Wearable Split Computing with DANCE
Data Adaptive Neural Compression Engine

DAN JACOBELLIS, The University of Texas at Austin, USA

Split computing is an approach to mobile, embedded, and wearable sensing where minimal on-device computation is used to transform raw sensor data into compressed representations that can efficiently transmitted for further processing in the cloud. The widening gap between foundation models and conventional perception approaches in terms of versatility, ease of deployment, and computational cost makes split-computing more attractive than ever. However, standard lossy compression techniques have not kept up with the high number and fidelity of modern sensors, leading vast amounts of data to be stored indefinitely or discarded entirely. To address this, we propose DANCE (**D**ata **A**daptive **N**eural **C**ompression **E**ngine), a framework to build high-efficiency neural codecs via specialization to a given dataset or sensor. For the 7-channel spatial audio and fisheye camera on the Project Aria smartglasses, split-computing via DANCE outperforms conventional codecs like JPEG in terms of both compression ratio and machine perceptual quality, while offering more than 100x more efficient inference compared to existing neural audio and image compression methods. Our code and additional examples are available at danjacobellis.net/SPLIT.

CCS Concepts: • **Theory of computation → Data compression**; • **Human-centered computing → Mobile computing**; • **Hardware → Displays and imagers**; **Sound-based input / output**; • **Computing methodologies → Neural networks**; **Image compression**.

Additional Key Words and Phrases: wearable devices, lossy compression, neural compression, embedded systems, sensors, tinyML, smartglasses, real-time processing.

1 INTRODUCTION

Historically, the size, weight, and power (SWaP) constraints of wearable devices necessitate two modes of operation: (1) a low-power segmentation stage to detect for relevant activity which triggers (2) a computation-intensive processing and prediction stage [3]. Although suitable for applications with high inactivity, some applications, such as hearing [15] and vision [8] assistants, may be used frequently throughout the day. Advancements in sensors, data converters, and mobile processors now allow wearables to record 24-hour spatial audio, multi-view video, and more on smartwatch-sized batteries [11]. Still, the high-fidelity signals collected by such wearables cannot be used in real-time, since the additional power required to process on-device or transmit to the cloud is untenable.

Techniques to lower model resource requirements (aka tinyML) address many difficulties with on-device deployment, but fail to deliver the versatility and simple development process of larger foundation models [2]. Utilizing more potent lossy compression would allow real-time transmission for cloud processing, a paradigm known as split computing [10]. However, the only systems that offer sufficient compression rates to efficiently transmit the information-dense measurements collected by modern sensors use neural networks which are also too expensive, in terms of memory and power consumption, for on-device operation [5] [4].

To address these limitations, we propose DANCE (**D**ata **A**daptive **N**eural **C**ompression **E**ngine). While current learned compression architectures are highly specialized for human perceptual quality of RGB images and stereo audio, DANCE uses a relatively simpler, but parameterized, rate-distortion autoencoder[1] to efficiently encode 1D, 2D, and 3D signals with any number of channels. Our experiments include the development of a neural codec for compression of the 7-channel audio and fisheye camera collected from Aria smartglasses[11] in the the Aria Everyday Activities

Author's Contact Information: Dan Jacobellis, danjacobellis@utexas.edu, The University of Texas at Austin, Austin, TX, USA.

dataset [9]. We show that split computing using DANCE offers better performance compared to the most common audio and image codecs, MP3 and JPEG, while offering reduced computational requirements compared to the state of the art neural codecs like EnCodec [5] and DGML [4].

2 BACKGROUND AND RELATED WORK

Conventional compression standards or "codecs" use efficient subquadratic transforms, like the the discrete cosine transform (DCT) used in JPEG and MP3, to eliminate redundancies and compact signal energy into fewer sub-band coefficients. Quantization matrices designed based on human sensitivity are used to allocate more bits to the most perceptually important frequencies. These methods offer significantly higher compression ratios than lossless coding and remain effective for broad range of signal types (e.g. natural images, synthetic images, speech, music, etc), yet only require minimal memory and computation to encode data. The compression standards of the 1990s like JPEG and MP3 that adopt this approach remain ubiquitous for mobile, embedded, and wearable sensing, and ASICS that implement these standards are widely available.

The success of deep learning in the 2010s led to an increased focus on neural network-based compression. Autoencoders trained using rate-distortion [1] and vector-quantization [12] objectives quickly surpassed transform coding using wavelets and remain state of the art for both for audio [5] and image [6] compression.

Since audio and video streaming are the most widely used applications of lossy compression, ongoing research efforts focus on increasing the decoder efficiency in neural compression systems [14], but the corresponding encoders remain prohibitively expensive for most mobile and wearable applications. As a result, split computing applications must either discard additional information via resolution reduction or perform compression of intermediate features [10]. While the feature compression approach has many benefits, it does not permit reconstruction of the original signal, making it difficult or impossible to implement new types of perception in a backwards-compatible way or utilize the latest foundation models for zero-shot learning.

Inspired by the successful deployment of asymmetric (expensive encoder, cheap decoder) neural compression systems on mobile devices, we aim to develop a complementary asymmetric architecture that leverages a powerful, but expensive decoder to allow extremely cheap encoding for mobile and wearable sensing.

3 DANCE

The overall workflow of DANCE is shown in Figure 1. The encoder consists of two alternating layer types: (1) strided convolution and (2) generalized divisive normalization [7]. For the decoder, we adopt the architecture of [4], which includes attention blocks, residual layers, and transposed strided convolution. The compression objective is achieved using the same entropy bottleneck technique as Balle et. al. [1]. Since we do not rely on any form of vector quantization, this autoencoder structure permits a wide range range of 1D, 2D, and 3D inputs, including spatial audio, hyperspectral images, and many others, simply by choosing the corresponding type of multidimensional convolution. However, a limitation is that all signals must still be uniformly sampled on a regular grid, and convolutional layers with dimension four or greater are not supported by any major deep learning framework. Finally, the width, depth, and latent size of the networks are chosen using heuristics that take into account the user-specified rate-distortion Lagrangian λ as well as the dimension and number of channels in the input.

4 METHODOLOGY

Our experiments are based on data collected via Project Aria smartglasses [11] from the Aria Everyday Activities (AEA) dataset [9], which consists of several scripted human activity sequences

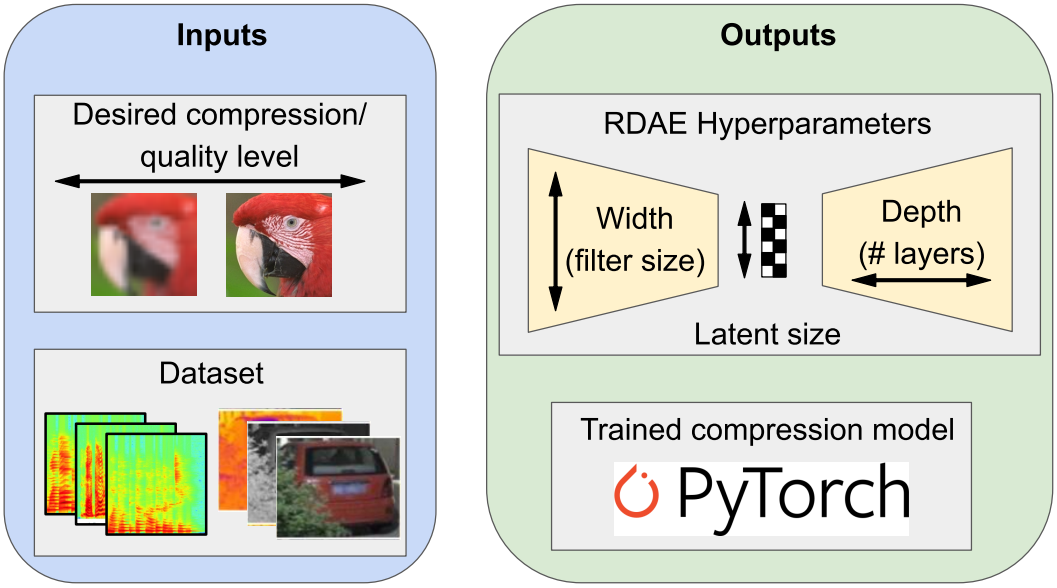

Fig. 1. Overview of DANCE. The user specifies the desired compression/quality level and provides a dataset of 1D, 2D, or 3D uniformly sampled signals with an arbitrary number of channels. Heuristics are used to choose the width (filter size), depth (number of layers), and latent size of a rate-distortion autoencoder. Training is performed to produce a customized, efficient compression model in PyTorch

across various locations. The Aria smartglasses can efficiently collect and store many streams of high-resolution data, but lack the ability to perform any type of real-time perception, making it an ideal platform for split-computing. We download the full AEA dataset (roughly 300 GB), consisting of dozens of data sequences collected using the default recording profile on the Aria smartglassses, which includes 1 frame per second images from the 8MP front-facing fisheye camera, and 48kHz audio from the 7-channel microphone array.

Fig. 2. Comparison of an image compressed using JPEG and DANCE. An additional 8× resolution reduction in applied to the JPEG image before encoding in order to reach a similar final file size. The compression ratio of the JPEG image is 726:1, while the compression ratio of the DANCE image is 817:1.

4.1 Fisheye camera compression and visual activity recognition

We extract 120k frames from the AEA dataset, and divide them into a training split (100k images) and a validation split (20k images), with no sequences being shared between the two splits. Although the original recording profile samples 8MP (2816×2816) images at one frame per second, the VRS files contain (1408×1408) images which are compressed using JPEG at quality level 80. However, because of the high resolution and quality level, this amount of compression should be considered near-lossless for virtually all common computer vision systems, which typically reduce the resolution to 512 or lower during pre-processing.

Kitchen	Cooking	Food	Drink	Spill	Table	Television
Phone	Laptop	Video game	Board game	Clothes	Laundry	

Table 1. 13 classes used to evaluate zero-shot image classification performance.

Fig. 3. Some example images from the Aria Everyday Activities dataset along with the class label assigned by performing zero-shot classification by applying CLIP to the original image without additional compression.

Based on the scripted activities in the dataset, we choose 13 classes, shown in Table 1, that represent common visual elements in the dataset. We train a low-complexity image codec using DANCE on a single RTX 4090 for 20 epochs. Figure 2 shows a visual comparison of the resulting codec compared to JPEG. Using CLIP, we compute the similarity between the 20k near-lossless images in the validation set and the 13 classes listed in Table 1. We then filter out any images whose maximum clip logit is less than 18, resulting only in images that have an associated class with confidence greater than 85%. Some examples of these high-confidence images are shown in Figure 3. Then, we encode and decode each of these filtered images using three compression methods: (1) JPEG, (2) DGML[4], (3) and DANCE, and recompute the CLIP similarity on the compressed versions of the images. Finally, we count the fraction of compressed images whose zero-shot predictions

are identical to the uncompressed images, and report the accuracy in Table 2. This label retention accuracy is used as a way to measure the performance of each compression method in the context of a typical task enabled by split computing (zero-shot image classification using CLIP). To estimate the computational requirements, we directly count the number of parameters in the pytorch encoder models, and estimate the MACs/pixel required by the convolutional layers in the encoder for both DANCE and DGML, also reported in Table 2.

Method	Encoder Params	Enc. MACs/px	Compression Ratio	Accuracy
RR+JPEG	0	<10	760:1	27.2%
DGML[4]	2.53 Million	99,200	408:1	56.8%
DANCE	0.285 Million	506	783:1	51.9%

Table 2. Fisheye image compression results. Since even the lowest JPEG quality setting only gives moderate compression ratios, we apply additional resolution reduction of 8× to achieve a similar compression ratio as our method. Accuracy reported is the fraction of compressed images that had identical zero-shot classification predictions using CLIP as the uncompressed version

4.2 Learned spatial audio compression

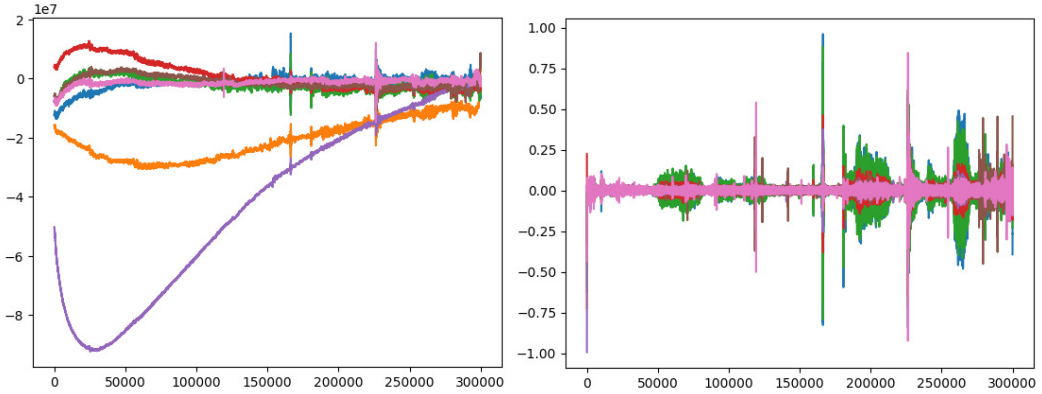

Fig. 4. Before (left) and after (right) preprocessing the 7-channel audio of the aria dataset, which exhibits significant drift.

We extract the all audio sequences from the AEA dataset (roughly 30GB) in 6-second segments, and divide into training and validation splits. We develop a prepossessing pipeline, shown in Figure 4, consisting of a companding operation and notch filter to to address the large drift in DC offset and high dynamic range which led to poor performance in initial attempts to train the compression model. We train an audio codec using DANCE for 20 epochs. A visual comparison of the reconstruction quality is shown in 5. We compare the performance of three compression methods: (1) MP3, (2) EnCodec[5], and (3) DANCE based on the signal to spatial distortion ratio, and the signal to residual distortion ratio both of which are highly correlated with the performance of downstream spatial audio perception tasks [13]. The results are shown in Table 3.

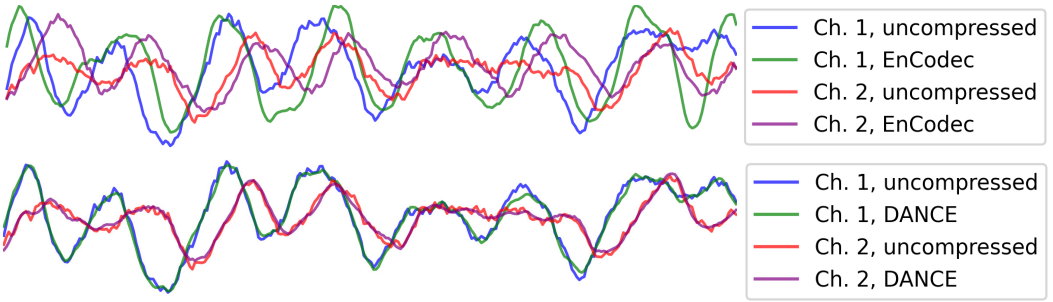

Fig. 5. Comparison of the first two channels of a recording in the validation set, compressed using EnCodec and DANCE. Although EnCodec technically supports multiple channels, it's training objectives, which focus on human perceptual quality, average out most of the differences between the channels. By training on only on the target array geometry, DANCE is easily able to model these channel differences.

Method	Enc. Params	Compression Ratio	SSDR	SRDR
MP3	0	31:1	19.1 dB	7.16 dB
EnCodec[5]	7.43 Million	114:1	2.79 dB	-11.7 dB
DANCE	0.331 Million	601:1	13.6 dB	2.52 dB

Table 3. Spatial audio compression results. SSDR is the signal to spatial distortion ratio and SRDR is the signal to residual distortion ratio.

5 RESULTS

The key results are shown in Table 2 for image compression and Table 3 for audio compression. Visual comparisons are also shown in Figure 2 and Figure 5. In both cases, the codecs learned with DANCE offer extremely high compression ratios and better quality than conventional codecs, but without using specialized architectures, perceptual losses, or adversarial training. For images, the zero-shot classification retention remains close to DGML [4], but with a savings of over 190× in MACs/pixel. For spatial audio, the DANCE-based codec achieves nearly 200× higher compression ratio MP3 and over 5× higher compression ratio than EnCodec[5]. In addition, the DANCE-based codec improves both the signal to spatial distortion ratio and the signal to residual distortion ratio by over 10dB compared to EnCodec.

6 DISCUSSION

Our results indicate a large opportunity to apply DANCE to various mobile and wearable sensing applications by allowing immediate processing via split-computing instead of indefinitely storing or discarding data, a practice that is all-too common today. However, several limitations of this work should be pointed out, as they form the basis for key improvements in future work.

Firstly, The heuristics used to map the dataset dimension, channel count, and desired compression ratio to encoder width, depth, and latent size were determined based on a handful of experiments across just a few datasets, are are likely far from optimal. Applying our framework to a wider variety of datasets and creating a procedure to systematically and statistically create this mapping would be necessary for our approach to see it's full potential.

Secondly, our analysis of the computational cost does not include the cpu-heavy entropy coding stage. Even with the large improvements in the analysis transform, finding efficient approaches to

the entropy coding stage that are applicable to a wide range of inputs is a challenging, but necessary problem to solve in order to enable split computing on the widest range of devices.

Finally, while there are many reasons to believe that the chosen metrics (CLIP retention, SSDR, and SRDR) are good predictors of downstream performance in the context of split-computing, they are still noisy proxies to the true performance. Additionally, our we do not compare to direct on-device inference, which has seen rapid improvements and remains a strong competitor to split computing.

REFERENCES

[1] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. 2017. End-to-end optimized image compression. In *5th International Conference on Learning Representations, ICLR 2017*.

[2] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258* (2021).

[3] Andreas Bulling, Ulf Blanke, and Bernt Schiele. 2014. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Computing Surveys (CSUR)* 46, 3 (2014), 1–33.

[4] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. 2020. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 7939–7948.

[5] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. 2022. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438* (2022).

[6] Emiel Hoogeboom, Eirikur Agustsson, Fabian Mentzer, Luca Versari, George Toderici, and Lucas Theis. 2023. High-fidelity image compression with score-based generative models. *arXiv preprint arXiv:2305.18231* (2023).

[7] Nick Johnston, Elad Eban, Ariel Gordon, and Johannes Ballé. 2019. Computationally efficient neural image compression. *arXiv preprint arXiv:1912.08771* (2019).

[8] Guoxin Li, Jiaqi Xu, Zhijun Li, Chao Chen, and Zhen Kan. 2022. Sensing and navigation of wearable assistance cognitive systems for the visually impaired. *IEEE Transactions on Cognitive and Developmental Systems* 15, 1 (2022), 122–133.

[9] Zhaoyang Lv, Nickolas Charron, Pierre Moulon, Alexander Gamino, Cheng Peng, Chris Sweeney, Edward Miller, Huixuan Tang, Jeff Meissner, Jing Dong, et al. 2024. Aria Everyday Activities Dataset. *arXiv preprint arXiv:2402.13349* (2024).

[10] Yoshimoto Matsubara, Ruihan Yang, Marco Levorato, and Stephan Mandt. 2023. Sc2 benchmark: Supervised compression for split computing. *Transactions on machine learning research* (2023).

[11] Kiran Somasundaram, Jing Dong, Huixuan Tang, Julian Straub, Mingfei Yan, Michael Goesele, Jakob Julian Engel, Renzo De Nardi, and Richard Newcombe. 2023. Project Aria: A new tool for egocentric multi-modal AI research. *arXiv preprint arXiv:2308.13561* (2023).

[12] Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems* 30 (2017).

[13] Karn N Watcharasupat and Alexander Lerch. 2024. Quantifying Spatial Audio Quality Impairment. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 746–750.

[14] Yibo Yang and Stephan Mandt. 2023. Computationally-efficient neural image compression with shallow decoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 530–540.

[15] Fraser Young, Li Zhang, Richard Jiang, Han Liu, and Conor Wall. 2020. A Deep learning based wearable healthcare iot device for ai-enabled hearing assistance automation. In *2020 international conference on machine learning and cybernetics (ICMLC)*. IEEE, 235–240.