Are Visual Recognition Models Robust to Image Compression?

João Maria Janeiro [1] **Stanislav Frolov** [2 3] **Alaaeldin El-Nouby** [1 4] **Jakob Verbeek** [1]

Abstract

Reducing the data footprint of visual content via image compression is essential to reduce storage requirements, but also to reduce the bandwidth and latency requirements for transmission. In particular, the use of compressed images allows for faster transfer of data, and faster response times for visual recognition in edge devices that rely on cloud-based services. In this paper, we first analyze the impact of image compression using traditional codecs, as well as recent state-of-the-art neural compression approaches, on three visual recognition tasks: image classification, object detection, and semantic segmentation. We consider a wide range of compression levels, ranging from 0.1 to 2 bits-per-pixel (bpp). We find that for all three tasks, the recognition ability is significantly impacted when using strong compression. For example, for segmentation mIoU is reduced from 44.5 to 30.5 mIoU when compressing to 0.1 bpp using the best compression model we evaluated. Second, we test to what extent this performance drop can be ascribed to a loss of relevant information in the compressed image, or to a lack of generalization of visual recognition models to images with compression artefacts. We find that to a large extent the performance loss is due to the latter: by finetuning the recognition models on compressed training images, most of the performance loss is recovered. For example, bringing segmentation accuracy back up to 42 mIoU, i.e. recovering 82% of the original drop in accuracy.

1. Introduction

Mobile devices with high resolution vision sensors, but limited storage and compute capabilities, are ubiquitous: including smartphones, watches, and AR/VR devices. Image compression is critical to facilitate storage of the captured data on-device, and to reduce the required channel bandwidth and latency for remote storage. State-of-the-art recognition models that enable analysis of visual data, rather than just storing it, are currently without exception based on deep learning. They impose heavy memory and compute requirements, despite significant efforts to reduce inference cost, e.g. using efficient architectures (Howard et al., 2017; Iandola et al., 2016; n & Le, 2019), weight compression (Masana et al., 2017; Tai et al., 2016), quantization (Fan et al., 2021; Jacob et al., 2018; Lin et al., 2016), and network pruning (Ghosh et al., 2018; LeCun et al., 1990; Li et al., 2017; Veniat & Denoyer, 2018). The use of state-of-the-art vision models for low-latency applications, therefore, requires transmission of the data to compute servers in compressed format, and recognition models should be robust to artefacts that may be introduced by compression.

Prior works have focused on faster and more efficient processing, by learning vision recognition decoders directly on compressed features (Park & Johnson, 2022; Wiles et al., 2022). Another focus has been on faster transfer of data, through split computation with compressed data (Choi & Bajić, 2018; Nakahara et al., 2021). The aforementioned works require architectural changes to the networks, and novel methods. Moreover, previous work mostly considers a single compression algorithm, JPEG in (Park & Johnson, 2022), HEVC (Sullivan et al., 2012) in (Choi & Bajić, 2018), and VQ-VAE in (Wiles et al., 2022). To the best of our knowledge, the impact of image compression on visual recognition has not been systematically studied.

In this work, we evaluate to what extent state-of-the-art visual recognition models are robust to compression of the input images across three tasks: image classification, object detection and semantic segmentation, on ImageNet (Deng et al., 2009), COCO (Caesar et al., 2018) and ADE20K (Zhou et al., 2017), respectively. We explore both neural compression methods, as well as traditional hand-engineered codecs. We consider bitrates from 2 bits-per-pixel (bpp) down to 0.1 bpp, ranging from high-quality compression to an extreme compression regime where visible artefacts are introduced.

We find that for all tested codecs, image compression leads

[1]Meta AI [2]RPTU Kaiserslautern-Landau [3]German Research Center for Artificial Intelligence (DFKI) [4]Inria. Correspondence to: Jakob Verbeek <jjv@meta.com>, João Maria Janeiro <joao-mariajaneiro1@gmail.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

to a degradation of visual recognition performance, in particular at low bitrates. A-priori, it is not clear what is causing the degradation: compression can lead to a loss of detail which makes the recognition tasks intrinsically harder, or the recognition models do not generalize well to compressed images due to a lack of robustness to the domain shift introduced by the compression artefacts. By finetuning the recognition models on compressed images we can mitigate the domain shift, and test what causes the observed performance degradation. We find that most of the performance loss can be recovered using the finetuned models, suggesting that the performance reduction can be attributed to the models' inability to generalize to images with compression artefacts, rather than the presence of compression artefacts increasing the difficulty of recognition.

To summarize, we make the following contributions:

- We evaluate the impact on image classification, object detection and semantic segmentation accuracy, when compressing images with state-of-the-art traditional as well as learned neural codecs.
- We observe significant degradations in recognition accuracy at low bitrates of 0.1 bpp, and find that this is mostly caused by the inability of recognition models to generalize to images with compression artefacts.
- We show that most of the accuracy loss can be recovered by finetuning recognition models on compressed images, in particular when using neural compression. For detection and segmentation with finetuning, the mAP and mIoU obtained using original images can be approximated up to 0.5 points with images compressed to 0.4 bpp, reducing the image data size by a factor 4 and 12 for segmentation and detection, respectively.

2. Related work

Neural compression methods. Most neural image compression methods follow an autoencoder architecture as a way to achieve a good reconstruction from a small latent representation space, see e.g. (Ballé et al., 2018; Wiles et al., 2022; El-Nouby et al., 2023; Mentzer et al., 2019; Rippel & Bourdev, 2017). An entropy model is employed to estimate the probability distribution over quantized latents, which is in turn used by an entropy coder —typically an arithmetic coder— to compress the latent representation in a lossless manner into a bit stream, see e.g. (MacKay, 2003).

Vision tasks from compressed latent space. Several prior works have explored learning visual recognition models from compressed latent representations (Park & Johnson, 2022; Wiles et al., 2022; Wang et al., 2022). For example, (Park & Johnson, 2022) trains a ViT (Dosovitskiy et al., 2021) directly on JPEG coefficients, and expresses common data augmentations in the same space. They evaluate on ImageNet classification (Deng et al., 2009), and achieve similar performance to the RGB model. On the other hand, (Wang et al., 2022) instead trains a CNN on the frequency-domain features, and assesses its performance in object detection and image classification tasks. For video, (Wiles et al., 2022) uses a VQ-VAE autoencoder (Oord et al., 2017; Razavi et al., 2019) at the frame level, and learns video classification models on the bottleneck representation. This reduces memory and compute requirements, allowing processing of minute to hour long videos.

Split computing with compression. The computation of a model can be divided between the user's device and the cloud. Several works make use of image/feature compression for faster data transfers (Choi & Bajić, 2018; Nakahara et al., 2021; Cohen et al., 2021). In (Choi & Bajić, 2018) an object detection model is trained to compensate for the lossy feature compression artefacts. Transmission of an image compressed at different bitrates until the desired recognition quality is achieved is explored in (Nakahara et al., 2021).

All the aforementioned works focus on a single compression method, and develop new techniques for a single task. Meanwhile, our focus is not to create a new method, but rather to systematically evaluate existing compression methods for several representative recognition tasks.

3. Experimental setup

This section covers the compression methods employed in this study, the recognition tasks used to evaluate their effectiveness, as well as their training and testing setup.

3.1. Image compression codecs

We use four state-of-the-art compression codecs: two traditional compression codecs, BPG (Bellard) and WebP (web), and two neural compression methods based on the hyperprior model (Ballé et al., 2018). In particular, we use the Mean and Scale (M&S) hyperprior model (Minnen et al., 2018), and the Gaussian Mixture Model (GMM) hyperprior (Cheng et al., 2020). M&S combined a mean and scale hyperprior with an autoregressive context model, for better rate-distortion trade-offs. GMM improves over M&S by replacing the Gaussian likelihood model over the latents by a Gaussian mixture model, which better captures the conditional distributions given the hyperlatents. In Fig. 1, we present an image compressed at three different rates by BPG and GMM hyperprior, to illustrate the image quality and artefacts at the bitrates considered in our experiments. We utilize the PIL library (pil) for WebP, Bellard's implementation for BPG (Bellard), and the CompressAI library (Bégaint et al., 2020) library for the neural codecs. To compute the image sizes in bit-per-pixel (bpp), we use the CompressAI

Figure 1. Image compressed at three different bitrates using BPG and GMM hyperprior. Black square provides a zoom of the central area.

library for the neural codecs and WebP, while for BPG we divide the image file size by the number of pixels. For the original images in the datasets, we compute the bpp based on the JPEG filesizes.

3.2. Visual recognition tasks

We consider image classification, object detection and semantic segmentation as representative recognition tasks. For classification and segmentation, we use a Swin-T backbone (Liu et al., 2021), combined with an MLP head for classification, and an UPerNet head (Xiao et al., 2018) for segmentation. For detection, the backbone is a ResNet-50 (He et al., 2016), with a Disentangled Dense Object Detector (DDOD) head (Chen et al., 2021). We use implementations of the MMClassification (Contributors, 2020a), MMDetection (Chen et al., 2019), and MMSegmentation (Contributors, 2020b) libraries. We evaluate the models on ImageNet (Deng et al., 2009) for classification, COCO (Caesar et al., 2018) for detection, and ADE20K (Zhou et al., 2017) for segmentation. For each task we use the standard evaluation metrics: accuracy for classification, mean average precision (mAP) for detection, and mean intersection-over-union (mIoU) for segmentation.

In our experiments we evaluate the public checkpoints released for the different models in the corresponding libraries, which are trained on the original images in the datasets. We experimentally observe that the recognition accuracy of these models deteriorates when evaluated on compressed images. This could be due to a loss of detail when compressing, which makes the recognition tasks intrinsically harder, or because the recognition models lack robustness and do not generalize well to compressed images. To investigate how these factors contribute, we finetune the models using compressed versions of the training images, so that the models adapt to compression artefacts, and the original domain shift in the input data is eliminated.

In practice, we use the same amount of finetuning iterations as were originally used to adapt the pre-trained backbones to the different tasks. For classification, the model is finetuned for 30 epochs, for detection 12 epochs, and for segmentation 160k iterations. We finetune models separately for each compression level.

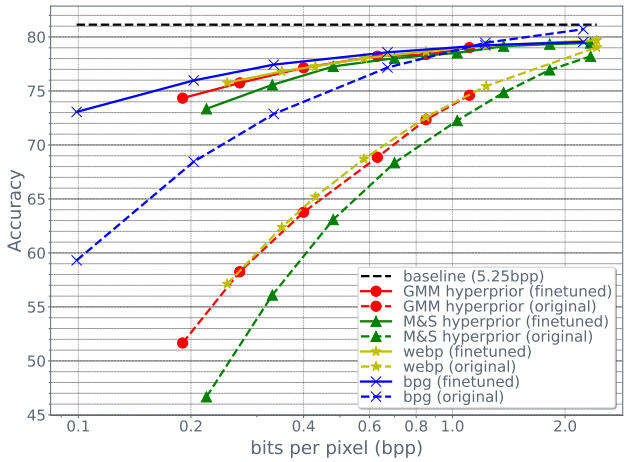
To factor out the influence of additional training, we also finetune the baseline models on the original datasets, for the same amount of additional epochs. We select the best scoring model, original or finetuned, as the baseline. For classification and segmentation, finetuning the original model did not improve accuracy, while for detection finetuning did improve the original model.

4. Experimental results

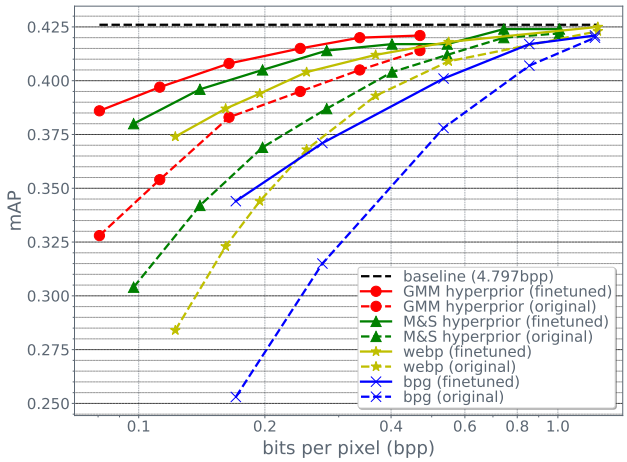
We present our main experimental results in Fig. 2, and discuss and interpret the results below.

Classification. When using the baseline model trained on the original images for classification (dashed curves in Fig. 2a), we found that compressing images with BPG has the least impact on recognition accuracy, followed by WebP and GMM hyperprior which yield comparable impacts. Finetuning the model on compressed images (solid curves) yields a significant improvement in results. For example, improving accuracy from 59.5% to 73% for BPG compression at 0.1 bpp, relative to baseline accuracy of 81% on the original images (5.2 bpp). This shows that, to a large extent, the accuracy drop observed when testing on compressed images, is due to the lack of generalization of the original model to images with compression artefacts. After finetuning, at 1 bpp the accuracy is around 79% for all compression methods; a 3% loss w.r.t. the baseline model while reducing the bitrate by a factor five.

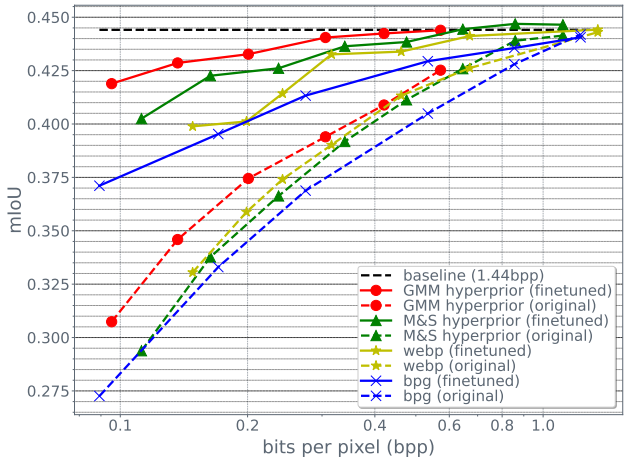
Object detection. Interestingly, the results for object detection on COCO in Fig. 2b, show a different ordering of



(a) Results for image classification on ImageNet.



(b) Results for object detection on COCO.



(c) Results for semantic segmentation on ADE20K.

Figure 2. Visual recognition results with compressed images. The horizontal dashed black line is the baseline result obtained using the original images. Other curves evaluate models trained on original images (original, dashed), and model finetuned using compressed images (finetuned, solid), test images are compressed using WebP, BPG, M&S and GMM hyperprior codecs.

results w.r.t. the different compression codecs. Here, the traditional codecs BPG and WebP lead to bigger drops in accuracy than the neural compression models. In fact, even when finetuning on compressed images the results for BPG (solid blue) are worse than using the original model on images compressed using the neural codecs (dashed green and red). Similar to the classification experiment, the drop in object detection accuracy can to a large extent be recovered by finetuning the model on compressed images. For example, for the neural codecs at 0.1 bpp, the initial drop of 10 points or more in mAP is reduced to under 5 points. At 0.4 bpp, after finetuning the GMM hyperprior model is able to reduce the bit rate by more than a factor 10, while reducing the mAP by only 0.5 (from 42.5 to 42.0) w.r.t. the baseline model on the original images.

Semantic segmentation. For semantic segmentation we observe similar trends as for detection: BPG compression hurts accuracy most, and GMM hyperprior compression has least impact. When compressing images with the GMM hyperprior codec to 0.1 bpp, an mIoU of 31% is obtained using the baseline model, while the finetuned model obtains 42%. In comparison, the baseline model on the original images (1.44 bpp) obtains 44.5%. At 0.6 bpp the mIoU of the finetuned model on GMM hyperprior compressed images matches the performance of the baseline model on the original images.

5. Conclusion

We investigated the impact of image compression on visual recognition, using both traditional codecs and recent neural compression methods for compression levels ranging from moderate (2 bpp) to very strong compression (0.1 bpp). We find that strong compression has a big negative impact on the accuracy for tasks such as image classification, object detection and semantic segmentation. Our experiments show that this is to a large extent due to the lack of generalization of these models to images with compression artefacts. By finetuning the recognition models on compressed images, we find that most of the loss in accuracy on compressed images can be recovered.

Our findings can contribute to deploy visual recognition for users in resource and bandwidth limited settings. In future work we want to explore to what extent our findings can be used to reduce I/O bound latency when training visual recognition models on internet-scale datasets. In particular, it is interesting to explore training recognition models directly on the latent compressed image representations, rather than passing through the usual RGB representation.

Photo credits. Figure 1 main photo by Ajay Suresh, CC License 2.0 (cc2). Figure 1 small photo by Zhaoshan75, CC License 4.0 (cc4). Figure 2 by Deensel, CC License 2.0 (cc2).

References

Creative commons attribution 2.0 generic license. https://creativecommons.org/licenses/by/2.0/deed.en.

Creative commons attribution-share alike 4.0 international license. https://creativecommons.org/licenses/by-sa/4.0/deed.en.

PIL library. https://pillow.readthedocs.io/.

WebP image format. https://developers.google.com/speed/webp/.

Ballé, J., Minnen, D., Singh, S., Hwang, S. J., and Johnston, N. Variational image compression with a scale hyperprior. In *ICLR*, 2018. URL https://openreview.net/forum?id=rkcQFMZRb.

Bégaint, J., Racapé, F., Feltman, S., and Pushparaja, A. CompressAI: a pytorch library and evaluation platform for end-to-end compression research. *arXiv preprint*, arXiv:2011.03029, 2020.

Bellard, F. BPG image format. https://bellard.org/bpg/.

Caesar, H., Uijlings, J., and Ferrari, V. COCO-Stuff: Thing and stuff classes in context. In *CVPR*, 2018.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. MMDetection: Open MMLab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

Chen, Z., Yang, C., Li, Q., Zhao, F., Zha, Z.-J., and Wu, F. Disentangle your dense object detector. In *ACM International Conference on Multimedia*, 2021.

Cheng, Z., Sun, H., Takeuchi, M., and Katto, J. Learned image compression with discretized Gaussian mixture likelihoods and attention modules. In *CVPR*, 2020. URL https://arxiv.org/abs/2001.01568.

Choi, H. and Bajić, I. V. Deep feature compression for collaborative object detection. In *ICIP*, 2018.

Cohen, R. A., Choi, H., and Bajić, I. V. Lightweight compression of intermediate neural network features for collaborative intelligence. *IEEE Open Journal of Circuits and Systems*, 2:350–362, 2021.

Contributors, M. OpenMMLab's image classification toolbox and benchmark. https://github.com/open-mmlab/mmclassification, 2020a.

Contributors, M. MMSegmentation: OpenMMLab semantic segmentation toolbox and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020b.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. URL https://arxiv.org/abs/2010.11929.

El-Nouby, A., Muckley, M. J., Ullrich, K., Laptev, I., Verbeek, J., and Jégou, H. Image compression with product quantized masked image modeling. *TMLR*, 2023.

Fan, A., Stock, P., Graham, B., Grave, E., Gribonval, R., Jegou, H., and Joulin, A. Training with quantization noise for extreme model compression. In *ICLR*, 2021. URL https://arxiv.org/abs/2004.07320.

Ghosh, S., Yao, J., and Doshi-Velez, F. Structured variational learning of Bayesian neural networks with horseshoe priors. *ICML*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. *arXiv preprint arXiv:1602.07360*, 2016.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *CVPR*, 2018.

LeCun, Y., Denker, J., and Solla, S. Optimal brain damage. In *NeurIPS*, 1990.

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. Pruning filters for efficient convnets. *ICLR*, 2017.

Lin, D., Talathi, S., and Annapureddy, S. Fixed point quantization of deep convolutional networks. In *ICML*, 2016.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

MacKay, D. J. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

Masana, M., van de Weijer, J., Herranz, L., Bagdanov, A. D., and Alvarez, J. M. Domain-adaptive deep network compression. In *ICCV*, 2017.

Mentzer, F., Agustsson, E., Tschannen, M., Timofte, R., and Gool, L. V. Practical full resolution learned lossless image compression. In *CVPR*, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Mentzer_Practical_Full_Resolution_Learned_Lossless_Image_Compression_CVPR_2019_paper.html.

Minnen, D., Ballé, J., and Toderici, G. D. Joint autoregressive and hierarchical priors for learned image compression. In *NeurIPS*, 2018. URL https://papers.nips.cc/paper/2018/hash/53edebc543333dfbf7c5933af792c9c4-Abstract.html.

n, M. T. and Le, Q. V. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. URL https://arxiv.org/abs/1905.11946.

Nakahara, M., Hisano, D., Nishimura, M., Ushiku, Y., Maruta, K., and Nakayama, Y. Retransmission edge computing system conducting adaptive image compression based on image recognition accuracy. In *IEEE Vehicular Technology Conference*, 2021.

Oord, A. v. d., Vinyals, O., and Kavukcuoglu, K. Neural discrete representation learning. In *NeurIPS*, 2017. URL https://papers.nips.cc/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf.

Park, J. and Johnson, J. RGB no more: Minimally-decoded JPEG vision transformers. *arXiv preprint*, 2211.16421, 2022.

Razavi, A., van den Oord, A., and Vinyals, O. Generating diverse high-fidelity images with VQ-VAE-2. In *NeurIPS*, 2019.

Rippel, O. and Bourdev, L. Real-time adaptive image compression. In *ICML*, 2017. URL https://arxiv.org/abs/1705.05823.

Sullivan, G. J., Ohm, J.-R., Han, W.-J., and Wiegand, T. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, 2012. doi: 10.1109/TCSVT.2012.2221191.

Tai, C., Xiao, T., Zhang, Y., Wang, X., et al. Convolutional neural networks with low-rank regularization. *ICLR*, 2016.

Veniat, T. and Denoyer, L. Learning time/memory-efficient deep architectures with budgeted super networks. In *CVPR*, 2018.

Wang, X., Zhou, Z., Yuan, Z., Zhu, J., Sun, G., Cao, Y., Zhang, Y., and Sun, K. FD-CNN: A frequency-domain FPGA acceleration scheme for CNN-based image processing applications. *ACM Trans. Embed. Comput. Syst.*, 2022.

Wiles, O., Carreira, J., Barr, I., Zisserman, A., and Malinowski, M. Compressed vision for efficient video understanding. In *ACCV*, 2022. URL https://arxiv.org/abs/2210.02995.

Xiao, T., Liu, Y., Zhou, B., Jiang, Y., and Sun, J. Unified perceptual parsing for scene understanding. In *ECCV*, 2018. URL https://arxiv.org/abs/1807.10221.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. Scene parsing through ADE20K dataset. In *CVPR*, 2017. URL https://openaccess.thecvf.com/content_cvpr_2017/html/Zhou_Scene_Parsing_Through_CVPR_2017_paper.html.