OmniZip: Learning a Unified and Lightweight Lossless Compressor for Multi-Modal Data

Supplementary Material

The supplementary material includes deeper explanations of the proposed method and extended experiments, covering: (1) detailed descriptions of the modality-unified tokenization, modality-routing context learning, modality-routing feedforward design, and reparameterization training strategy; (2) implementation details of multi-modal datasets and training/testing process; (3) extended lossless compression performance using adjusted bits/Byte as the metric.

1. Backbone Selection

We compare three common backbone architectures for the probability prediction model: Transformer [28], Mamba [16], and RWKV [23]. These models are evaluated in terms of compression efficiency (bits/Byte), computational cost (MACs), and inference speed (KB/s).

As for Transformer, We use a causal, decoder-only Transformer implemented in JAX/Haiku. Input tokens are embedded, scaled, and summed with sinusoidal positional encodings, then processed through a stack of eight multi-head self-attention (8 heads) and feed-forward layers, each followed by residual connections and LayerNorm. For Mamba and RWKV, we adopt their official implementations[1][2]. The trained models are converted into CoreML packages and benchmarked on a MacBook Pro 2024 with an Apple M4 chip using CPU inference.

2. Modality-Unified Tokenization

To enable multi-modal lossless compression, we design a reversible modality-unified tokenization strategy that maps heterogeneous data into a shared token space. The main challenge arises from the fundamentally distinct structures of different modalities. Moreover, to ensure lossless reconstruction, the tokenization process must be fully invertible.

For text-like data, including natural language, gene sequences, and databases, we follow [32] and adopt a SentencePiece BPE tokenizer [27] with a vocabulary size of 16K. To better adapt to domain-specific characteristics, we extend the vocabulary with symbolic tokens: like nucleotide bases (A, T, G, C) for gene sequences, and common SQL keywords (e.g., SELECT, FROM, WHERE) for databases. Formally, given a text string S, the tokenizer performs:

$$S_{\text{text}} \xrightarrow{\text{SPM BPE}} [x_1, x_2, \ldots, x_n], \quad x_i \in \mathcal{V}_{\text{text}}, \ |\mathcal{V}_{\text{text}}| = 16\text{K}. \tag{1}$$

[1]Mamba: https://github.com/state-spaces/mamba.
[2]RWKV: https://github.com/BlinkDL/RWKV-LM.

For image-like data, including natural, medical, and tactile images, we partition each image into $16 \times 16 \times 3$ patches to preserve local spatial correlations. Pixels within each patch are flattened in raster-scan order, and each RGB channel is expanded sequentially: $[R_1, G_1, B_1, R_2, G_2, B_2, \ldots]$. We treat each 8-bit sub-pixel as an independent token (i.e., $|\mathcal{V}_{\text{image}}| = 256$), thereby maintaining inter-pixel and inter-channel correlations. For grayscale medical images, each 8-bit intensity value is directly mapped to a single token. Tactile data are handled in two ways: visual tactile maps are processed as regular RGB images, while 3D tactile force vectors (x, y, z) are linearly mapped into pseudo-RGB triplets for tokenization.

Speech is continuous and hard to discretize without loss. Hence, we adopt a next-byte prediction scheme, reading the raw byte stream and treating each byte as a token:

$$S_{\text{speech}}(t) \xrightarrow{\text{each byte}} [x_1, x_2, \ldots, x_n], \quad x_i \in [0, 255], \tag{2}$$

yielding a 256-size vocabulary shared with image-like modalities (i.e., $|\mathcal{V}_{\text{speech}}| = 256$).

After tokenization, we merge all modality vocabularies into a single unified token space:

$$\mathcal{V}_{\text{uni}} = \mathcal{V}_{\text{text}} \cup \mathcal{V}_{\text{image}} \cup \mathcal{V}_{\text{speech}}. \tag{3}$$

Each sequence is prefixed with a modality identifier token, i.e., <image>, <medical>, <tactile>, <text>, <gene>, <database>, or <speech>, allowing the model to learn modality-aware priors during probability estimation.

Before softmax and arithmetic coding, we apply modality-specific masking to restrict the active token propabilities and improve the compression efficiency. For example, during image compression, only tokens in $\mathcal{V}_{\text{image}}$ are retained, and all others are zeroed out:

$$p_{\text{image}}(x_i|x_{<i}) = \text{softmax}(o(x_i|x_{<i}) \odot M_{\text{image}}), \tag{4}$$

where M_{image} is a binary mask selecting image tokens, $o(x_i|x_{<i}$ is the output logit before softmax.

3. Modality-Routing Context Learning

As shown in Fig. 1, each RWKV block [23] consists of two components: a Time Mixing module and a multilayer perceptron (MLP) [26]. The Time Mixing module serves as a lightweight alternative to attention, modeling contextual dependencies through a recurrent formulation. Instead of computing pairwise attention, it maintains a running state that summarizes past tokens, allowing linear-time inference.

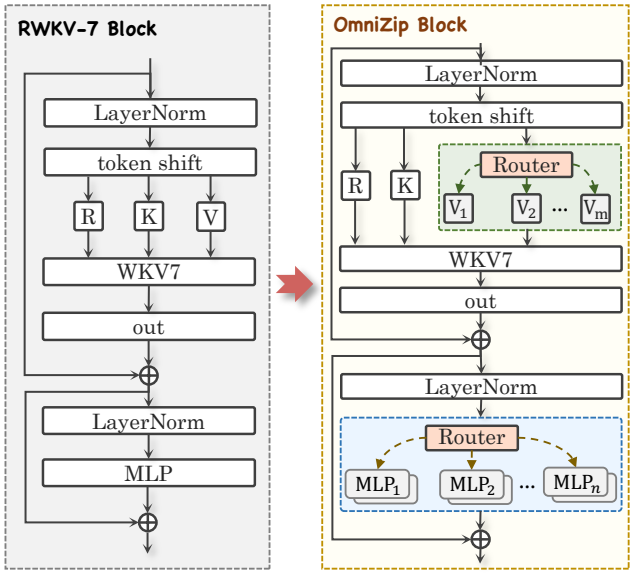

Figure 1. Key structural differences between RWKV-7 and OmniZip model. Left: an RWKV-7 block. Right: an OmniZip block.

As stated, different modalities exhibit distinct contextual dependencies. For instance, text are one-dimensional sequential and have mostly short-range semantic dependencies. Images are two-dimensional and flattened into sequences, with inter-pixel and inter-channel correlations. Speech is continuous and temporally structured.

To enhance multi-modal adaptability, we integrate a mixture-of-experts (MoE) mechanism [6] into the Time Mixing module. MoE dynamically selects specialized experts based on input tokens, enabling conditional computation and reducing unnecessary activation. Specifically, given an input token x_i, a router network predicts a score:

$$g_{i,e} = \text{softmax}(x_i \cdot W_g)_e = \frac{\exp(x_i \cdot W_{g,e})}{\sum_{e'=1}^{E} \exp(x_i \cdot W_{g,e'})}, \quad (5)$$

where E is the expert count, W_g is the routing projection. $g_{i,e}$ means the likelihood that expert e should process x_i.

To keep efficiency, we adopt top-k sparse routing, where only the experts with top-k highest scores are activated, and their outputs are aggregated as a weighted sum. We experiment with applying MoE to different components and find that the V layer benefits most from such adaptive routing. Conceptually, the K layer serves as a semantic index, and the R layer acts as a temporal gate, while the V layer encodes concrete contextual content. Allowing diversity in V layer enables the model to adapt more flexibly to modality-specific information. Hence, we apply MoE only to the V layer, sharing K and R layers across all V experts. Formally,

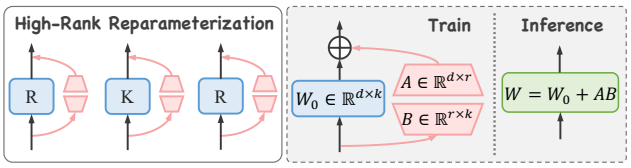

Figure 2. Illustration of the reparameterization training strategy.

the V-projection output in our MoE design is computed as:

$$\text{V}(x_i) = \sum_{e \in \text{top-}k} \hat{g}_{i,e}\, e(x_i), \quad (6)$$

where each expert e is a distinct V projection layer, and $\hat{g}_{i,e}$ denotes the re-normalized routing score.

To ensure model compactness, we use $E = 4$ experts and $k = 2$. This setup increases the number of parameters by only three additional V layers per Time Mixing module, yet notably improves the multi-modal adaptability.

4. Modality-Routing Feedforward

In the RWKV backbone [23], each block integrates a feedforward MLP that projects intermediate representations into a higher-dimensional space, applies nonlinear transformations, and fuses contextual information from the Time Mixing module. This structure enhances model expressiveness but treats all tokens uniformly, regardless of their modality. In fact, a single shared MLP is insufficient to capture the distinct properties of heterogeneous modalities.

To better support multi-modal compression, we replace this MLP with a modality-routing feedforward module based on MoE. The routing mechanism follows the design in Sec. 3, while the experts here are small MLPs instead of V layers. Each MLP expert adopts a hidden factor of $2\times$, half that of the original large MLP. With four experts and top-$k = 2$, the number of activated parameters per token remains nearly identical to the original design.

5. Reparameterization Training Strategy

Following [32], we adopt a high-rank reparameterization training strategy to enhance OmniZip's compression capacity without increasing inference complexity. During training, each R, K, and V projection layer in the Time Mixing module is augmented with an auxiliary high-rank branch to improve representational power. At inference, these auxiliary branches are merged back into the main path through structural reparameterization, maintaining a compact single-path design, as illustrated in Fig. 2.

Specifically, instead of using parallel branches like 1×1 convolutions or shortcuts as in [14], we follow the strategy in [32] and reparameterize each branch as a product of two high-rank matrices: $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times k}$, where the

Table 1. Details of the utilized multi-modal datasets (16 datasets from 7 modalities).

Dataset	Type	Train	Test	Description
Kodak [12]	image	✗	✓	24 film-scanned photos (768×512) with balanced natural and synthetic scenes. We use it for testing.
CLIC-P [9]	image	✗	✓	41 high-resolution DSLR images (mostly 2K) with rich color and texture diversity. We use it for testing.
CLIC-M [9]	image	✗	✓	61 smartphone photos (mostly 2K) containing real-world noise and artifacts. We use it for testing.
DIV2K [1]	image	✓	✓	900 2K-resolution images with a large diversity of contents. We follow the official split (800 train, 100 test).
Axial [5]	medical	✓	✓	2D knee MRI slices (256×256) in the axial plane. We follow the official split ($\sim$38K train, $\sim$4.1K test).
Coronal [5]	medical	✓	✓	2D knee MRI slices (256×256) in the coronal plane. We follow the official split ($\sim$33K train, $\sim$3.5K test).
Sagittal [5]	medical	✓	✓	2D knee MRI slices (256×256) in the sagittal plane. We follow the official split ($\sim$34K train, $\sim$3.6K test).
TouchandGo [30]	tactile	✓	✓	A tactile dataset of 140 trajectories (640×480), with 60% for training and the rest for testing.
ObjectFolder [15]	tactile	✓	✓	A tactile dataset of 1000 trajectories (120×160), with 60% for training and the rest for testing.
enwik8 [10]	text	✓	✗	The first 100MB of the English Wikipedia dump. We use it for testing.
enwik9 [11]	text	✗	✓	The first 1GB of the English Wikipedia dump. We use it for testing.
Gutenberg [25]	text	✓	✓	A library containing 75K eBooks. We randomly select 2K and 1K books for training and testing, respectively.
GenoSeq [8]	gene	✓	✓	17 genomics sequencing dataset with FastQ format. We use 10 for training and 7 for testing.
DNACorpus [24]	gene	✓	✓	DNA sequences of 15 species, we follow the official split (370MB for training and 246MB for testing).
Spider [31]	database	✓	✓	5693 SQL queries on 200 databases. We follow the official split (314MB for training and 60MB for testing).
WikiSQL [17]	database	✓	✓	24241 SQL queries from Wikipedia. We follow the official split (38MB for training and 12MB for testing).
LibriSpeech [22]	speech	✓	✓	A 1000-hour English speech dataset. We follow the official split (1GB for training and 600MB for testing).

main-path's weight is $W_0 \in \mathbb{R}^{d\times k}$ and $r \gg d, k$. During training, the layer's output is the sum of the main branch and the bypass branch. These branches are merged at inference via structural reparameterization: $W = W_0 + A \times B$, resulting in a single-path structure that reduces both runtime and memory usage. Importantly, although the branches are merged during inference, the learned multi-branch parameters are preserved, maintaining high capability.

In OmniZip, this strategy is applied to all the R/K/V layers in the Time Mixing module. Following [32], we set the decomposition rank r to $4\times$ the embedding dimension.

6. Multi-Modal Datasets

We conduct experiments across seven data types, including natural images, medical images, tactile signals, text, gene sequences, databases, and speech, covering a total of 16 datasets for training and evaluation, as shown in Tab. 1.

To ensure balanced and stable multi-modal training, we normalize the data scale across modalities by restricting each modality's training set to approximately 1 GB. For large datasets, samples are randomly drawn to maintain representative coverage of content diversity, while smaller datasets are augmented through simple transformations such as random cropping/flipping, token shuffling, or noise perturbation (depending on the modality type).

During training, we also adopt a balanced batch sampling strategy to prevent overfitting toward some modalities. Specifically, the dataloader cycles through all modalities in a round-robin manner, dynamically constructing each batch by uniformly sampling from the shuffled indices of every modality. This ensures that each batch contributes an equal number of samples per modality, allowing the model to learn modality-invariant patterns.

Regarding the model input format, as stated in Sec. 2, for image-like data, each input corresponds to a flattened $16 \times 16 \times 3$ patch, resulting in a sequence length of 768 tokens. Text-like and speech data are processed as sequential streams, with each input sequence containing 1024 tokens.

7. Training and Testing Process

To accelerate the overall pipeline, we incorporate optimizations like Cython [3], Numba [19], and distributed parallelization. For model evaluation and deployment, we test OmniZip across three hardware platforms: (1) NVIDIA A100 GPU (for training and high-throughput testing), (2) MacBook Pro 2024 (Apple M4 CPU), and (3) iPhone 17 Pro (A19 NPU). For CPU and NPU evaluation, we convert the trained PyTorch models into CoreML packages [18] and conduct profiling via Xcode Instruments. For GPU evaluation, we switch the GPU to performance mode and ensure no other processes are running on the device. We then measure inference latency by executing the model 1000 times consecutively and report the average runtime.

8. Adjusted Multi-Modal Compression Results

To provide a more realistic evaluation of deployment efficiency, we adopt the adjusted bits-per-Byte metric, which incorporates model size into the compressed data size. It penalizes large models and reflects the trade-off between compression performance and model storage overhead.

As shown in Tab. 2 and Tab. 3, OmniZip-S achieves superior adjusted compression efficiency across all seven modalities, including image-like, text-like, and speech

Table 2. Adjusted compression performance (bits/Byte) on image-like (natual image, medical image, and tactile) datasets. $*$ denotes pretrained LLMs, † indicates that some results are reprodeced by us. Other values are taken from their papers.

Compressor	#Params↓	Multi Modal	Adjusted bits/Byte↓								
			Kodak	CLIC-P	CLIC-M	DIV2K	TouchandGo	ObjectFolder	Axial	Coronal	Sagittal
Llama3*† [13]	8B	✓	8490321	971320	553618	291465	322642	214197	693352	750368	970961
RWKV*† [13]	7B	✓	7429031	849906	484417	255032	282312	187423	606683	656573	823341
DLPR*† [2]	22.3M	✗	23670	2710	1545	815	900	601	-	-	-
L3C*† [21]	5M	✗	5310	610	349	185	203	138	439	473	594
P2LLM [7]	8B	✗	8490319	971318	553616	291463	-	-	-	-	-
aiWave [29]	695M	✗	-	-	-	-	-	-	59372	64254	80574
BCM [20]	12.5M	✗	-	-	-	-	-	-	1088	1176	1475
OmniZip-S	4.8M	✓	5097	586	335	178	195	132	421	454	570
OmniZip-M	38M	✓	40332	4618	2632	1387	1534	1020	3298	3568	4475
OmniZip-L	152M	✓	161319	18458	10521	5540	6131	4072	13178	14261	17883

Table 3. Adjusted compression performance (bits/Byte) on text-like (natural language, gene sequence, database) and speech datasets. $*$ denotes pretrained LLMs, † indicates that some results are reprodeced by us. Other values are taken from their papers.

Compressor	#Params↓	Multi Modal	Adjusted bits/Byte↓						
			enwik9	Gutenberg	Spider	WikiSQL	GenoSeq	DNACorpus	LibriSpeech
Llama3*† [13]	8B	✓	129	362459	2125	10835	500	1138	212
RWKV*† [13]	7B	✓	113	317152	1859	9480	437	996	186
tszip [4]	169	✗	4	7658	46	230	13	26	-
L3TC [32]	12	✗	2	546	6	18	3	4	-
OmniZip-S	4.8M	✓	1	219	3	8	2	3	4
OmniZip-M	38M	✓	2	1723	11	52	4	7	5
OmniZip-L	152M	✓	3	6888	42	207	11	23	8

datasets. OmniZip-M and OmniZip-L also exhibit satisfactory balance between compression efficiency and deployability. Among modality-specific compressors, Mentzer et al. [21] and Zhang et al. [32] achieve competitive adjusted compression efficiency on image-like and text-like datasets, respectively. Though [13] demonstrate strong compression efficiency on text-like and speech datasets, their massive parameter sizes dominate the total compression cost, resulting in significantly higher adjusted bits-per-Byte values.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135, 2017. 3

[2] Yuanchao Bai, Xianming Liu, Kai Wang, Xiangyang Ji, Xiaolin Wu, and Wen Gao. Deep lossy plus residual coding for lossless and near-lossless image compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5): 3577–3594, 2024. 4

[3] Stefan Behnel, Robert Bradshaw, Craig Citro, Lisandro Dalcin, Dag Sverre Seljebotn, and Kurt Smith. Cython: The best of both worlds. *Computing in Science and Engineering*, 13 (2):31–39, 2011. 3

[4] Fabrice Bellard. ts_zip: Text Compression using Large Language Models. https://bellard.org/ts_zip/, 2023. Accessed: 2024-08-10. 4

[5] Nicholas Bien, Pranav Rajpurkar, Robyn L Ball, Jeremy Irvin, Allison Park, Erik Jones, Michael Bereket, Bhavik N Patel, Kristen W Yeom, Katie Shpanskaya, et al. Deep-learning-assisted diagnosis for knee magnetic resonance imaging: development and retrospective validation of mrnet. *PLoS medicine*, 15(11):e1002699, 2018. 3

[6] Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*, 2024. 2

[7] Kecheng Chen, Pingping Zhang, Hui Liu, Jie Liu, Yibing Liu, Jiaxin Huang, Shiqi Wang, Hong Yan, and Haoliang Li. Large language models for lossless image compression: Next-pixel prediction in language space is all you need. *arXiv preprint arXiv:2411.12448*, 2024. 4

[8] Clarivate Analytics. Geneseq™ database. https://clarivate.com/intellectual-property/patent-intelligence/geneseq/, 2024. Accessed: October 2025. 3

[9] CLIC. CLIC: Workshop and challenge on learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 3

[10] Wikipedia Community. enwik8: First 100m bytes of the english wikipedia dump. https://cs.fit.edu/~mmahoney/compression/enwik8.zip, 2006. Preprocessed version used for character-level language modeling benchmarks. 3

[11] Wikipedia Community. enwik9: Complete english wikipedia dump (xml). https://cs.fit.edu/~mmahoney/compression/enwik9.zip, 2007. Full Wikipedia XML dump preprocessed for large-scale language modeling. 3

[12] Eastman Kodak Company. Kodak lossless true color image suite. Internal Research Dataset, 1999. 24 uncompressed PNG images, 768x512 resolution. 3

[13] Gregoire Deletang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, Marcus Hutter, and Joel Veness. Language modeling is compression. In *The Twelfth International Conference on Learning Representations*, 2024. 4

[14] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13733–13742, 2021. 2

[15] Ruohan Gao, Yen-Yu Chang, Shivani Mall, Li Fei-Fei, and Jiajun Wu. Objectfolder: A dataset of objects with implicit visual, auditory, and tactile representations. *arXiv preprint arXiv:2109.07991*, 2021. 3

[16] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. 1

[17] Wonseok Hwang, Jinyeong Yim, Seunghyun Park, and Minjoon Seo. A comprehensive exploration on wikisql with table-aware word contextualization. *arXiv preprint arXiv:1902.01069*, 2019. 3

[18] Apple Inc. Core ml, 2023. Apple's framework for integrating machine learning models into apps. 3

[19] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: A llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, pages 1–6, 2015. 3

[20] Xiangrui Liu, Meng Wang, Shiqi Wang, and Sam Kwong. Bilateral context modeling for residual coding in lossless 3d medical image compression. *IEEE Transactions on Image Processing*, 33:2502–2513, 2024. 4

[21] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Practical full resolution learned lossless image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10629–10638, 2019. 4

[22] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015. 3

[23] Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Haowen Hou, Janna Lu, William Merrill, Guangyu Song, Kaifeng Tan, Saiteja Utpala, et al. Rwkv-7" goose" with expressive dynamic state evolution. *arXiv preprint arXiv:2503.14456*, 2025. 1, 2

[24] Diogo Pratas and Armando J Pinho. A dna sequence corpus for compression benchmark. In *International Conference on Practical Applications of Computational Biology & Bioinformatics*, pages 208–215. Springer, 2018. 3

[25] Project Gutenberg. Project gutenberg corpus. 2013. Available at: https://www.gutenberg.org/. 3

[26] Hassan Ramchoun, Youssef Ghanou, Mohamed Ettaouil, and Mohammed Amine Janati Idrissi. Multilayer perceptron: Architecture optimization and training. 2016. 1

[27] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units, 2016. Accessed: 2024-08-10. 1

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. Accessed: 2024-08-10. 1

[29] Dongmei Xue, Haichuan Ma, Li Li, Dong Liu, and Zhiwei Xiong. Aiwave: Volumetric image compression with 3-d trained affine wavelet-like transform. *IEEE Transactions on Medical Imaging*, 42(3):606–618, 2022. 4

[30] Fengyu Yang, Chenyang Ma, Jiacheng Zhang, Jing Zhu, Wenzhen Yuan, and Andrew Owens. Touch and go: Learning from human-collected vision and touch. *arXiv preprint arXiv:2211.12498*, 2022. 3

[31] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*, 2018. 3

[32] Junxuan Zhang, Zhengxue Cheng, Yan Zhao, Shihao Wang, Dajiang Zhou, Guo Lu, and Li Song. L3tc: Leveraging rwkv for learned lossless low-complexity text compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 13251–13259, 2025. 1, 2, 3, 4