, which rely on a large number of parameters to output photo-realistic images. On the other hand, semi-parametric generation has been proposed in order to exploit strengths of both approaches [59, 66, 69]. In particular, the use of patches, reminiscent of the old methods, seems to achieve great accuracy [27, 50, 70].

Storing a large image database poses a problem when it comes to querying it in order to extract the needed sample. Looking for guiding images, we must employ an algorithm that will quickly find a similar picture or patch. To this end, we borrow from the literature that employs caches [29, 56] and in particular nearest-neighbours search [27, 36, 82], where pretrained models are used as visual feature extractors, and the weights of the image encoders are fixed.
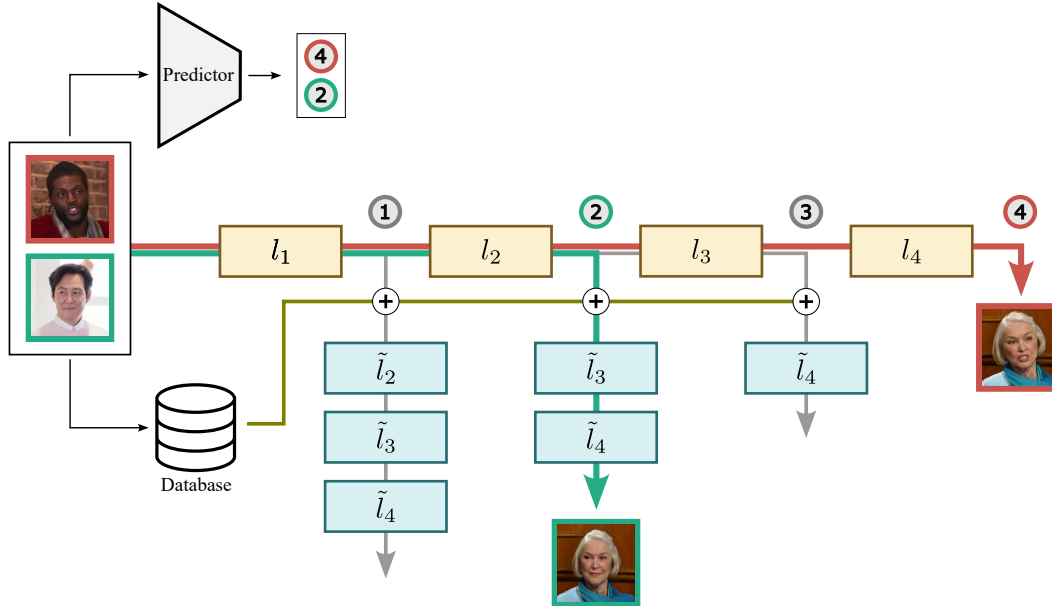
Figure 1. Our pipeline. In this example, the backbone generator is composed of yellow modules l_1 through l_4. We append three branches, thus adding early exits 1 through 3. Each branch has a different depth, and is composed of lightweight modules $\tilde{l}_i$. We show the computational path for two distinct inputs. The top input (red contour) is fed to the predictor (shown at the top), which deems it complex enough to require exit 4 for the given quality threshold. The bottom input (green contour), instead, needs only exit 2 to satisfy quality requirements. For both examples, an auxiliary image is retrieved from the database, in order to guide the synthesis.

3. Our method

Our goal is to implement an early exit mechanism into a GAN model in order to render quickly easier images. To this end, we implement three elements:

3.1. Depth-varying exit branches

As discussed, GANs are composed of two competing DNNs: a generator G and a discriminator D. The former is designed to synthesize arbitrary images when given a low-dimensional random vector of features: $G : z \rightarrow g$. The latter learns to distinguish between the generated images' distribution $p_g = G(p_z)$ and the one from the original examples p_{data}. Their objectives can be summarized in the form of a minimax game:

$$\min_G \max_D \mathcal{L}_{\text{Adv}}(G, D) =$$
$$\mathbb{E}_{x \sim p_{data}} \left[\log D(x) \right] + \mathbb{E}_{z \sim p_z} \left[\log D(G(z)) \right]. \quad (1)$$

By providing conditions c (*e.g.* in the form of labels) to both generator and discriminator, the former can learn to synthesize images from a subspace of p_g: $G(p_z, c) = p_g(c) \subset p_g$.

Any GAN generator is composed of a series of convolutional modules we label l_i. The output of each module, namely $l_k \circ l_{k-1} \circ \cdots \circ l_1(z, c)$ constitutes a candidate for an early exit, but it is not a rendered image. For this reason, we need to process it by a series of additional convolutions, before we can retrieve an image from it. These new

convolutional $\tilde{l}_i$ modules constitute what we call a *branch*. As portrayed in Fig. 1, we append branches to the backbone architecture after each of its modules. Their depth, *i.e.* the number of modules they are made of, varies in accordance with their attach point. For a backbone built out of N modules, after module k, we append a branch of length $N - k$. The branches' modules are less complex, than the backbones', their width, *i.e.* number of channels, is decreased. In this way, at the output of each branch $\tilde{l}_N \circ \cdots \circ \tilde{l}_{k+1} \circ l_k \circ l_{k-1} \circ \cdots \circ l_1(z, c)$, we retrieve an image rendered with a lesser number of computations than at the backbone's output. Each branch is trained by adversarial loss with copies of the backbone original discriminator.

3.2. Exit predictor

During the inference phase, having a set of trained branches, each image can be synthesised through a different exit. Given a quality threshold, we want to be able to select only the branch that will achieve it performing the least possible calculations. To do this, we employ a neural network we call *predictor* P, constituted by convolutional and fully connected layers (see supplementary material for details on its architecture and training). We train our predictor by supervised learning, using the backbone inputs (z, c) as training examples, and LPIPS scores S for images generated by branches as labels.

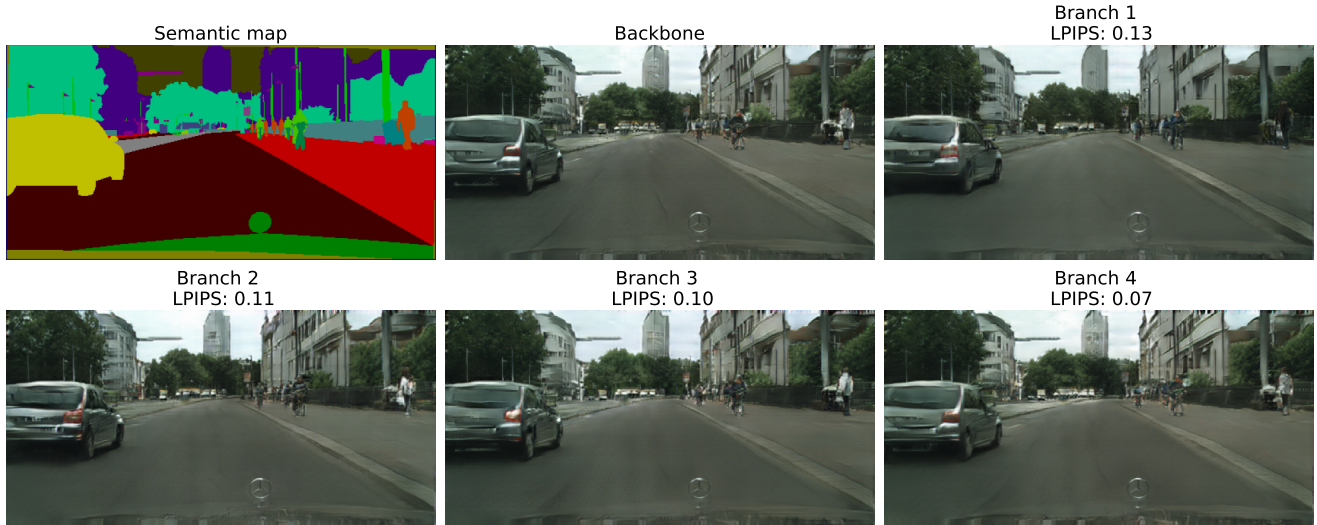$$\mathcal{L}_{pred}(z, c; S) = \|P(z, c) - S\|^2. \quad (2)$$

Figure 2. Examples of branches' outputs for the OASIS pipeline. The input consists of a semantic map and a 3D noise.

Once trained, by feeding an input to the trained predictor, we can quickly get an estimation of each branch's output quality, and thus use this information to route the computational flow toward the exit which performs the least computations, while upholding the threshold.

3.3. Database

To further improve synthesis quality, we shift from a purely parametric method to a semi-parametric, in which the generating process is guided by patches fetched from a relatively small database. This ensures an increase in quality more prominent in earlier exits, which are the fastest, but suffer the most from the quality decrease due to their lower number of parameters. By adding a moderate amount of memory and computations, we achieve better results, harmonizing the output quality of different branches.

In the database, we store a collection of key-value pairs. Keys are given by applying to the images all the trained layers of the backbone prior to the first branch, and cutting the obtained features into non-overlapping patches. Values are obtained by applying the trained layers of the backbone up to its middle, and cutting the resulting features into patches. During inference, we process each input trough the backbone, up to the layer prior to the first branch. We then take the resulting features, cut them into patches, and for each patch we search the database for the closest key. Once we retrieve the values corresponding to all patches, we glue them together and concatenate the obtained features to the input of each branch.

3.4. Computational saving metric

To quantify the success of our method, we introduce a simple measure of the saved computations. Since we trade

quality for computations, we can use the ratio [saved computations] / [quality loss]. As measure units we will use, respectively, GFLOPs* and LPIPS [83]. For instance, in the cross-reenactment of face expressions, we achieve a mean quality gain of 1.3×10^3 GFLOPs/LPIPS, meaning that lowering the quality threshold by $+0.01$ LPIPS will yield a decrease of 13 GFLOPs.

4. Implementations

Our method can be applied to a multitude of DNNs for different synthesis tasks. To showcase its generality, we apply it to two distinct image synthesis tasks: (1) Outdoors photographs synthesis starting from a semantic label map, using the Cityscapes dataset [11], and taking as backbone the OASIS architecture [63]; (2) Neural head avatars synthesis, starting from a picture that acts as the avatar's target expression and position, and using as backbone the Mega-Portraits architecture [17].

4.1. Landscapes from semantic map

For the implementation of synthesis by semantic map, we used outdoor images with semantic maps from the Cityscapes dataset [11]. We implemented our pipeline taking as backbone the OASIS model [63], which takes as input a semantic map in conjunction with a 3D noise tensor for diversifying outputs. The OASIS generator consists of 6 SPADE ResNet modules [57], which in our definitions constitute the backbone modules $l_i, \ i \in [\![1, 6]\!]$. We appended 4 branches, one after each backbone module l_1 to l_4. The branches' modules $\tilde{l}_i$ were SPADE ResNet modules as well, and their length varied in order to preserve
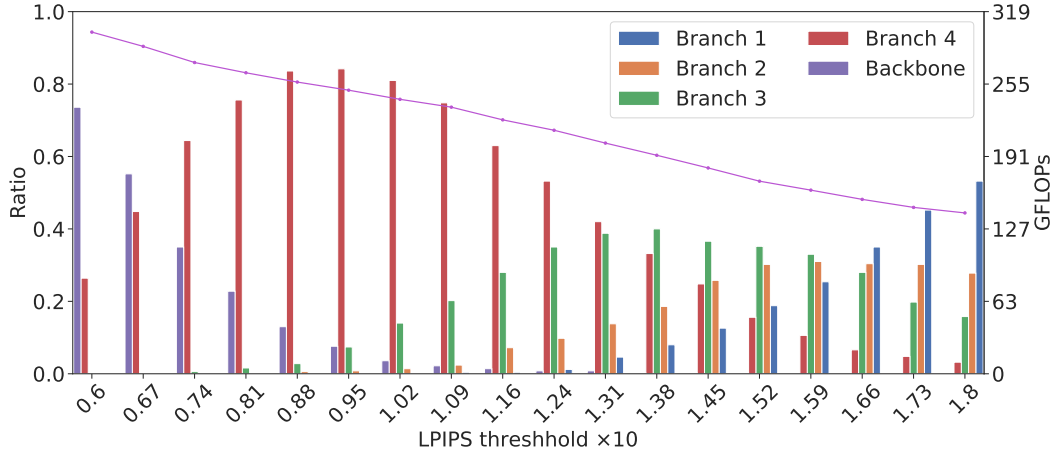
---

*Floating point operations

Figure 3. Distribution of computations among branches of the OASIS backbone for a range of imposed LPIPS thresholds. For each threshold, the predictor routes the computation towards one of five possible exits based on the input's complexity it learned. As quality requirements decrease, the use of the first branches becomes more prominent. All distributions were obtained sampling the same 500 test images and using SF=1/4. Overall GFLOPs for each distribution are shown by the solid line, while their absolute values are shown on the right.
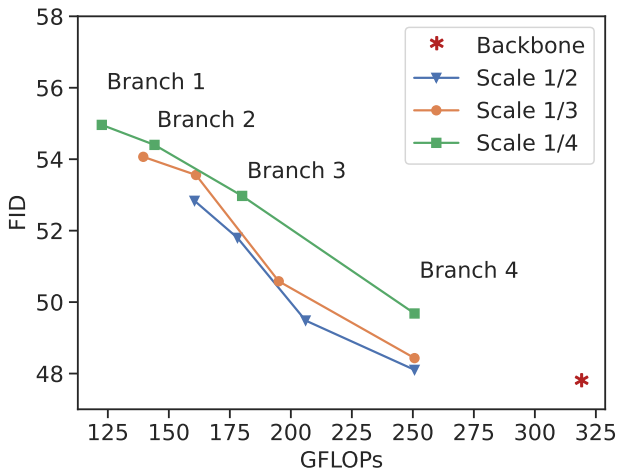

Figure 4. Relation between quality (expressed in FID units) and computations for all branches at different scale factors of the OASIS implementation, with the use of the guiding database.

SF	1	2	3	4	BB
1/2	157	171	193	227	
1/3	137	154	182	227	**319**
1/4	120	138	168	227	

Table 1. Comparison between GFLOPs of all 5 computational routes through branches and the OASIS backbone (BB, rightmost column). Different rows correspond to different scale factors (SF). The scale factor does not equally affect all modules, since we imposed a minimum number of channels equal to 64, after which no further scaling is imposed.

image synthesized by the backbone.

$$\mathcal{L}_{\text{Branch}} = \mathcal{L}_{\text{OASIS}} + \alpha \mathcal{L}_{\text{VGG}} + \beta \mathcal{L}_{\text{LPIPS}}, \qquad (3)$$

where α and β are hyperparameters we chose in order to equalize the losses' contribution. A thorough list of all hyperparameters and training details is given in the supplementary material.

In order to implement the key-value database for guiding image generation (as discussed in Sec. 3.3), we randomly selected 500 semantic maps from the train dataset. For each one of them, we created 100 different inputs using a fixed set of 3D noises. We fed the inputs into the first 2D convolutional layer and the subsequent ResNet module of the backbone. The obtained features were then divided into $8 \times 16 = 128$ non-overlapping patches, in accordance with their resolution, which gave us the keys. The values were extracted by processing the inputs up to the third ResNet module of the backbone and cutting the obtained features into the same patches. The database is populated once at the beginning of the training phase. To decrease the redundancy
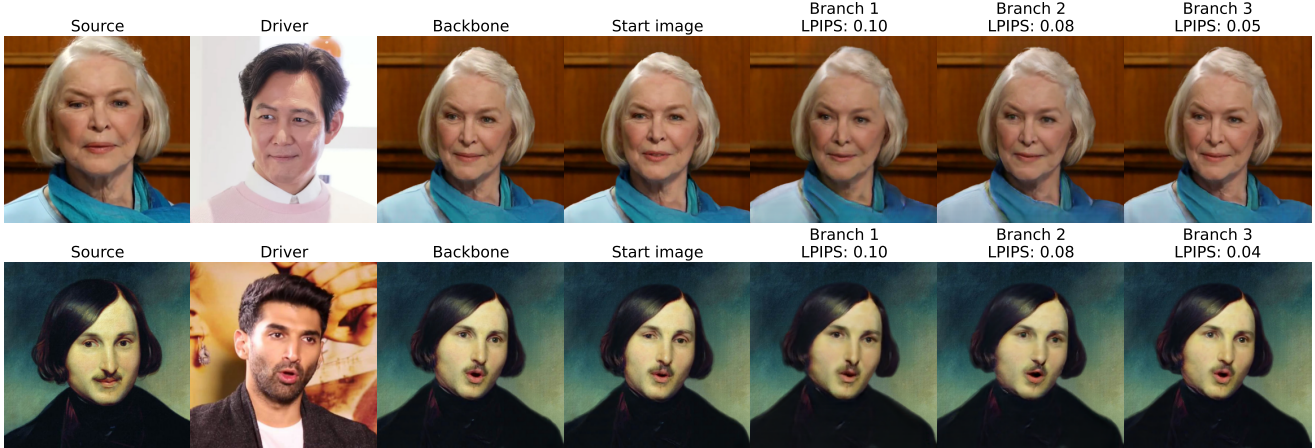
$k + \text{len} = 6 \ \forall k \in [\![1, 4]\!]$, as discussed in Sec. 3.1. They constituted a lightweight variant of the backbone modules since we reduced their width, *i.e.* number of channels, by imposing a *scale factor* (SF) $s = 1/2, \ 1/3, \ 1/4$ in order to reduce computations. A detailed explanation of how we scale down channel numbers is given in the supplementary material. We thus created a total of 5 computational routes for each scale factor, their GFLOPs are listed in Table 1.

We trained each branch by imposing adversarial losses, as in Eq. (1), generated by competing against copies of the OASIS discriminator. Alongside, we also imposed VGG [37] and LPIPS [83] losses using as ground truth the

Figure 5. Examples of branches' outputs for the MegaPortraits pipeline. The model has a source face whose traits must be transferred to the driver's face. The synthesis is guided by the start image, which was selected from a database containing 960 frames with the source appearance, because of its similarity to the driver's pose and expression. LPIPS is measured cutting out the background.

in keys, we applied FPS sampling [19] to them. During the forward phase, after an input was processed through the first 2D convolutional layer and the subsequent ResNet layer, it was divided into 128 patches. Subsequently, the database was searched for the key most similar to each patch with the aid of the FAISS library [38]. All 128 retrieved values were then glued accordingly. We used this composed feature to guide the synthesis process by concatenating it to each branch's input after due resizing performed by a convolution. The resulting distribution of quality among all branches, evaluated by the Fréchet inception distance (FID) [32], is shown in Fig. 4.

Finally, the pipeline comprehending all generating branches and the backbone, together with the database guidance, was used to produce the dataset for training the predictor (as discussed in Sec. 3.2). Since the OASIS input consists of a semantic map and a high-dimensional random noise space, we restricted the training to 100 fixed noise vectors in combination with the Cityscapes train set. In this way we achieved a mean error of 5% on the validation set.

The overall result for the whole pipeline at SF= 1/4 is summarized by Fig. 3. The latter shows the distribution of branches chosen by the predictor at various quality thresholds. One can see how different thresholds affect the exit's choice: while imposing very high quality narrows the spectrum of possible exits, at lower (but nonetheless high) requirements, all additional branches are utilized. Most importantly, the GFLOPs count shows a dramatic decrease of computations when earlier branches are used. By approximating the GFLOPs curve to a constant slope, we can estimate a mean gain factor of 1.2×10^3 GFLOPs/LPIPS.

SF	1	2	3	BB
1/3	65	100	136	
1/6	51	89	135	**213**
1/15	47	85	127	

Table 2. Comparison between GFLOPs of all 4 computational routes through branches and the MegaPortraits backbone (BB, rightmost column). Different rows correspond to different scale factors (SF).

4.2. Neural head avatars

For the neural head avatar implementation, we exploited the VoxCeleb2 dataset [10]. We based ourselves on the MegaPortraits generating method [17] for 512×512 pixels images. This pipeline consists of multiple steps ensuring the transfer of traits from a source face to a driver face. We took as backbone modules l_i, $i \in [\![1, 9]\!]$ its final set of modules comprehending 9 residual blocks, which amount to a total of 213 GFLOPs. We attached 3 branches, one after backbone's block number 2, 4, and 6. Their modules $\tilde{l}_i$ were the same residual blocks, and their respective depth, i.e. number of modules, mirrored that of the remaining path: 8, 6 and 4, thus maintaining $k + \text{len} = 9 \ \forall k \in \{2, 4, 6\}$. To lighten the branches, we imposed three different scale factors to the modules' width, i.e. number of channels. Their overall GFLOPs are listed in Tab. 2.

We trained our branches by imposing adversarial losses, as in Eq. (1), obtained competing with copies of the MegaPortraits discriminator. Alongside, we imposed VGG [37], MS-SSIM [74] and $\mathcal{L}_1$ losses between the branches' and the backbones' synthetic images. Additionally, we used the backbone's intermediate features to impose a feature-matching loss (FM) [72] and retained the original gaze

loss (GL) [17].

$$\mathcal{L}_{\text{Branch}} = \mathcal{L}_{\text{Adv}} + c_1 \mathcal{L}_{\text{VGG}} + c_2 \mathcal{L}_{\text{MS-SSIM}} + c_3 \mathcal{L}_1 + c_4 \mathcal{L}_{\text{FM}} + c_5 \mathcal{L}_{\text{GL}}, \quad (4)$$

where coefficients c_i were chosen to harmonize the losses' effects. A list of all hyperparameters and training details is given in the supplementary material.

We populated our database by pictures of the source face with a plethora of different orientations and expressions. At each iteration, we searched the database for the face most similar to the driver's, *i.e.* the one which orientation and expression we want to obtain. To perform this search, we fed the driver to the first module of MegaPortraits, which extrapolates the angles describing face direction, and a multi-dimensional vector which encodes face expression. We exploited this feature and designed a different key-value search. We employed 3 angles for the encoding of face directions, while the expression space is 512-dimensional. Once we obtained a key characterizing the driver, we looked for the closest one from the images in the database. The retrieved value was then concatenated to the input of each branch module $\tilde{l}_i$ after due resizing. The resulting distribution of quality among branches is shown in Fig. 6.

Finally, we trained the predictor as discussed in Sec. 3.2, on LPIPS scores obtained comparing the branches' with the backbone's output. Afterwards, we were able to impose any quality threshold and the predictor was able to choose the path that satisfied it with the least computation. The overall results for the whole pipeline are summarized by Fig. 7. One can see how lower-quality thresholds can be maintained with a great decrease in GFLOPs due to the use of lighter branches. By approximating the GFLOPs curve to a constant slope, we can estimate a mean gain factor of 1.3×10^3 GFLOPs/LPIPS.

5. Ablation study

Although the image generation is possible without the database of guiding images, we find it essential for ensuring the quality of earlier branches. It can be in fact argued that its implementation harmonizes exits' output quality, by affecting the most the earliest branches, as testified by Fig. 8.

Additionally, the database can be used to amend for the deficiency of the training set. As we will see in the next section, part of the difficulty in rendering is due to a lack of the DNN training, which may very well be inherent to the specific task, as for neural head avatars generation. By providing guiding examples, we somewhat "patch" the holes in the training.

As discussed, our implementation of the dynamical routing relies on the creation of suitable early exit, as well as the use of a predictor. The latter is essential to enforce custom
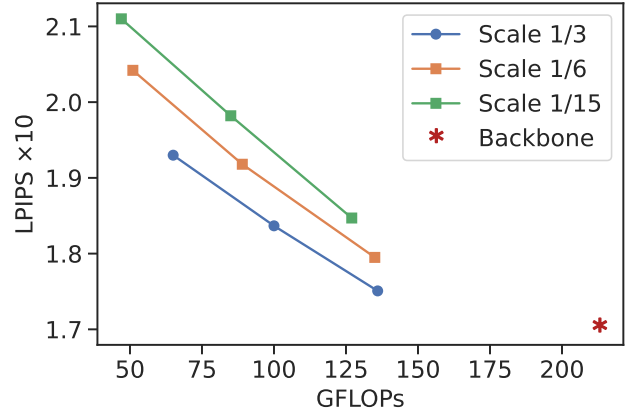

Figure 6. Relation between quality (expressed in LPIPS units) and computations for all branches at different scale factors of the MegaPortraits implementation, with the use of the guiding database.
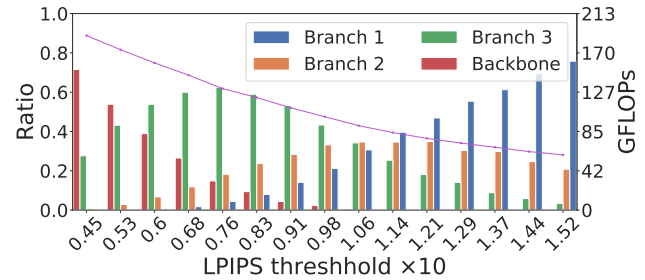

Figure 7. Distribution of computations among branches of the MegaPortraits backbone for a range of imposed LPIPS thresholds. For each threshold, the predictor routes the computation towards one of four possible exits based on the input's complexity it learned. All distributions were obtained by sampling the same 702 test images and using SF=1/15. Overall GFLOPs for each distribution are shown by the solid line, while their absolute values are shown on the right.

quality thresholds, since the use of single exits is will produce only images with fixed quality distributions.

Furthermore, although all branches have a certain mean quality, captured by their FIDs (see Fig. 4 as an example), we can't rely on just a single branch to produce images with consistent quality. The variation in quality of each exit is quite wide and it gets wider in the earliest ones, as portrayed in Fig. 8. The predictor prevents this by choosing a heavier branch when quality can't be provided by a lighter one. The comparison between quality distributions of images obtained from single branches and those obtained by the use of the predictor, set to output a threshold equal to the branches' mean quality, is shown in Fig. 9. We can clearly see how the predictor enforces the threshold by routing difficult images towards the next branches, thus shifting the distribution.
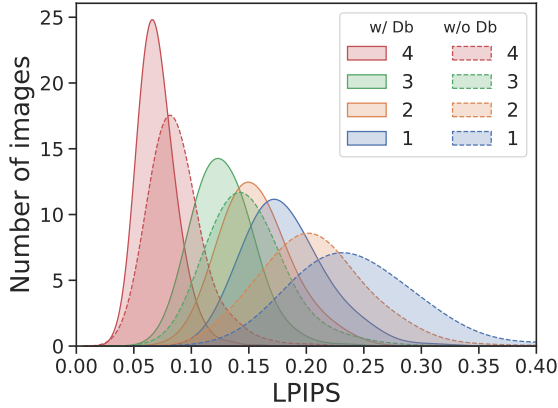
Figure 8. Comparison between quality distributions for the OASIS pipeline with SF=1/4, with the use of the guiding database and without it. LPIPS were obtained by comparison with backbones' images. Curves were drawn sampling 500 images and applying kernel density estimation with bandwidth 0.3.
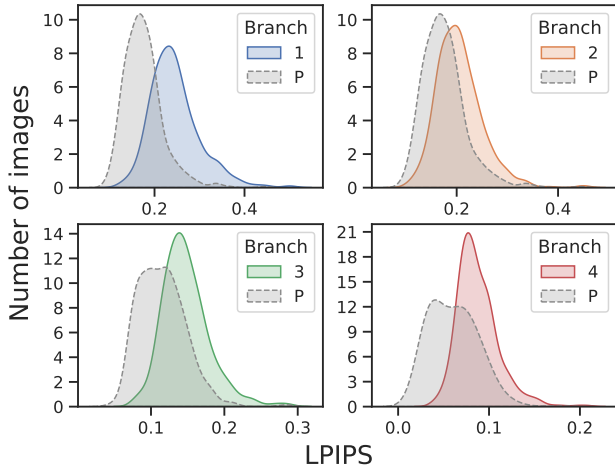

Figure 9. Comparison between quality distributions of single OASIS branches, and quality distributions obtained by use of the predictor (P). The predictor was set to enforce thresholds equal to the branches' mean quality. LPIPS were obtained by comparing images of branches for SF=1/4 with backbones' images. Each distribution is sampled by inputting 500 semantic maps with random noises. The curves are the result of kernel density estimation with bandwidth 0.3.

6. Discussion

Our method is widely applicable, since it can be applied to all models that employ a multi-layer decoder, as illustrated in Fig. 1. The presence of multiple layers is our only requirement, since branches take as input the output of these layers. This includes models that take random noise as input, such as StyleGAN. Such implementation is almost identical to the one for the OASIS model, only without the concatenation of a semantic map to the noises used in the
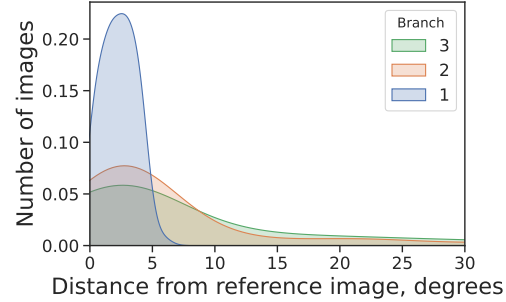

Figure 10. Comparison between number of images routed to different branches in relation to their head rotation. We used the branches for SF=1/15. Distributions were obtained sampling 702 images in total. Curves are the result of kernel density estimation with bandwidth 0.5. The quality threshold was set at 0.09 LPIPS.

database and for the predictor training.

As we stated, not all images are equally difficult to generate. This irregularity lays at the core of our method. A multitude of reasons is responsible for such uneven difficulty distribution. For instance, if we consider the neural head avatar generation problem, one may argue that the DNN is not ideally trained. Some head rotations or expressions may be less present during the training phase, and thus require a heavier model to output images with high quality. We analyzed this problem by comparing images with different head rotations and expressions, and their quality. Specifically, by using our pipeline, we generated 702 head avatars and looked at which branch they were routed by the predictor. By plotting the faces distribution in relation to the angle between them and database images used for guidance, we could clearly see how the rendering difficulty is correlated with this distance. The greater the angle between the two images, the higher the difficulty gets, as reported in Fig. 10.

7. Limitations

Although our method can save a great amount of computations, it has some limitations. One can not apply our pipeline as it is to transformers and other synthesis algorithms that don't comprehend a decoder. There is no single recipe for populating the database. We chose to populate it randomly, but this may actually not be the best choice. Since we need to generate a training dataset for the predictor, we need additional training inputs, thus the size of viable databases is increased. All the branches need additional training, and the memory used for storing the whole pipeline is higher that the one used for the original DNN. We assume the batch size is equal to 1 due to real-time application scenario. In case of batched execution, it will require some orchestration and batch accumulation algorithm in order to infer efficiently, since different images in batch may undergo different computational routes.

References

[1] Thiemo Alldieck, Mihai Zanfir, and Cristian Sminchisescu. Photorealistic monocular 3d reconstruction of humans wearing clothing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1506–1515, June 2022. 2

[2] Ali Alqahtani, Xianghua Xie, and Mark W. Jones. Literature review of deep network compression. *Informatics*, 8(4), 2021. 1

[3] Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. In *International Conference on Machine Learning*, pages 527–536. PMLR, 2017. 2

[4] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 535–541, New York, NY, USA, 2006. Association for Computing Machinery. 1

[5] Egor Burkov, Igor Pasechnik, Artur Grigorev, and Victor Lempitsky. Neural head reenactment with latent pose descriptors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[6] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once for all: Train one network and specialize it for efficient deployment. In *International Conference on Learning Representations*, 2020. 2

[7] Jianchuan Chen, Ying Zhang, Di Kang, Xuefei Zhe, Linchao Bao, Xu Jia, and Huchuan Lu. Animatable neural radiance fields from monocular rgb videos. *arXiv preprint arXiv:2106.13629*, 2021. 2

[8] Xinshi Chen, Hanjun Dai, Yu Li, Xin Gao, and Le Song. Learning to stop while learning to predict. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1520–1530. PMLR, 13–18 Jul 2020. 2

[9] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 1

[10] J. S. Chung, A. Nagrani, and A. Zisserman. Voxceleb2: Deep speaker recognition. In *INTERSPEECH*, 2018. 6

[11] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 4

[12] Xin Dai, Xiangnan Kong, and Tian Guo. Epnet: Learning to exit with flexible multi-branch network. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, CIKM '20, page 235–244, New York, NY, USA, 2020. Association for Computing Machinery. 2

[13] Lei Deng, Guoqi Li, Song Han, Luping Shi, and Yuan Xie. Model compression and hardware acceleration for neural networks: A comprehensive survey. *Proceedings of the IEEE*, 108(4):485–532, 2020. 1

[14] Rongkang Dong, Yuyi Mao, and Jun Zhang. Resource-constrained edge ai with early exit prediction. *Journal of Communications and Information Networks*, 7(2):122–134, 2022. 2

[15] Michail Christos Doukas, Evangelos Ververas, Viktoriia Sharmanska, and Stefanos Zafeiriou. Free-headgan: Neural talking head synthesis with explicit gaze control. *arXiv preprint arXiv:2208.02210*, 2022. 2

[16] Michail Christos Doukas, Stefanos Zafeiriou, and Viktoriia Sharmanska. Headgan: One-shot neural head synthesis and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14398–14407, October 2021. 2

[17] Nikita Drobyshev, Jenya Chelishev, Taras Khakhulin, Aleksei Ivakhnenko, Victor Lempitsky, and Egor Zakharov. Megaportraits: One-shot megapixel neural head avatars. In *Proceedings of the 30th ACM International Conference on Multimedia*. Association for Computing Machinery, 2022. 1, 2, 4, 6, 7

[18] Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-adaptive transformer. In *ICLR 2020-Eighth International Conference on Learning Representations*, pages 1–14, 2020. 2

[19] Y. Eldar, M. Lindenbaum, M. Porat, and Y.Y. Zeevi. The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing*, 6(9):1305–1315, 1997. 6

[20] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *ACM Trans. Graph.*, 40(4), jul 2021. 2

[21] Stanislav Frolov, Tobias Hinz, Federico Raue, Jörn Hees, and Andreas Dengel. Adversarial text-to-image synthesis: A review. *Neural Networks*, 144:187–209, 2021. 1

[22] Anna Frühstück, Krishna Kumar Singh, Eli Shechtman, Niloy J. Mitra, Peter Wonka, and Jingwan Lu. Insetgan for full-body image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7723–7732, June 2022. 2

[23] Liang Gonog and Yimin Zhou. A review: Generative adversarial networks. In *2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 505–510, 2019. 2

[24] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. 2

[25] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Commun. ACM*, 63(11):139–144, oct 2020. 2

[26] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021. 1

[27] Niv Granot, Ben Feinstein, Assaf Shocher, Shai Bagon, and Michal Irani. Drop the gan: In defense of patches nearest neighbors as single image generative models. In *Proceedings*

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 13460–13469, June 2022. 2

[28] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural head avatars from monocular rgb videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18653–18664, June 2022. 2

[29] Edouard Grave, Armand Joulin, and Nicolas Usunier. Improving neural language models with a continuous cache. In *International Conference on Learning Representations*, 2017. 2

[30] James Hays and Alexei A Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), 2007. 2

[31] Tong He, Yuanlu Xu, Shunsuke Saito, Stefano Soatto, and Tony Tung. Arch++: Animation-ready clothed human reconstruction revisited. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11046–11056, October 2021. 2

[32] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 6

[33] Tao Hu, Tao Yu, Zerong Zheng, He Zhang, Yebin Liu, and Matthias Zwicker. Hvtr: Hybrid volumetric-textural rendering for human avatars. *arXiv preprint arXiv:2112.10203*, 2021. 2

[34] Phillip Isola and Ce Liu. Scene collaging: Analysis and synthesis of natural images with semantic layers. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2013. 2

[35] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1, 2

[36] Menglin Jia, Bor-Chun Chen, Zuxuan Wu, Claire Cardie, Serge Belongie, and Ser-Nam Lim. Rethinking nearest neighbors for visual classification. *arXiv preprint arXiv:2112.08459*, 2021. 2

[37] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision – ECCV 2016*, pages 694–711, Cham, 2016. Springer International Publishing. 5, 6

[38] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019. 6

[39] Mica K. Johnson, Kevin Dale, Shai Avidan, Hanspeter Pfister, William T. Freeman, and Wojciech Matusik. Cg2real: Improving the realism of computer generated images using a large collection of photographs. *IEEE Transactions on Visualization and Computer Graphics*, 17(9):1273–1285, 2009. 2

[40] Moritz Kappel, Vladislav Golyanik, Mohamed Elgharib, Jann-Ole Henningson, Hans-Peter Seidel, Susana Castillo, Christian Theobalt, and Marcus Magnor. High-fidelity neural human motion transfer from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1541–1550, June 2021. 2

[41] Moritz Kappel, Vladislav Golyanik, Mohamed Elgharib, Jann-Ole Henningson, Hans-Peter Seidel, Susana Castillo, Christian Theobalt, and Marcus Magnor. High-fidelity neural human motion transfer from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1541–1550, June 2021. 2

[42] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3301–3310. PMLR, Jun 2019. 1, 2

[43] Taras Khakhulin, Vanessa Sklyarova, Victor Lempitsky, and Egor Zakharov. Realistic one-shot mesh-based head avatars. In *European Conference of Computer vision (ECCV)*, 2022. 2

[44] Jean-François Lalonde, Derek Hoiem, Alexei A. Efros, Carsten Rother, John Winn, and Antonio Criminisi. Photo clip art. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3):3, August 2007. 2

[45] Stefanos Laskaridis, Alexandros Kouris, and Nicholas D. Lane. Adaptive inference through early-exit networks: Design, challenges and directions. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*, EMDL'21, page 1–6, New York, NY, USA, 2021. Association for Computing Machinery. 2

[46] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1989. 1

[47] Sam Leroux, Steven Bohez, Elias De Coninck, Tim Verbelen, Bert Vankeirsbilck, Pieter Simoens, and Bart Dhoedt. The cascading neural network: building the internet of smart things. *Knowledge and Information Systems*, 52(3):791–814, 2017. 2

[48] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 2

[49] Xiangjie Li, Chenfei Lou, Zhengping Zhu, Yuchi Chen, Yingtao Shen, Yehan Ma, and An Zou. Predictive exit: Prediction of fine-grained early exits for computation-and energy-efficient inference. *arXiv preprint arXiv:2206.04685*, 2022. 2

[50] Yikang LI, Tao Ma, Yeqi Bai, Nan Duan, Sining Wei, and Xiaogang Wang. Pastegan: A semi-parametric method to generate image from scene graph. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 2

[51] Enrique S. Marquez, Jonathon S. Hare, and Mahesan Niranjan. Deep cascade learning. *IEEE Transactions on Neural*

Networks and Learning Systems, 29(11):5475–5485, 2018. 2

[52] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2

[53] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11256–11264, June 2019. 1

[54] Augustus Odena, Dieterich Lawson, and Christopher Olah. Changing model behavior at test-time using reinforcement learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017. 2

[55] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2642–2651. PMLR, 06–11 Aug 2017. 2

[56] Emin Orhan. A simple cache model for image recognition. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 2

[57] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4

[58] Sida Peng, Zhen Xu, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Animatable implicit neural representations for creating realistic avatars from videos. *arXiv preprint arXiv:2203.08133*, 2022. 2

[59] Xiaojuan Qi, Qifeng Chen, Jiaya Jia, and Vladlen Koltun. Semi-parametric image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[60] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1060–1069, New York, New York, USA, 20–22 Jun 2016. PMLR. 2

[61] Simone Scardapane, Danilo Comminiello, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Differentiable branching in deep networks for fast inference. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4167–4171, 2020. 2

[62] Simone Scardapane, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Why should we add early exits to neural networks? *Cognitive Computation*, 12(5):954–966, 2020. 2

[63] Edgar Schönfeld, Vadim Sushko, Dan Zhang, Juergen Gall, Bernt Schiele, and Anna Khoreva. You only need adversarial supervision for semantic image synthesis. In *International Conference on Learning Representations*, 2021. 4

[64] Pourya Shamsolmoali, Masoumeh Zareapoor, Eric Granger, Huiyu Zhou, Ruili Wang, M. Emre Celebi, and Jie Yang. Image synthesis with adversarial networks: A comprehensive survey and case studies. *Information Fusion*, 72:126–146, 2021. 1

[65] Maying Shen, Pavlo Molchanov, Hongxu Yin, and Jose M. Alvarez. When to prune? a policy towards early structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12247–12256, June 2022. 1

[66] Yupeng Shi, Xiao Liu, Yuxiang Wei, Zhongqin Wu, and Wangmeng Zuo. Retrieval-based spatially adaptive normalization for semantic image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11224–11233, June 2022. 2

[67] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 2

[68] Surat Teerapittayanon, Bradley McDanel, and H.T. Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469, 2016. 2

[69] Yi-Hsuan Tsai, Xiaohui Shen, Zhe Lin, Kalyan Sunkavalli, Xin Lu, and Ming-Hsuan Yang. Deep image harmonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[70] Hung-Yu Tseng, Hsin-Ying Lee, Lu Jiang, Ming-Hsuan Yang, and Weilong Yang. Retrievegan: Image synthesis via differentiable patch retrieval. In *Computer Vision – ECCV 2020*, pages 242–257. Springer International Publishing, 2020. 2

[71] Lei Wang, Wei Chen, Wenjia Yang, Fangming Bi, and Fei Richard Yu. A state-of-the-art review on image synthesis with generative adversarial networks. *IEEE Access*, 8:63514–63537, 2020. 1

[72] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 6

[73] Xin Wang, Yujia Luo, Daniel Crankshaw, Alexey Tumanov, Fisher Yu, and Joseph E Gonzalez. Idk cascades: Fast deep learning by learning not to overthink. *arXiv preprint arXiv:1706.00885*, 2017. 2

[74] Z. Wang, E.P. Simoncelli, and A.C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers*, volume 2, pages 1398–1402, 2003. 6

[75] Wei Wen, Hanxiao Liu, Yiran Chen, Hai Li, Gabriel Bender, and Pieter-Jan Kindermans. Neural predictor for neural architecture search. In *Computer Vision – ECCV 2020*, pages 660–676. Springer International Publishing, 2020. 2

[76] Maciej Woł czyk, Bartosz Wójcik, Klaudia Bał azy, Igor T Podolak, Jacek Tabor, Marek Śmieja, and Tomasz Trzcinski.

Zero time waste: Recycling predictions in early exit neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 2516–2528. Curran Associates, Inc., 2021. 2

[77] Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. Berxit: Early exiting for bert with better fine-tuning and extension to regression. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: Main Volume*, pages 91–104, 2021. 2

[78] Qunliang Xing, Mai Xu, Tianyi Li, and Zhenyu Guan. Early exit or not: Resource-efficient blind quality enhancement for compressed images. In *Computer Vision – ECCV 2020*, pages 275–292. Springer International Publishing, 2020. 1, 2

[79] Jae Shin Yoon, Duygu Ceylan, Tuanfeng Y. Wang, Jingwan Lu, Jimei Yang, Zhixin Shu, and Hyun Soo Park. Learning motion-dependent appearance for high-fidelity rendering of dynamic humans from a single camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3407–3417, June 2022. 2

[80] Egor Zakharov, Aleksei Ivakhnenko, Aliaksandra Shysheya, and Victor Lempitsky. Fast bi-layer neural synthesis of one-shot realistic head avatars. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 524–540. Springer International Publishing, 2020. 2

[81] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[82] Renrui Zhang, Rongyao Fang, Peng Gao, Wei Zhang, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *arXiv preprint arXiv:2111.03930*, 2021. 2

[83] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4, 5

[84] Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C. Bühler, Xu Chen, Michael J. Black, and Otmar Hilliges. I m avatar: Implicit morphable head avatars from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13545–13555, June 2022. 2

[85] Zerong Zheng, Han Huang, Tao Yu, Hongwen Zhang, Yandong Guo, and Yebin Liu. Structured local radiance fields for human avatar modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15893–15903, June 2022. 2

[86] Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. Bert loses patience: Fast and robust inference with early exit. In *Advances in Neural Information Processing Systems*, volume 33, pages 18330–18341. Curran Associates, Inc., 2020. 2