The Lossy Horizon: Error-Bounded Predictive Coding for Lossy Text Compression (Episode I)

Anonymous Author(s)
Affiliation
Address
email

Abstract

Large Language Models (LLMs) can achieve near-optimal lossless compression by acting as powerful probability models. We investigate their use in the lossy domain, where reconstruction fidelity is traded for higher compression ratios. This paper introduces Error-Bounded Predictive Coding (EPC), a lossy text codec that leverages a Masked Language Model (MLM) as a decompressor. Instead of storing a subset of original tokens, the EPC allows the model to predict masked content and stores minimal, rank-based corrections only when the model's top prediction is incorrect. This creates a graceful residual channel that offers continuous rate-distortion control. We compare EPC to a simpler Predictive Masking (PM) baseline and a transform-based Vector Quantisation with a Residual Patch (VQ+RE) approach. Through a rigorous evaluation that includes precise bit accounting and rate-distortion analysis, We demonstrate that EPC consistently dominates PM, offering superior fidelity at a significantly lower bit rate by more efficiently utilising the model's intrinsic knowledge.

1 Introduction

I study large language models (LLMs) as practical lossy compressors, with an emphasis on post-training deployment. The core principle is that an MLM can act as a decompression engine: if a token is predictable from context, it need not be stored [1]. Building on this, We introduce and compare three families of lossy codecs. The first, Predictive Masking (PM), is a simple baseline that subsamples the token stream. The second, Error-Bounded Predictive Coding (EPC), refines this by storing minimal, rank-indexed corrections only when the model's top prediction fails. This provides a bit-efficient residual channel. The third, Vector Quantisation with a Residual Patch (VQ+RE), serves as a substantial baseline from the transform-coding paradigm, compressing latent states while guaranteeing bounded error. Our evaluation focuses on the rate (bits per character, BPC) versus distortion (fidelity) for each codec, with rigorous bit accounting for both payload and static model costs, framing the results against strong lossless compressors [2, 3].

2 Related Work

Using predictive models for compression is a classical idea [2, 4]. Recent work leverages LLMs with arithmetic coding to approach the entropy limits for lossless text compression [1, 5]. Our PM and EPC methods adapt this paradigm to a lossy setting, where an MLM provides a powerful conditional model for reconstruction. EPC's use of a rank-indexed residual stream is novel for text compression with LLMs, drawing inspiration from residual coding in other domains. VQ is a common technique for lossy compression of latent representations [6]; our VQ+RE baseline enhances it with an explicit error-correction layer, making it a competitive benchmark for text.

3 Methodology

I standardise the models, datasets, and metrics in all experiments.

Models and Datasets Primary models are MLMs: `bert-base-cased`, `RoBERTa` [7], and `DistilRoBERTa` [8]. The main corpus is `WikiText-103`. All datasets have non-overlapping train, validation, and test splits.

Metrics and Cost Accounting The rate is measured in BPC [1]. Distortion is measured by character-level fidelity (1 - error rate), ChrF, and BERTScore [9]. The amortised BPC accounts for the static size of the model, tokeniser, and any coders over N_{copies} uses.

Entropy coding details (AC/rANS). We report truthful entropy numbers: the Bernoulli flag stream and the K-way rank symbols are encoded with a standard range-ANS (rANS) using tabled frequencies, so the quoted bits are the exact code lengths produced by the coder. For the vocabulary-sized fallback token stream we report two quantities (when shown): (i) an *ideal* adaptive arithmetic lower bound $\sum_t -\log_2\big((c_{x_t}+\alpha)/(C+\alpha V)\big)$ under an online unigram with Laplace smoothing (α=1), and (ii) a practical rANS approximation in which, at each step, we encode a binary alphabet $\{s, \neg s\}$ with frequencies $\{c_s,\ C-c_s\}$ and then update counts (here c_s is the current count of symbol s, C the total count, and V the vocabulary size). This $\{s, \neg s\}$ trick is a valid ANS coder and slightly overestimates the ideal bound because the complement mass is merged; implementing a full per-step V-way CDF would close this small gap but is orthogonal to our contribution and does not affect the relative RD trends.

3.1 Compression Pipeline

The pipeline consists of three stages: model specialisation, compression, and reconstruction.

Stage 1: Model Specialisation. To prepare the MLM for high-rate, predictability-based masking, We use an adaptive curriculum. The training data is split into a `fine-tuning_set` (90%) and a disjoint `policy_set` (10%). Each epoch, token predictability (surprisal) is computed on the `policy_set` using the current model. This policy then dictates masking for the `fine-tuning_set`, on which the model is updated. The masking rate increases linearly from 0.2 to 0.8 over several epochs to stabilise training [10, 11].

Stage 2: Compression Codecs. I evaluate three lossy codecs built upon the specialised MLM:

- **Predictive Masking (PM).** Let $x_{1:N}$ be the token sequence. A subset of indices $\mathcal{M}$ are masked, while the remaining indices are kept and denoted by $\mathcal{S}$. We select $\mathcal{M}$ by identifying tokens with the lowest model surprisal, $s_i = -\log_2 q_\theta(x_i \mid X_{\setminus i})$, up to a target masking fraction p_{mask}. The payload consists of two parts: a bit vector indicating the positions of the kept tokens, and the kept tokens themselves, which are entropy-coded. Reconstruction involves deterministically infilling the masked positions $\{x_i\}_{i\in\mathcal{M}}$ with the most likely token predicted by the specialised MLM from Stage 1, i.e., $\arg\max q_\theta(\cdot \mid X_{\mathcal{S}})$.
- **Error-Bounded Predictive Coding (EPC).** EPC begins identically to PM by selecting a mask set $\mathcal{M}$. However, instead of discarding the original tokens at masked positions, it introduces a residual stream. For each masked position $i \in \mathcal{M}$, if the model's top-1 prediction is incorrect, the EPC stores a minimal correction. Let r_i be the rank of the true token x_i in the model's predictive distribution. If $r_i > 1$, EPC transmits an override flag followed by a compact representation of the correction. For ranks $2 \le r_i \le K$ (where K is a hyperparameter), this correction is the rank index itself. For ranks $r_i > K$, EPC can optionally fall back to transmitting the full token, allowing for lossless reconstruction of the masked subset. This design provides two controls: p_{mask} trades kept tokens for model predictions, while the rank threshold K controls the trade-off between the correction stream's bit rate and its error-bounding capability.
- **Vector Quantisation with Residual Patching (VQ+RE)** As a transform-based baseline, We formalise a codec that operates in the model's latent space. This VQ pipeline compresses the key and value vectors within each attention head against learned codebooks. To mitigate

train-test mismatch, the model is trained with quantised representations using scheduled self-feeding. The training objective combines the standard language modelling loss with VQ commitment and code utilisation losses. After an initial reconstruction from the quantised latent states, a residual patching step is applied. This step computes a diff between the original and reconstructed text and transmits corrections for the mismatched tokens using the same rank-based strategy as EPC. This guarantees a bounded error and provides a fair comparison against EPC's token-space residual stream.

A complete derivation for each is in the Appendix.

Stage 3: Reconstruction and Refinement. For all codecs, decompression involves deterministic, confidence-ordered infilling. For EPC and VQ+RE, this can be iterated. Let g_ϕ be a small refinement head and $\hat{X}^{(0)}$ be the initial decode. For $s = 1, \ldots, T_{\text{it}}$ (default $T_{\text{it}} = 2$):

$$\begin{aligned} u_i^{(s)} &= 1 - \max_v p_\phi(v \mid \hat{X}^{(s-1)}, i), \\ \mathcal{U}^{(s)} &= \text{top-}M_s \text{ indices of } u^{(s)}, \\ \hat{x}_i^{(s)} &= \arg\max_v p_\phi(v \mid \hat{X}^{(s-1)}, i),\ We \in \mathcal{U}^{(s)}. \end{aligned} \tag{1}$$

This improves predictions without extra side information. The refinement head is counted in the static model size.

3.2 Implementation and Evaluation Protocol

The experiments are run on NVIDIA V100 GPUs using `PyTorch` and `Hugging Face`. The key results are averaged over five random seeds. Our primary objective is to compare the rate-distortion performance of PM, EPC, and VQ+RE. We fine-tune the MLM as described in Stage 1. We then generate RD curves by sweeping key parameters for each codec: masking rate p_{mask} for PM and EPC, and rank threshold K for EPC and VQ+RE's residual stream. The curves plot BPC vs. Fidelity, providing a clear comparison of their efficiency.

4 Experiments and Results

4.1 Limitations

5 Conclusion

References

[1] Deletang G, Ruoss A, Duquenne PA, Catt E, Genewein T, Mattern C, Grau-Moya J, Wenliang LK, Aitchison M, Orseau L, Hutter M, Veness J. Language modeling is compression. [Internet]. In: The Twelfth International Conference on Learning Representations; 2024. [cited 2025 May 15]. Available from: https://openreview.net/forum?id=jznbgiynus.

[2] Shannon CE, Weaver W. A mathematical theory of communication. [Internet]. Bell Syst Tech J. 1949;27(4):623–656. [cited 2025 May 17]. Available from: https://doi.org/10.1002/j.1538-7305.1948.tb00917.x.

[3] Witten IH, Neal RM, Cleary JG. Arithmetic coding for data compression. [Internet]. Communications of the ACM. 1987;30(6):520–540. [cited 2025 May 17]. Available from: https://doi.org/10.1145/214762.214771.

[4] Rissanen J. Modeling by shortest data description. [Internet]. Automatica. 1978;14(5):465–471. [cited 2025 May 17]. Available from: https://doi.org/10.1016/0005-1098(78)90005-5.

[5] Valmeekam CSK, Narayanan K, Kalathil D, Chamberland JF, Shakkottai S. LLMZip: lossless text compression using large language models. [Internet]. arXiv.org; 2023 Jun 7 [cited 2025 May 15]. Available from: https://doi.org/10.48550/arXiv.2306.04050.

[6] van den Oord A, Vinyals O, Kavukcuoglu K. Neural discrete representation learning. [Internet]. In: Guyon I, Von Luxburg U, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R, editors. *Advances in Neural Information Processing Systems*; 2017. Vol. 30. Curran Associates, Inc. [cited 2025 Jul 19]. Available from: https://proceedings.neurips.cc/paper_files/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf.

[7] Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V. RoBERTa: a robustly optimized BERT pretraining approach. [Internet]. arXiv.org; 2019 Jul 24 [cited 2025 Jun 21]. Available from: https://arxiv.org/abs/1907.11692.

[8] Sanh V, Debut L, Chaumond J, Wolf T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. [Internet]. *arXiv preprint arXiv:1910.01108*. 2020. [cited 2025 Jul 19]. Available from: https://arxiv.org/abs/1910.01108.

[9] Zhang T, Kishore V, Wu F, Weinberger KQ, Artzi Y. BERTScore: Evaluating text generation with BERT. [Internet]. arXiv.org; 2019. [cited 2025 Jun 9]. Available from: https://arxiv.org/abs/1904.09675.

[10] Weinshall D, Cohen G, Amir D. Curriculum learning by transfer learning: theory and experiments with deep networks. [Internet]. arXiv.org; 2018 Feb 11 [cited 2025 Jul 21]. Available from: https://arxiv.org/abs/1802.03796.

[11] Kong Y, Liu L, Wang J, Tao D. Adaptive curriculum learning. [Internet]. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV); 2021 Oct 11–17; Montreal, Canada. IEEE; 2021. [cited 2025 Jul 21]. Available from: https://doi.org/10.1109/iccv48922.2021.00502.

A Detailed Codec Formalisms

A.1 Predictive Masking: Rate-Accurate Formalism

Let $x_{1:N}$ be the token sequence. Let $\mathcal{M} \subset \{1, \ldots, N\}$ be the masked indices, $|\mathcal{M}| = \lfloor p_{\text{mask}} N \rfloor$, and $\mathcal{S} = \{1, \ldots, N\} \setminus \mathcal{M}$ the kept indices with fraction $p_{\text{keep}} = 1 - p_{\text{mask}}$. Denote model surprisal at i by $s_i = -\log_2 q_\theta(x_i \mid X_{\setminus i})$.

Mask set selection. I use windowed equalisation. Partition $\{1, \ldots, N\}$ into windows $\{W_w\}_w$; choose per-window thresholds τ_w such that

$$\begin{aligned} \mathcal{M} &= \bigcup_w \big\{ i \in W_w : s_i \le \tau_w \big\}, \\ |\mathcal{M} \cap W_w| &= \big\lfloor p_{\text{mask}} \cdot |W_w| \big\rfloor. \end{aligned} \tag{2}$$

I cap masked runs by enforcing a maximum run-length, which bounds local error cascades.

Payload. Positions are encoded with a succinct bitvector; the cost satisfies

$$\text{bits}_{\text{pos}}^{\text{PM}} \;\approx\; N \min\big\{ H_2(p_{\text{keep}}),\; \mathcal{R}_{\text{RLE}}(p_{\text{keep}}) \big\}, \tag{3}$$

where $H_2(\cdot)$ is the binary entropy and $\mathcal{R}_{\text{RLE}}$ is the expected rate for run-length encoding. Kept tokens are entropy-coded in natural order with an auxiliary autoregressive coder P_ψ:

$$\text{bits}_{\text{tok}}^{\text{PM}} \;=\; \sum_{i \in \mathcal{S}} -\log_2 P_\psi\big(x_i \mid x_j : j \in \mathcal{S}, j < i\big). \tag{4}$$

Total payload bits for PM are $\text{bits}_{\text{PM}} = \text{bits}_{\text{pos}}^{\text{PM}} + \text{bits}_{\text{tok}}^{\text{PM}}$.

Reconstruction. I deterministically infill $\{x_i\}_{i \in \mathcal{M}}$ with $\arg\max$ under $q_\theta(\cdot \mid X_{\setminus \mathcal{M}})$, using the same specialisation from Stage 1. The run-length cap guarantees at least one ground-truth token per window, which stabilises the local context during decoding.

A.2 Error-Bounded Predictive Coding: Rank-Indexed Residuals

Let $\mathcal{M}$ be selected as above. For each $i \in \mathcal{M}$, let $q_i(\cdot) = q_\theta(\cdot \mid X_{\setminus \mathcal{M}})$ be the MLM distribution and let $r_i \in \{1, 2, \ldots\}$ be the rank of the ground-truth token x_i in the descending order of q_i's probabilities. Fix a rank threshold $K \ge 2$.

Payload. Positions are coded as in (3). We introduce an override flag $z_i = \mathbf{1}[r_i > 1]$ and encode it with a Bernoulli coder. With masked-set top-1 accuracy $p_1 = \Pr(r_i = 1)$,

$$\text{bits}_{\text{flag}}^{\text{EPC}} \;\approx\; |\mathcal{M}| H_2(1 - p_1). \tag{5}$$

If $z_i = 1$ and $2 \le r_i \le K$, We transmit a rank index with code length $c_{\text{rank}}(r_i)$. If $r_i > K$, We fall back to the full token code of length $\ell(x_i)$. The correction stream cost is

$$\text{bits}_{\text{corr}}^{\text{EPC}} = \sum_{i \in \mathcal{M}} \big(\mathbf{1}[2 \le r_i \le K] \cdot c_{\text{rank}}(r_i) + \mathbf{1}[r_i > K] \cdot \ell(x_i) \big). \tag{6}$$

Total payload bits for EPC are $\text{bits}_{\text{EPC}} \approx \text{bits}_{\text{pos}}^{\text{PM}} + \text{bits}_{\text{flag}}^{\text{EPC}} + \text{bits}_{\text{corr}}^{\text{EPC}}$.

Distortion control. If fallback is enabled for all $r_i > K$, EPC reconstructs the masked subset losslessly. In the lossy regime, We can disable the fallback or constrain it with a budget $\beta \in [0, 1]$. The masked-set token error rate D_{masked} is non-increasing in K and β. This exposes two orthogonal controls: p_{mask} trades positions versus modelling load, while K and β interpolate between a cheap residual stream and exact correction.

Reconstruction. At decode, We run the same specialised MLM to obtain the ranked list at each $i \in \mathcal{M}$, apply rank overrides where provided, and otherwise use $\arg\max$ as in PM.

A.3 Vector Quantisation: Stabilised, Bounded-Error Formulation

The goal is to train exactly what We deploy: keys/values are quantised during training, scheduled self-feeding removes train-test mismatch, and codebooks are learnt via EMA with utilisation regularisation.

Notation. Let $x_{1:T}$ be input tokens. The transformer has L layers and H heads. At layer ℓ, token t has hidden state $h_t^{(\ell)} \in \mathbb{R}^d$. For head h, queries, keys, and values are computed as $Q_t^{(\ell,h)} = h_t^{(\ell-1)} W_Q^{\ell,h}$, $K_u^{(\ell,h)} = h_u^{(\ell-1)} W_K^{\ell,h}$, $V_u^{(\ell,h)} = h_u^{(\ell-1)} W_V^{\ell,h}$, with $W_\cdot^{\ell,h} \in \mathbb{R}^{d \times d_h}$. Each head maintains two codebooks $\mathcal{C}_K^{\ell,h} = \{c_{K,j}^{\ell,h}\}_{j=1}^{K_K}$ and $\mathcal{C}_V^{\ell,h} = \{c_{V,j}^{\ell,h}\}_{j=1}^{K_V}$, where $c_{\cdot,j}^{\ell,h} \in \mathbb{R}^{d_h}$.

Quantise K/V at training time. Nearest-neighbour quantisation maps vectors to codebook entries:

$$\kappa_u^{(\ell,h)} = \arg\min_j \|K_u^{(\ell,h)} - c_{K,j}^{\ell,h}\|_2^2, \quad \tilde{K}_u^{(\ell,h)} = c_{K,\kappa_u^{(\ell,h)}}^{\ell,h}, \tag{7}$$

and similarly for values V to get $\tilde{V}_u^{(\ell,h)}$. Attention then uses these quantised vectors:

$$\begin{aligned} A_{t,u}^{(\ell,h)} &= \mathrm{softmax}_u \left(\frac{Q_t^{(\ell,h)} \, \tilde{K}_u^{(\ell,h)\top}}{\sqrt{d_h}} \right), \\ o_t^{(\ell,h)} &= \sum_{u=1}^{T} A_{t,u}^{(\ell,h)} \tilde{V}_u^{(\ell,h)}. \end{aligned} \tag{8}$$

I apply the straight-through estimator (STE) for backpropagation.

Scheduled self-feeding. To remove exposure bias between training and testing, We replace teacher activations with their quantised counterparts with probability α_τ at the training step τ, using an inverse sigmoid schedule. At test time, $\alpha_\tau = 1$.

EMA codebooks and commitment. Codebooks are updated using an exponential moving average (EMA) with decay ρ. We also added a commitment loss:

$$\mathcal{L}_{\text{commit}} = \beta \sum_u \|\mathrm{sg}[K_u] - \tilde{K}_u\|_2^2 + \gamma \sum_u \|K_u - \mathrm{sg}[\tilde{K}_u]\|_2^2, \tag{9}$$

and analogously for V, where $\mathrm{sg}[\cdot]$ is the stop-gradient operator.

Code usage regularisation. To avoid dead codes, We penalise the divergence of the empirical code usage distribution from a uniform prior.

Full training objective. The final loss combines the standard cross-entropy language modelling loss with VQ-specific terms:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{commit}} + \mathcal{L}_{\text{util}}^{K} + \mathcal{L}_{\text{util}}^{V} + \mathcal{L}_{\text{sem}}, \tag{10}$$

where $\mathcal{L}_{\text{sem}}$ is a lightweight semantic consistency loss. Training uses the quantised $\tilde{K}, \tilde{V}$ (blended by α_τ), EMA updates, and the STE.