DOI: 10.61782/fa.2025.0530

LOSSY COMPRESSION FOR DISTRIBUTED ACOUSTIC SENSING

Lucas Eckert[1] **Vlad Dumitru**[2] **Martin Hagmüller**[1*]

[1] Signal Processing and Speech Communication Laboratory
[2] Institute of Engineering Geodesy and Measurement Systems
Graz University of Technology, Austria

ABSTRACT

This paper evaluates the performance of lossy compression algorithms for data from Distributed Acoustic Sensing (DAS) systems, which use fibre optic cables to detect vibrations at discrete locations along their length. While DAS provides high spatial and temporal resolution with real-time, continuous acquisition, it generates vast amounts of data, creating challenges for both transmission and storage. To address these issues, various compression algorithms, commonly used in music and speech processing, were tested on publicly available DAS datasets. The evaluation considered key metrics such as reconstruction fidelity, computation time, and compression rate. Since DAS systems operate at much lower sampling rates than audio applications, acquired data is collected in a buffer and processed at higher sampling rates by the audio codecs. After comparing multiple algorithms, the OPUS codec was selected due to its flexibility across bit rates, low latency, consistent performance, and high adaptability. A real-time compression system was developed based on OPUS, capable of handling five-digit channel counts. The system is configurable to meet task-specific requirements, allowing adjustments between compression rate and reconstruction accuracy as needed. The proposed solution significantly reduces storage needs and enables efficient low-bandwidth data transmission, making it well suited for real-time DAS applications.

Keywords: *Distributed Acoustic Sensing, lossy compression, OPUS codec*

**Corresponding author*: *hagmueller@tugraz.at.*

1. INTRODUCTION

Distributed Acoustic Sensing (DAS) is a technology that uses fiber optic cables to detect vibrations along the length of the cable. Unlike traditional geophones which measure displacement or particle velocity, the measurand of DAS interrogator units is strain rate, i.e. the temporal change in strain. The sensing technique has found applications across various domains, including seismic monitoring (e.g., [1]), pipeline security (e.g., [2]), and structural health monitoring (e.g., [3]), due to its high spatial and temporal resolution and real-time data acquisition capabilities. However, the vast amount of data generated by DAS systems presents significant challenges in terms of storage, transmission, and processing.

The high demand of resources stems from its continuous, high-frequency sampling across potentially long distances with densely spaced measurement positions, resulting in terabytes of data even for relatively short observation periods. Additionally, DAS data is often acquired in remote areas with highly limited transmission rates like mountain ranges or the seafloor. Efficient data compression methods are therefore essential to utilize DAS data effectively. Compression not only reduces storage requirements but also enhances data transmission efficiency, enabling real-time monitoring and analysis in resource-constrained environments.

Although there is a lot of literature on compression of data in general (e.g., [4]), or specific like multichannel compression for medical purposes (EEG/ECG) (e.g., [5]) or lossy and lossless compression of music (e.g., [6]) or speech (e.g., [7]). At the time of writing, only few publications on the topic of compression for DAS systems are known to the authors (e.g., [8], [9]). This paper addresses the critical need for effective DAS data compression techniques. We focus on already existing algorithms used in speech and music compression and evaluate their use for DAS data. Compression rates are assessed and compared in terms of computational complexity and reconstructed signal quality. We additionally evaluate multi-class classification on original and OPUS-compressed data to further assess the quality and usability of the algorithm for tasks like acoustic event recognition.

1.1 Publicly Available DAS Datasets

There are multiple DAS datasets available publicly. In [10] eight are listed and a combined repository PubDAS with all of these is presented. The paper also lists datasets that are available online, but not contained in PubDAS. Also, there is data from the Imperial Valley Dark Fiber Project [11]. While [8] state the data they used, and it is partly contained in PubDAS, their exact used time frames are not available.

We used the Brady Hot Springs PoroTomo dataset [12], since it did not only contain noise, but also an earthquake. It was used for both the codec comparison described in Sec. 3 as well as the parameter study in Sec. 3.2.

Brady Hot Springs PoroTomo:

- Recorded: 2016-03-21 at 07:37:51 a.m.
- Length: 30 seconds
- Channels: 8721
- Data type: 16-bit integer
- Contains an Earthquake: M 4.3 - 23km ESE of Hawthorne, Nevada; Time: 2016-03-21 07:37:10 (UTC); Location: 38.479 N 118.366 W; Depth: 9.9 km

1.2 Related Work

Data compression is established in many fields, but only few studies focus on DAS specifically. In [8] lossless techniques are used on DAS data in both real time and post hoc scenarios. For evaluation of their compression techniques, they used the Sacramento FOSSA dataset described in [1] and partly contained in the PubDAS repository [10]. They also used data from the Imperial Valley Dark Fiber Project (Imp Valley) [13] and from Monterey Bay [14]. On these datasets up to 40 % reduction, or a compression rate of up to 1.65, was achieved.

In [9] lossy compression techniques were used to compress DAS data. The methods used were wavelet compression, Zfp floating point compression and truncated SVD compression. For evaluation the FORESEE dataset [15], which is also contained in PubDAS [10], was used as well as the Brady Hot Springs PoroTomo dataset [12] for event detection. In this paper the norm of the noise is computed. Noise is defined by the difference of the original data D and the reconstructed compressed data $\tilde{D}$. The norm is then computed using the normalized Frobenius norm error according to Eq. 3.

The results of the lossless compression performed in [8] are given in Tab. 2 together with the approximate results of the lossy compression done by [9] and compared to our own results.

2. COMPRESSION ALGORITHMS

Data compression depends on compression rate, quality, computational complexity, delay, editability, and error resilience. The key is balancing data reduction while retaining necessary information. Real-time processing requires low complexity and minimal delay. Editability and error resilience determine ease of modification and robustness against transmission errors. Depending on the application, different trade-offs are required. There are two main approaches to data reduction.

(i) Lossless approaches focus on reducing redundancy and include information theoretic methods like Huffman-coding or dictionary-based methods like Lempel-Ziv-Welch-coding in addition to linear prediction. For lossless coders, the quality assessment is not applicable, but the other criteria mentioned above are. Well known lossless audio coders are FLAC, ALAC or AIFF.

(ii) Lossy compression reduces both redundancy and irrelevance, meaning data cannot be exactly reconstructed after decoding. Here, the main goal is to achieve maximum reduction while preserving essential information, which varies by application. For audio, psychoacoustic principles exploit masking effects, audibility thresholds, and ear sensitivity to reduce data without affecting perception. Quantization noise is not perceivable if it does not exceed the perceptual threshold. Predictive techniques like Linear Predictive Coding (LPC) improve efficiency by transmitting only model parameters and prediction errors. While music codecs prioritize perceptual quality, speech codecs focus on intelligibility and real-time processing. The original signal is not kept, but regenerated upon reconstruction. There are many codecs for audio and speech compression, some of which will be compared in this paper. In structural health monitoring, seismic detection, or pipeline security, compression must retain event detectability. Algorithms used on raw data must also perform the same on compressed data. Depending on the application, a controlled loss of detection may be acceptable to achieve higher compression, especially when only specific events are relevant.

The efficiency of the compression is measured in compression ratio R (Eq. 1) or reduced size in percent.

Due to its higher compression rates, we used lossy compression techniques. Due to the structure of DAS data, having many channels and recording continuously, real-time capability and computational efficiency need to be factors for performance evaluation.

3. COMPRESSION OF DAS DATA

For an initial comparison of lossy compression codecs, *FFmpeg* [16] was used. The codecs for our comparison are listed below, including the library used for en-/decoding. If no library is stated, the native implementation of *FFmpeg* was used. In addition to the lossy codecs, *FLAC* was used to compare the compression rates.

- MP3 - encoder: libmp3lame
- GSM - encoder: libgsm
- AMR-NB - en-/decoder: OpenCORE AMR-NB
- AMR-WB - encoder: Android VisualOn AMR-WB; decoder: OpenCORE AMR-WB
- OPUS - en-/decoder: libopus
- FLAC

Most publicly available datasets are stored in the HDF5-format. For comparison purposes, the dataset is extracted and converted to 16-bit integer if necessary. Note, that the compression rate refers to the 16-bit version of the dataset, even if originally recorded at a higher bit depth. Then every channel is compressed with FFmpeg, using the corresponding encoder of the chosen codec. Audio and speech codecs do not operate at the same sample rates as DAS data is sampled at. The *original sample rate* is denoted as f_{so} and usually lies below 2 kHz. The *compression sample rate* f_{sc} is set according to the capabilities of the codecs. *OPUS*, *MP3* and *FLAC* are tested at an f_{sc} of 8 kHz, 16 kHz, 24 kHz and 48 kHz. No resampling needs to be done, one second of the audio file at $f_{so} = 2$ kHz then consists of 4, 8, 12 and 24 seconds, respectively, of the DAS data. *GSM* and *AMR-NB* are tested at an f_{sc} of 8 kHz and *AMR-WB* at 16 kHz. The lossy codecs allow a range of bits used per second to encode the signal (bit rate). The *bit per second* (bps) are not to be confused with *bit per sample* (*bit/sample*). For $f_{sc} = 48$ kHz and a bit rate of $r_b = 96$ kbps the signal will have 2 bit/sample. We test *GSM*, *AMR-NB* and *AMR-WB* at their highest possible bit rates of 13 kbps, 12.2 kbps and 23.85 kbps respectively, resulting in ≈ 1.5 bit/sample. *OPUS* and *MP3* are tested at $[0.5, 1, 2, 3, 4]$ bit/sample. After compression, the size of the compressed data is determined, and the compression rate R calculated (see Eq. 1).

$$R = \frac{S_{\text{orig}}}{S_{\text{comp}}}, \tag{1}$$

where S_{orig} is the size of the original data and S_{comp} is the size of the compressed data.

This is followed by reconstruction using the corresponding decoder. To assess the quality of the compressed data, two different time domain measures Q were used: mean absolute scaled error (MASE) and Normalized Frobenius Norm Error (NFNE) as described in [9]. It was also considered to include the mean squared error (MSE) and the mean absolute percentage error (MAPE). The MSE is dependent on the amplitude on the signal and weighs outliers heavily, due to the square. MAPE was excluded, because the DAS data includes zero and close to zero values, for which MAPE tends to infinity. We let $\mathbf{Y}$ represent the original data and $\hat{\mathbf{Y}}$ the encoded (compressed) and decoded (reconstructed) data.

MASE. This measure was chosen due to the following properties. In contrast to the MSE, it is scale invariant, meaning that it is independent of the amplitude of the data, and it does not weigh outliers heavily. Additionally, in contrast to MAPE, it does not tend to infinity for values close to zero. For non-time-series data, it is estimated by Eq. 2. With this measure, classification is clear with the best achievable value being 0. So it is neither necessary nor possible to alter the data as with the MSE, since division of the data by a maximum value would cancel out in any case.

$$Q_{\text{MASE}} = \frac{\sum_{i=1}^{n} |Y_i - \hat{Y}_i|}{\sum_{i=1}^{n} |Y_i - \overline{Y}|} \tag{2}$$

NFNE. In [9] a *normalized Frobenius norm error* is proposed for comparison of compression techniques. It is given in Eq. 3,

$$Q_{\text{NFNE}} = \frac{||\mathbf{Y} - \hat{\mathbf{Y}}||_F}{||\mathbf{Y}||_F}, \tag{3}$$

where $||\cdots||_F$ denotes the Frobenius norm. The norm of the difference is divided by the norm of the original data, but it is not scale invariant. This is due to outliers also weighing in more heavily. It is also robust against close to zero values within the data, since the overall norm is computed before division. The best achievable value is 0. However, this measure can yield values > 1, if some strong outliers are present. It also yields exactly 1 if the compression data would be empty. This makes this measure hard to interpret.

To show the behaviour of the compression algorithms in the frequency domain, we estimated the power spectral density (PSD) of white noise for 1000 channels and averaged them for original and compressed data each. The result can be seen in Fig. 2.

3.1 Comparison of Codecs

We use the Brady Hot Springs PoroTomo dataset as specified in Sec. 1.1 to compare the codecs. Q_{MASE} (Eq. 2) is shown vs compression rate R (Eq. 1) in Fig. 1.

As expected, we can see that R and Q_{MASE} depend on the bit/sample. We further see that MP3 and OPUS perform better than the speech codecs. The AMR codecs are way apart from the rest, but GSM can compete with the worst results of OPUS. For MP3 and OPUS there are several markers per bit/sample each in Fig. 1. This is due to different $f_{sc} = [8, 16, 24, 48]$ kHz being tested, the bigger the marker, the higher f_{sc}. The FLAC codec was excluded from this figure, since the Q_{MASE} of a lossless codec is always 0. But the resulting R for FLAC differed slightly depending on the sample rate, ranging from $R = 1.85$ for $f_{sc} = 8$ kHz to $R = 1.95$ for $f_{sc} = 48$ kHz. For MP3, the Q_{MASE} does not vary much for different sample rates, but R does. For OPUS R does not change much for 2, 3 or 4 bit/sample but Q_{MASE} does. For 1 bit/sample, both vary significantly. The difference in R can not be reasonably explained without looking into the internal structure, but might be similar to the compression rate explanation in Sec. 3.2. Q_{MASE} is lower for higher sample rates. This difference might be explained by comparing the PSDs of encoded and reconstructed white noise, as seen in Fig. 2 for 2 bit/sample. All reconstructed PSDs seem to cut off the signal at a certain frequency depending on the sample rate. For both MP3 and OPUS this cutoff-frequency is the lowest for $f_{sc} = 48$ kHz.

The codecs AMR-NB, AMR-WB, GSM, and OPUS at a sampling rate of $f_{sc} = 8$ kHz exhibit the least accurate estimation of the PSD compared to the uncompressed reference (thick black line). The PSDs of MP3-encoded signals tend to underestimate energy at low frequencies while overestimating it at high frequencies. OPUS at higher sampling rates provides the most accurate PSD approximation overall.

Both MP3 and OPUS apply low-pass filtering, with the effect of being more pronounced at higher sampling rates. This behaviour is attributed to their design as audio codecs, which prioritize perceptually relevant frequency bands for music and speech, typically located in the lower frequency range. Lower sampling rates are predominantly used for speech coding and aim to represent information across the entire band of perceptually relevant frequencies.

At lower bit or sample rates, PSD estimation becomes less accurate across all codecs, with some exhibiting significantly reduced cut-off frequencies. Nonetheless, OPUS maintains the best performance among them. At higher bit/sample rates, MP3 consistently underestimates the PSD across the entire spectrum, while OPUS achieves even greater accuracy.

OPUS at $f_{sc} = 16$ kHz and $f_{sc} = 24$ kHz shows a smoother spectral roll-off, indicating less aggressive filtering. Therefore, if high-frequency content carries relevant information, we recommend using $f_{sc} = 24$ kHz. Conversely, if the signal energy is concentrated in the low-frequency range—such as in seismic recordings or low-frequency event detection—$f_{sc} = 48$ kHz is preferred, as it yields superior PSD estimation in the lower frequency domain.

3.2 Parameter Study for Opus Codec

Based on the results presented in Sec. 3.1, we selected OPUS for implementation in a real-time encoding environment. The implementation was carried out in Rust using the audiopus crate with libopus 1.4. Every channel is encoded separately (mono). Initial testing revealed a constant sample offset between the original and reconstructed signals. This offset is independent of the frame size but depends on the sample rate, amounting to $52 \cdot f_{sc}/8$ kHz samples. To compensate, the input signal was padded with a sufficient number of frames to cover the offset. The padding marginally increases the encoded data size and is removed in the decoder. Frame sizes can be 2.5, 5, 10, 20, 40 or 60 ms, which results in different amount of samples for the same frame size and different sample rates. For the encoding process, inband forward error correction (FEC) and variable bit rate (VBR) were disabled. A constant bit rate (CBR) setting was used to preallocate memory. The encoder complexity parameter, which ranges from 0 to 10,

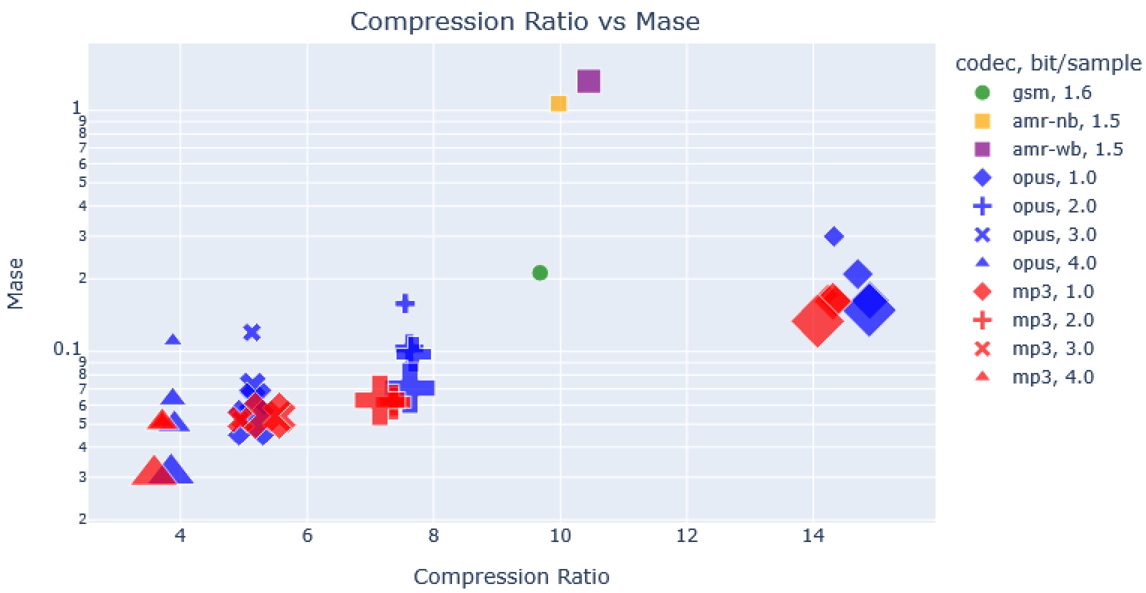

Figure 1. Q_{MASE} vs compression rate R. The dataset is Brady Hot Springs PoroTomo as specified in Sec. 1.1. The bit/sample given in the parentheses. MP3 and Opus were sampled at $f_{sc} = [8, 16, 24, 48]$ kHz, the higher f_{sc}, the bigger the marker size.

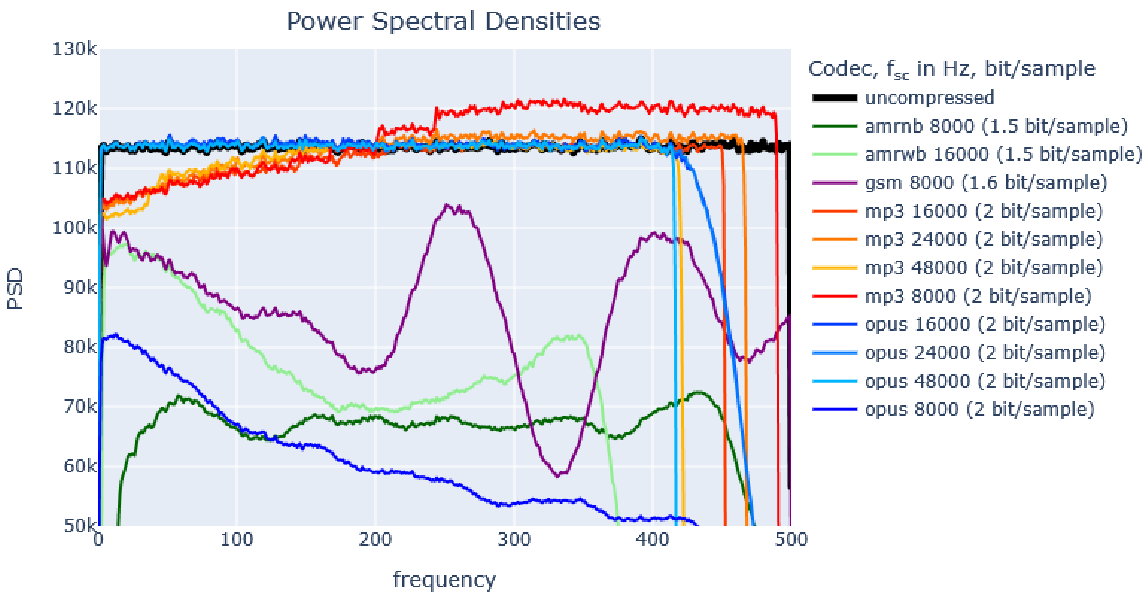

Figure 2. Power spectral density (PSD) using Welch's method of white noise, original (black) and encoded and reconstructed by OPUS and MP3. The PSDs in the red shades are for MP3, and in the blue shades for OPUS.

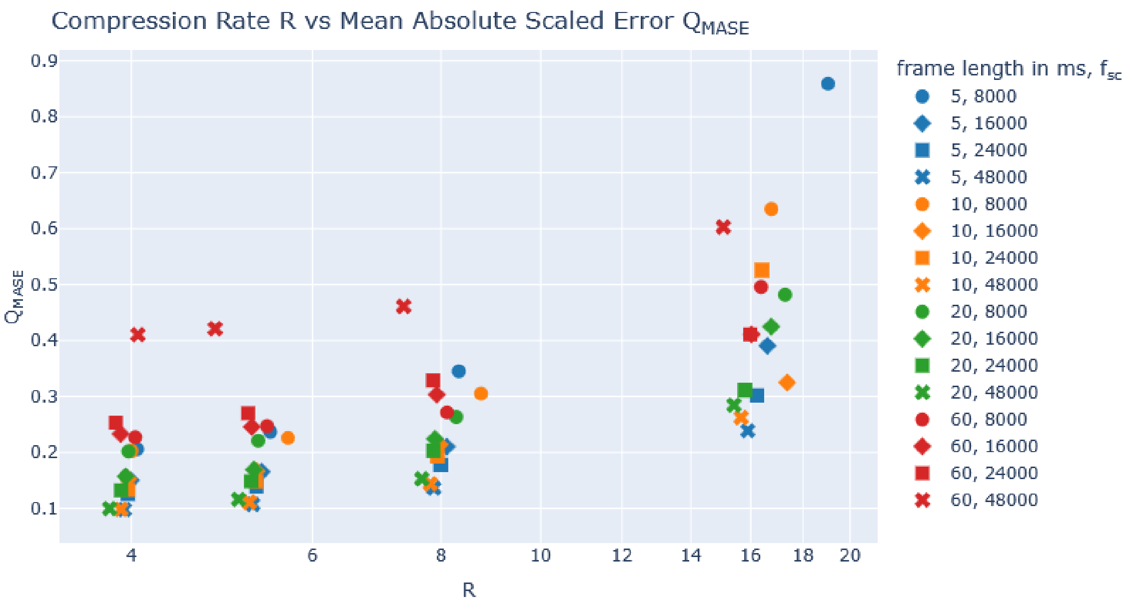

Figure 3. Q_{MASE} vs compression rate R. The dataset is Brady Hot Springs PoroTomo as specified in Sec. 1.1. The frame size in milliseconds is indicated by color, the f_{sc} by marker.

was set to 0 since experiments showed negligible influence on the output. From the three modes - audio, voice-over-IP (VoIP) and low-delay, that can be selected for the opus codec, we decided to use audio. VoIP will high pass filter the signal, which would not be target-oriented for DAS. Low-delay on the other hand will disable the speech optimized mode, which could also negatively affect the quality of the signal.

In the subsequent parameter study, various combinations of frame size, bit rate, and sample rate were evaluated. Even for equal bit/sample, minor variations in compression ratio R were observed. These variations stem from the fact that the frame size in samples does not necessarily divide the signal length evenly, padding to accommodate both the offset and full frame alignment is required and under CBR, only complete frames are encoded into the bitstream. Figure 3 shows the trade-off between R and Q_{MASE} across various parameter combinations, grouped by bit/sample ratios from 1 to 4. The 60 ms frame size consistently yields the poorest performance, aside from isolated outliers, while shorter frame sizes exhibit better results in both R and Q_{MASE}. For the PoroTomo dataset, OPUS achieves optimal performance at a sampling rate of $f_{sc} = 48$kHz across all bit/sample settings. This is likely due to the concentration of signal energy in lower frequencies, where accurate PSD reconstruction outweighs bandwidth extension. Although other datasets occasionally favour different f_{sc} values, the performance differences are small. We therefore recommend using $f_{sc} = 48$kHz with a 20 ms frame length for most applications.

Classification Accuracy. We analyse the performance of the OPUS codec on a multi-class classification problem in relationship with the achievable compression rates. The processing pipeline consists of MFCC feature extraction on overlapping windows, and the subsequent training on the resulting vectors using a random forest classifier. In the following table, *identity* refers to the identity function, i.e., no processing whatsoever. We show that training on OPUS-compressed data has only minor effects in comparison with the baseline of truncating the input data to `i16`.

3.3 Comparison to Related Work

We compare our results to the work described in Sec. 1.2 in Tab. 2. Both [9] and [8] used long time periods of DAS data for their tests. We instead used single files, containing 30 seconds to one minute of data, depending on the dataset. For the FOSSA dataset, data from the time span used in [8] is not available in the public repository Pub-

Table 1. Accuracy of a multi-class classification problem using original, truncated and OPUS compressed data.

algorithm	bit/sample	accuracy
identity (`f32`)	32	88.44%
truncation (`i16`)	16	87.65%
OPUS (`i16`)	0.5	86.53%
	2	87.38%
	4	87.38%

DAS [10]. Instead, data from September, 7th 2017 at 00:00:29 A.M. was used, to be comparable to the nighttime compression values achieved. The zfp-floating point compression was calculated for the FORESEE dataset by the authors. The difference in Q_{NFNE} can not be explained.

Overall, our results show that the compression rates and errors due to compression differ between datasets. It can also be seen, that OPUS can compete with the zfp-floating point fixed accuracy compression.

4. CONCLUSION

In this paper, we presented a real-time capable DAS compression system with adjustable quality settings that can be tailored to specific application requirements. The quality assessment demonstrates only minor degradation of classification performance in comparison to the uncompressed data. Our compression rates and quality measures compare well with the (sparse) existing literature. We propose the Mean Absolute Scaled Error (MASE) as a robust and scale-invariant metric for evaluating compression-induced distortion. Furthermore, we recommend using the PubDAS database or the Brady Hot Springs Data as a standardized benchmark for future comparative studies.

5. REFERENCES

[1] J. B. Ajo-Franklin, S. Dou, and N. J. Lindsey, "Distributed acoustic sensing using dark fiber for near-surface characterization and broadband seismic event detection," *Scientific reports*, vol. 9, no. 1, p. 1328, 2019.

[2] D. Hill and M. McEwen-King, "Distributed acoustic sensing for pipeline monitoring," 2009.

[3] V. Lanticq, F. Taillade, R. Gabet, and S. Delepine-Lesoille, *Distributed optical fibre sensors for Structural Health Monitoring: Upcoming challenges*. INTECH Open Access Publisher, 2009.

[4] Y. Collet, "Zstandard compression and the "application/zstd" media type," Tech. Rep. 8878, RFC Editor, February 2021.

[5] T.-H. Tsai and F.-L. Tsai, "Efficient lossless compression scheme for multi-channel ECG signal processing," *Biomedical Signal Processing and Control*, vol. 59, p. 101879, May 2020.

[6] C. K. Dai Tracy Yang and C.-C. J. Kuo, "High-fidelity multichannel audio coding," *EURASIP book series on Signal Processing and Communications*, vol. 2, 2006.

[7] L. Drude, J. Heymann, A. Schwarz, and J.-M. Valin, "Multi-channel opus compression for far-field automatic speech recognition with a fixed bitrate budget," 2021.

[8] B. Dong, A. Popescu, V. R. Tribaldos, S. Byna, J. Ajo-Franklin, and K. Wu, "Real-time and post-hoc compression for data from distributed acoustic sensing," *Computers & Geosciences*, vol. 166, p. 105181, Sept. 2022.

[9] A. Hafiz, S. Issah, and E. R. Martin, "Errors incurred in lossy compression of seismic data." `https://api.semanticscholar.org/CorpusID:269459971`.

[10] Z. J. Spica, J. Ajo-Franklin, and G. C. Beroza, "PubDAS: A PUBlic distributed acoustic sensing datasets repository for geosciences," *Seismological Research Letters*, vol. 94, pp. 983–998, Jan. 2023.

[11] J. Ajo-Franklin, P. Dobson, and V. Rodriguez Tribaldos, "Imperial valley dark fiber project continuous DAS data," 2020.

[12] K. Feigl, E. Reinisch, and J. Patterson, "Porotomo natural laboratory horizontal and vertical distributed acoustic sensing data." `https://gdr.openei.org/submissions/980`, Mar. 2016.

[13] J. Ajo-Franklin, V. Rodríguez Tribaldos, and A. Nayak, "The imperial valley dark fiber project: Toward seismic studies using das and telecom infrastructure for geothermal applications," *Seismological Research Letters*, vol. 93, pp. 2906–2919, 06 2022.

[14] N. Lindsey, C. Dawe, and J. Ajo-Franklin, "Photonic seismology in Monterey Bay: Dark fiber DAS illuminates offshore faults and coastal ocean dynamics," July 2019.

Table 2. Comparison of the performance of our own work on datasets used by related work. Codecs marked with * were used with FFmpeg. If no source is given, the calculations have been done by the authors. Results by [9] have been marked with [1], since they have been estimated from the diagram.

dataset	source	codec	R	Q_{MASE}	Q_{NFNE}
FOSSA [1]	[8]	ZigZag + TurboPfor	1.65	0	0
	[8]	Abs + TurboPfor	1.61	0	0
	—	flac*	1.35	0	0
	—	OPUS	15.37	0.459	0.426
	—	OPUS	7.81	0.239	0.226
	—	OPUS	3.91	0.090	0.079
	—	OPUS	1.99	0.054	0.045
Imp Valley [13]	[8]	ZigZag + TurboPfor	1.38	0	0
	[8]	Abs + TurboPfor	1.54	0	0
	—	flac*	1.26	0	0
	—	OPUS	16.71	0.711	0.623
	—	OPUS	7.87	0.394	0.350
	—	OPUS	3.91	0.101	0.089
	—	OPUS	1.98	0.045	0.040
FORESEE [15]	[9]	zfp-fp fixed accuracy	15.00	—	0.150[1]
	—	zfp-fp fixed accuracy	14.00	0.658	0.272
	[9]	zfp-fp fixed accuracy	8.00	—	0.050[1]
	—	zfp-fp fixed accuracy	9.85	0.483	0.181
	[9]	zfp-fp fixed accuracy	3.70	—	0.000[1]
	—	zfp-fp fixed accuracy	3.77	0.099	0.035
	—	flac*	1.58	0	0
	—	OPUS	18.10	0.532	0.346
	—	OPUS	7.87	0.340	0.194
	—	OPUS	3.81	0.166	0.092
	—	OPUS	1.98	0.134	0.054
PoroTomo [12]	—	flac*	1.95	0	0
	—	OPUS	15.66	0.262	0.213
	—	OPUS	7.88	0.136	0.104
	—	OPUS	3.95	0.098	0.070
	—	OPUS	1.91	0.094	0.064

[15] T. Zhu, J. Shen, and E. R. Martin, "Sensing earth and environment dynamics by telecommunication fiber-optic sensors: an urban experiment in Pennsylvania, USA," *Solid Earth*, vol. 12, pp. 219–235, Jan. 2021.

[16] FFmpeg Team, "FFmpeg." `https://ffmpeg.org/`, 2000.