

Understanding is Compression

Ziguang Li^{2,5,1†}, Chao Huang^{1,6†}, Xuliang Wang^{5,1†},
Haibo Hu^{1,6†}, Cole Wyeth³, Dongbo Bu^{1,5}, Quan Yu²,
Wen Gao², Xingwu Liu^{4,5*}, Ming Li^{3,5*}

¹Institute of Computing Technology, Chinese Academy of Science,
Beijing, China.

²Peng Cheng Laboratory, Shenzhen, China.

³School of Computer Science, University of Waterloo, N2L 3G1,
Waterloo, Ontario, Canada.

⁴School of Mathematical Sciences, Dalian University of Technology,
Dalian, China.

⁵Central China Institute of Artificial Intelligence, Zhengzhou, China.

⁶Ningbo Institute of Artificial Intelligence Industry, Ningbo, China.

*Corresponding author(s). E-mail(s): liuxingwu@dlut.edu.cn;
mli@uwaterloo.ca;

[†]These authors contributed equally to this work.

Abstract

Modern data compression methods are slowly reaching their limits after 80 years of research, millions of papers, and wide range of applications. Yet, the extravagant 6G communication speed requirement raises a major open question for revolutionary new ideas of data compression.

We have previously shown all understanding or learning are compression, under reasonable assumptions. Large language models (LLMs) understand data better than ever before. Can they help us to compress data?

The LLMs may be seen to approximate the uncomputable Solomonoff induction. Therefore, under this new uncomputable paradigm, we present LMCompress. LMCompress shatters all previous lossless compression algorithms, doubling the lossless compression ratios of JPEG-XL for images, FLAC for audios, and H.264 for videos, and quadrupling the compression ratio of bz2 for texts. The better a large model understands the data, the better LMCompress compresses.

Keywords: Lossless compression, large language models, Kolmogorov complexity, Solomonoff induction

1 Introduction

Before the reader starts to read this article, we invite you to reminisce about the everyday experiences: When you see a tiger, didn't you keep it in mind as a "large cat"? When you see 3.141592 ..., didn't you just note it down as a π ? When you see a bird, didn't you only focus on its unique features like size and color?

Yes, you have compressed the data! What makes you achieve this is not an ingeniously designed algorithm, but your understanding. Understanding makes compression, which is the motivation of this paper.

Data compression, either lossless or lossy, is an essential technology that underpins modern communication. Particularly, lossless compression allows the data to be perfectly reconstructed, which is indispensable for executable programs, text documents, genomics, cryptography, and multimedia archiving or production.

Numerous lossless compression methods have been developed, for example, ZIP, FLAC, PNG, and lossless H.264/H.265. These methods are largely confined to the information-theoretic framework established by C. Shannon over 80 years ago [1], relying on various frequency-based considerations or other computable properties (see Supplementary material for more details). These compression approaches, although being computationally tractable, have reached their limits after 80 years of research.

Dawn of profound transformation appears with the advent of large models. The principle of large models dates back to the well-known Solomonoff induction proposed in 1960's [2]. Rather than extracting computable features, large models approximate the uncomputable Solomonoff induction from a lot of data. They understand the data in this way, hence enabling efficient compression as we do in everyday experience.

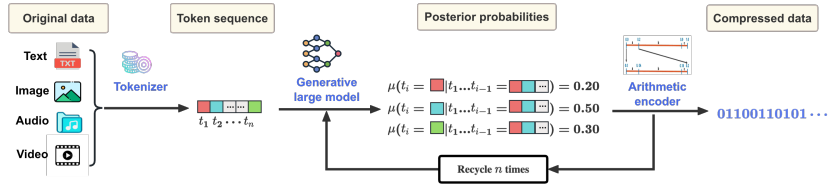


Fig. 1 The architecture of our LMCompress. First, the original data is transformed into a sequence of tokens. Then, this token sequence is fed into a generative large model, which outputs the predictive distribution for each token. Finally, arithmetic coding losslessly compresses the original data based on the predictive distributions. The tokenization module and the generative large model may vary according to the type of the data.

Based on the above observation, we advocate a new paradigm of compression, using large models to understand and consequently compress various data (see Fig. 1 for an overview). A precursor of our work has been independently published by us [3] and by a DeepMind team [4], with preceding work in [5] with similar ideas. These works demonstrated that arithmetic coding with a generative large model can improve the best traditional text compressors such as gzip a few folds.

This paper intends to comprehensively justify the idea that better understanding implies better compression with a focus on lossless compression for clean comparisons. To better understand the data with various formats, we use image GPT rather than a plain large language model (LLM) for images and videos, retrain an LLM with a small amount of audio data for audios, and employ domain-specific finetuned LLMs for domain texts. Based on the acquired understanding of data, we next apply arithmetic coding (see Supplementary material for details) to compress them. Lossless compression experiments show that by using LLM to understand data, we significantly improve compression ratios on all types of data, including texts, images, videos, and audios. Our method is several folds better than traditional algorithms, and a large margin better than the plain LLMs otherwise.

2 Methods

Traditional compression methods, whether lossy or lossless, depend on a *computable* function for characterizing data. Here, we propose LMCompress, a new Kolmogorov paradigm of compression rooted at the *uncomputable* Solomonoff induction. The Solomonoff induction is approximated by large models with never ending input data. The compression ratio should go up with better approximation of Solomonoff induction and better understanding of data.

It turns out that we have already passed the critical point as data accumulate and models improve. We demonstrate that LMCompress improves lossless compression of texts, images, videos and audios by several folds, far beyond the traditional methods.

The process of LMCompress is illustrated in Fig. 1. First, we decompose the original data into a sequence of tokens. Then, we feed this token sequence into a large generative model, which outputs the predictive distribution for each token. Finally, we use arithmetic coding to losslessly compress the original data based on these predictive distributions. To better understand the data of various formats, we use different tokenization methods and large generative models for different data types, which are described in more detail as follows.

2.1 Image Compression

We use the image-GPT model (iGPT, [6]) as the generative large model for images. Our choice of iGPT is driven by two key factors.

First, iGPT is a large-scale vision model that has been trained on a vast corpus of images, equipped with a thorough understanding of visual data. This makes iGPT well-suited for analyzing and processing images.

Second, iGPT is an autoregressive large vision model. When presented with a sequence of pixels, it can generate predictive probability for each pixel in the sequence. This capability is a prerequisite for arithmetic coding.

To compress an image using iGPT, we first concatenate the image’s pixels from top row to bottom row, transforming the two-dimensional visual data into a one-dimensional sequence of pixels. This pixel sequence is then fed into iGPT, yielding next-pixel probability for each pixel, based on which the image is compressed by using arithmetic coding.

The entire pixel sequence, however, cannot be fed into the model all at once due to the limited context window of iGPT model. In this case, we divide the sequence into non-overlapping segments, each of which can fit within iGPT’s context window. These individual segments are then fed into iGPT and compressed independently.

2.2 Video Compression

2.2.1 Lossless Video Compression

To the best of our knowledge, all existing open-source large video models do not naturally output probabilities. To circumvent this limitation, we have opted to leverage the image-based generative model iGPT instead. Since a video is fundamentally a sequence of frames, we propose to regard each frame as an image and compress the video frame-by-frame using iGPT as in Section 2.1.

At this stage, we have chosen not to exploit the inter-frame information for compression due to the following two concerns:

First, many types of videos, such as action movies, exhibit drastic changes from one frame to the next. In such cases, attempting to leverage information from previous frames is unlikely to be effective for compressing the current frame.

Second, even for the videos with relatively modest inter-frame variations, such as classroom lecture recordings, we have found that utilizing the inter-frame information does not actually improve the overall compression performance. This may be because the iGPT model is already able to sufficiently understand and model each individual frame on its own.

By compressing each video frame independently using iGPT, we can sidestep the challenge posed by the lack of large autoregressive video models. This frame-by-frame compression approach allows us to harness the powerful image understanding capabilities of iGPT, thus exempting the need to address the complexities of modeling temporal dependencies between video frames.

2.2.2 Lossy Video Compression

Lossy compression is the main stream in video compression as loss is acceptable or even unavoidable in most scenarios of video application. Numerous techniques have been proposed in this line, say DCVC series [7] and H.26X series. Basically, all the traditional methods compress videos by removing intra- and inter-frame redundancies. Recently, the development of Artificial Intelligence Generated Content has inspired a new direction, namely, not only removing redundancies to compress, but also generating details to help reconstruction. This is pioneered by “generative compression” [8], and has attracted much attention in image compression[9, 10].

In this study, we extend the “generative compression” idea to handle videos with an inspiration from [11], which proposed a transform-coding based method linked with generative large model. Specifically, we use DCVC results as prior and sample from the diffusion model DDPM. The sampling process is modeled as an inverse problem by subtracting the squared error between each frame generated by the diffusion model and the DCVC-decoded frame in each iteration, similar to [12]. Because gradient of

DCVC is necessary, we use the proxy loss function rather than the quantization step with additive uniform noise [13]. More details are provided in Supplementary material.

2.3 Audio Compression

Since audio is a type of sequential media, it is natural to leverage the large autoregressive models in compression. However, state-of-the-art open-source large audio models tend to discretize audio, which is inevitably lossy. To achieve lossless compression, we propose a model that handles audio at the signal level without discretization.

The basic idea is to treat audio as a sequence of frames. Each frame contains the amplitude information at a specific time point and can be represented by a constant number of bytes. We then map each byte into an ASCII character, effectively transforming the audio into a string of characters.

What we need is an autoregressive model that can understand this audio-as-string representation. To this end, we implement such a model by adding a low-rank adaptation layer to a large language model and subsequently fine-tuning the model on the acquired audio-as-string data. In this way, the fine-tuned model is able to estimate next-token probabilities for the audio string, which enables us to compress the audio using arithmetic coding.

Note that when the LLM has a limited context window, the audio string will be compressed piece-by-piece, as in Section 2.1.

2.4 Text Compression

Large language models have demonstrated impressive capability in compressing general texts [3, 4]. Intriguingly, they have potential to achieve even better compression ratios, provided that the texts to be compressed are restricted to specific domains.

The key lies in adapting LLM to better understand the target domain. This is implemented by incorporating an adaptation layer and fine-tuning the LLM via domain-specific texts, tailoring the model to the characteristics of the domain.

Then, to compress a text in a specific domain, we feed the text into the fine-tuned LLM. The LLM will estimate the next-token probabilities for the text, which can be leveraged by arithmetic coding to perform domain-specific compression. Again, when the LLM has a limited context window, the text will be compressed piece-by-piece, as in Section 2.1.

3 Results

We use compression ratio as the metric of compression performance, which refers to the ratio of the size of the original data to that of the compressed data. In general, the bigger the compression ratio, the better the compression performance.

Most of the baselines, say H.264 video compression standard, can work in both lossy and lossless modes. For fair comparison, all the baselines are set to their lossless modes in our experiments, except for DCVC in lossy video compression.

In addition, the compression ratio of an algorithm varies across datasets, so every lossless compression algorithm is evaluated on two different datasets in order to mitigate the effect brought about by potential data biases.

3.1 Image Compression

Dataset We evaluate the image compression performance of LMCompress on two benchmark image datasets, including *i*) ILSVRC2017 [14], a large-scale dataset containing millions of labeled images across thousands of categories, derived from the ImageNet corpus, and *ii*) CLIC2019 professional [15], which is specifically designed for evaluating image compression algorithms. CLIC2019 contains high-quality images with diverse characteristics such as natural scenes, textures, patterns, and structures, representative of real-world photography, multimedia, and visual content scenarios.

Since the datasets are too large, we sample 197 images from them. The total size of the images is 128 megabytes. Each image has three channels, corresponding to red, green, and blue, respectively, which are compressed separately. For each channel, we concatenate the rows of the image into a sequence, every 1024-pixel segment of which is fed into iGPT. Here the number 1024 is chosen to fit the context window of iGPT.

Table 1 shows the compression ratios of the baselines and LMCompress on the two datasets. The results demonstrate that our LMCompress significantly outperforms all the baselines on both datasets, more than doubling the compression ratios. Note that Chinchilla is also a family of large models. LMCompress shows better performance than Chinchilla, possibly because Chinchilla is only trained on language corpus while LMCompress is trained on image corpus hence can understand images better.

Table 1 Image compression ratios of state-of-the-art compressors and our LMCompress. Datasets: CLIC2019 and ISLVR

Method	CLIC2019	ISLVR
PNG	2.205 ¹	1.67
JPEG-XL	2.93 ¹	1.90
WebP	2.75 ¹	2.04
JPEG-2000	2.73 ¹	1.53
Chinchilla 7B	\	1.82 ²
Chinchilla 70B	\	2.08 ²
LMCompress	6.32	4.79

¹These results are from [16]

²These results are from [4]. Results on CLIC2019 are not presented since neither such results nor Chinchilla models are publicly available.

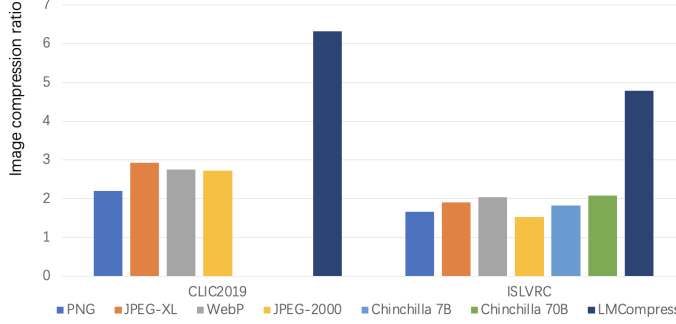


Fig. 2 Image compression ratios

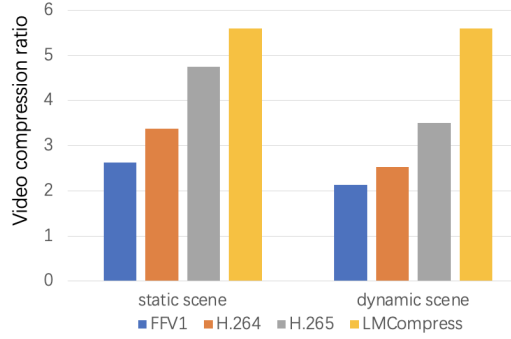


Fig. 3 Lossless video compression ratios of the state-of-the-art approaches and our LMCompress. Dataset: Xiph.org videos classified into “static scene” and “dynamic scene”

3.2 Video Compression

3.2.1 Lossless Video Compression

Datasets We use test data from Xiph.org, which has over 1000 video clips in the uncompressed YUV4MPEG format. Since the whole dataset is too massive, we sample 10 video clips as test data: 5 of static scenes and 5 of dynamic scenes. A static scene means that the consecutive frames change slightly or not change at all, for example, classroom recordings. The total size of static-scene videos is 162 megabytes. On the contrary, a dynamic scene means that the frames change drastically, for example, motion movies. The total size of dynamic-scene videos is 237 megabytes.

As shown in Fig. 3, LMCompress outperforms both baselines on both video types. On static scenes, LMCompress achieves over 20% improvement in compression ratio compared to the baselines. Even on dynamic scenes, LMCompress maintains its edge, achieving at least 50% improvement in compression ratio. We further observe that dynamic scenes are harder to compress than static scenes. A possible reason is that

in a dynamic-scene video, the actors tend to be in transient postures which are hard to understand or predict.

3.2.2 Lossy Video Compression

Datasets We utilize CIPR SIF Sequences from Xiph.org as our evaluation dataset for LMCompress. However, due to the limitations of the diffusion model’s resolution, we rescale the video size to 256×256 .

Metrics In addition to the primary metric of compression ratio, we also adopt three widely used metrics to assess video compression quality. These metrics include Peak-Signal-Noise-Ratio (PSNR) for measuring distortion, bits per pixel (bpp) for bitrate, and Fréchet Inception Distance (FID) for perceptual quality.

We follow the diffusion training setting and hyperparameters outlined in [12], which employs stochastic gradient descent for optimizing intermediate samples. The forward measurement operator utilized is the same as in DCVC [7]. We choose ELIC [17] as the I-frame compressor.

Table 2 Lossy video compression performance of the state-of-the-art approaches and our LMCompress. Dataset: CIPR SIF Sequences

	Compression ratio	bpp	PSNR \uparrow	FID \downarrow
DCVC	162	0.0945	29.0	153
DCVC-FM	269	0.0569	31.8	
LMCompress	582	0.0263	32.3	81

As demonstrated in Table 2, LMCompress exhibits superior performance compared to DCVC and DCVC-FM. Notably, LMCompress more than doubles the compression ratio, while keeping the other metrics as good or even better.

3.3 Audio Compression

Dataset We use LibriSpeech ASR corpus[18] and LJSpeech [19] as datasets to test audio compression. Both datasets are collected from the LibriVox project which covers nearly 1000 hours of 16 kHz English speech in audiobooks. Since the datasets are too large, we extract the first Gigabytes from the *train-clean-100* split of the LibriSpeech corpus and the first 256 Megabytes from LJSpeech.

To be processed by the LLMs in our experiment, each audio clip is transformed into a string of ASCII characters. Specifically, we right shift one bit for every byte in the frames so that each byte is a valid ASCII character. The discarded bits due to the shifts are stored independently. The strings are then divided into 2048-byte chunks, each of which is fed into the LLMs and compressed separately. Here, the number 2048 is chosen to fit in the context window of the LLMs.

Model fine-tuning We build LMCompress for audio compression by conducting supervised LoRA[20] fine-tuning on the LLaMA series model[21] LLaMA3-8B.

Note that LLaMA3-8B was pretrained on normal texts. To tailor it for audio compression, we fine-tune it on a corpus consisting of ASCII character strings derived from audio clips. For this end, we choose the first 64 megabytes in the *dev-clean* split of LibriSpeech as training data, with rank 8 and alpha 32.

Table 3 Audio compression ratios.
Dataset: LibriSpeech and LJSpeech

Method	LibriSpeech	LJSpeech
FLAC	3.23	3.21
LLaMA3-8B	4.45	4.02
Chinchilla-7B	4.24 ¹	\
Chinchilla-70B	4.76 ¹	\
LMCompress	6.07	6.22

¹These results are from [4]. Results on LJSpeech are not presented since neither such results nor Chinchilla models are publicly available.

The results are shown in Table 3. We obtained two observations: *i*) Large-model based methods outperform the state-of-the-art traditional method FLAC by 25%-94%, and *ii*) LMCompress, which was fine-tuned on audio corpus, outperforms other large-model based methods, with margins from 28% to 55%. Surprisingly, even though fine-tuned on the dataset LibriSpeech only, LMCompress improves the compression ratio of the raw LLaMA3-8B on LJSpeech by 55%.

3.4 Text Compression

Dataset Our benchmarks for domain-aware text compression are the MeDAL[22] and the Pile of Law[23]. MeDAL is a dataset in the domain of medicine. It is created from PubMed abstracts which are released in the 2019 annual baseline and primarily serves as a corpus for medical abbreviation understanding. On the other hand, Pile of Law is a dataset in the domain of law. It includes legal and administrative texts compiled from 35 sources.

In the experiments, we extract the first 1104 Megabytes from MeDAL and the *eurlerx* split from the Pile of Law corpus. Again, we divide the texts into segments of 2048 bytes so that every segment fits the context window of the large language models.

Model fine-tuning We build LMCompress for domain-aware text compression by fine-tuning LLaMA3-8B via supervised LoRA. For either domain dataset, we use the first 64 Megabytes for training, the next 16 Megabytes for validation, and the rest for testing.

Fig. 4 suggests that LMCompress outperforms all the baseline approaches. Its compression ratio on either dataset almost triples those of the best traditional methods. Compared to raw LLaMA3-8B, LMCompress improves the compression ratio by 8.5% on MeDAL and by 38.4% on Pile of Law.

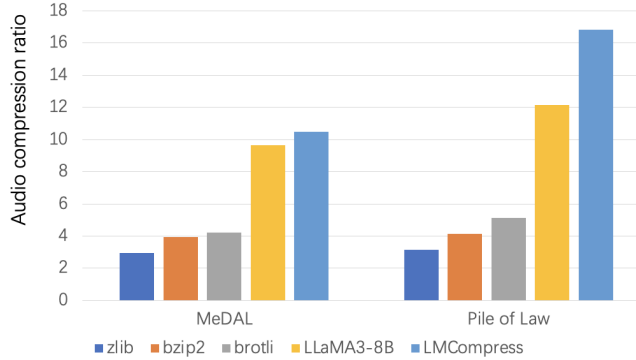


Fig. 4 Text compression ratios. Dataset: MeDAL and Pile of Law. LLaMA3-8B means the text compressor in [3] with LLaMA2-7B replaced by LLaMA3-8B

In summary, LMCompress has higher lossless compression ratios on various media than all traditional baselines and raw LLM-based algorithms. This evidence supports our claim that better understanding leads to better compression.

4 Conclusion

Communication in the past was generally governed by the Shannon paradigm, with coding efficiency upper bound by Shannon entropy. While exploring other computable features can further improve compression, large models may be seen to approximate the uncomputable Solomonoff induction, hence opening a new Kolmogorov paradigm of compression. As we have shown, this new way of lossless compression has achieved substantial improvements on various kinds of data. This new paradigm allows us to systematically understand the data we transmit, shattering the Shannon entropy upper bound in a great scale.

Our work sheds light on the 6G communication, especially when the bandwidth is limited from the satellites. It will be significantly benefited by understanding the data, with large models at both ends of communication to encode and decode. As the large models are specialized as agents, assisted with Retrieval-Augmented Generation, AI will understand the data to be transmitted much better. When the data need to be encrypted, our compression needs to be done before encryption. One can even imagine that the sides with superior models broadcast openly compressed messages, allowing only those with equal models to decipher as a first level of encryption, at no extra cost.

Though this paper is mainly on lossless compression, our work on lossy video compression indicates that the research presented here can be extended to the domain of lossy compression. Another future direction is to incorporate inter-frame information in lossless video compression.

Acknowledgements. This research is partially supported by Canada’s NSERC grant OGP0046506, and Canada Research Chair Program. We thank Nick Zhang and Paul Vitanyi for discussions on Solomonoff induction. We thank Cynthia Huang,

Yuqing Xie, Zhiying Jiang, Rui Wang, and Peijia Guo for their discussions and related work in [24] and [3].

Data availability. LSCRCV 2012 is available at <https://www.image-net.org/challenges/LSVRC/2012/index.php>. CLIC is available at <https://clic.compression.cc/2019/>. Librispeech is available at www.openslr.org/12. LJSpeech is available at <https://keithito.com/LJ-Speech-Dataset>. MeDAL is available at <https://github.com/McGill-NLP/medal>. Eurlax is available at <https://huggingface.co/datasets/pile-of-law/pile-of-law>. CIPR SIF Sequence (foreman) is available at <https://media.xiph.org/video/derf/>.

Code availability. The code has been uploaded to Code Ocean, and will be publicly available at Github after the manuscript is accepted.

References

- [1] Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal **27**(3), 379–423 (1948)
- [2] Solomonoff, R.: A formal theory of inductive inference. Inform. control **7**(1), 1–22 (1964)
- [3] Huang, C., Xie, Y., Jiang, Z., Lin, J., Li, M.: Approximating human-like few-shot learning with gpt-based compression. arXiv preprint arXiv:2308.06942 (2023)
- [4] Delétang, G., Ruoss, A., Duquenne, P.-A., Catt, E., Genewein, T., Mattern, C., Grau-Moya, J., Wenliang, L.K., Aitchison, M., Orseau, L., et al.: Language modeling is compression. arXiv preprint arXiv:2309.10668 (2023)
- [5] Bellard, F.: NNCP v2: Lossless data compression with transformer (2021). <https://bellard.org/nncp/nncp-v2.pdf>
- [6] Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., Sutskever, I.: Generative pretraining from pixels. In: International Conference on Machine Learning, pp. 1691–1703 (2020). PMLR
- [7] Li, J., Li, B., Lu, Y.: Deep contextual video compression. Advances in Neural Information Processing Systems **34**, 18114–18125 (2021)
- [8] Santurkar, S., Budden, D., Shavit, N.: Generative compression. In: 2018 Picture Coding Symposium (PCS), pp. 258–262 (2018). IEEE
- [9] Yang, R., Mandt, S.: Lossy image compression with conditional diffusion models. Advances in Neural Information Processing Systems **36** (2024)
- [10] Relic, L., Azevedo, R., Gross, M., Schroers, C.: Lossy image compression with foundation diffusion models. arXiv preprint arXiv:2404.08580 (2024)

- [11] Xu, T., Zhu, Z., He, D., Li, Y., Guo, L., Wang, Y., Wang, Z., Qin, H., Wang, Y., Liu, J., et al.: Idempotence and perceptual image compression. arXiv preprint arXiv:2401.08920 (2024)
- [12] Chung, H., Kim, J., Mccann, M.T., Klasky, M.L., Ye, J.C.: Diffusion posterior sampling for general noisy inverse problems. arXiv preprint arXiv:2209.14687 (2022)
- [13] Ballé, J., Laparra, V., Simoncelli, E.P.: End-to-end optimized image compression. arXiv preprint arXiv:1611.01704 (2016)
- [14] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., *et al.*: Imagenet large scale visual recognition challenge. International journal of computer vision **115**, 211–252 (2015)
- [15] CLIC: Workshop and challenge on learned image compression (2019). <https://clic.compression.cc/2019/>
- [16] Rhee, H., Jang, Y.I., Kim, S., Cho, N.I.: Lc-fdnet: Learned lossless image compression with frequency decomposition network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6033–6042 (2022)
- [17] He, D., Yang, Z., Peng, W., Ma, R., Qin, H., Wang, Y.: Elic: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5718–5727 (2022)
- [18] Panayotov, V., Chen, G., Povey, D., Khudanpur, S.: Librispeech: An asr corpus based on public domain audio book. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2015)
- [19] Ito, K., Johnson, L.: The LJ Speech Dataset (2017). <https://keithito.com/LJ-Speech-Dataset/>
- [20] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. In: ICLR 2022 (2022)
- [21] Touvron, H., Lavril, T., Izacard, G., Lachaux, X.M.M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G.: LLaMA: Open and Efficient Foundation Language Models (2023). <https://research.facebook.com/file/1574548786327032/LLaMA--Open-and-Efficient-Foundation-Language-Models.pdf>
- [22] Wen, Z., Lu, X.H., Reddy, S.: Medal: Medical abbreviation disambiguation dataset for natural language understanding pretraining. arXiv preprint arXiv:2012.13978 (2020)

- [23] Henderson, P., Krass, M., Zheng, L., Guha, N., Manning, C., Jurafsky, D., Ho, D.E.: Pile of law: Learning responsible data filtering from the law and a 256gb open-source legal dataset (2022) <https://doi.org/10.48550/arXiv.2207.00220>
- [24] Jiang, Z., Wang, R., Bu, D., Li, M.: A theory of human-like few-shot learning. arXiv preprint arXiv:2301.01047 (2023)