Online Minimum Description Length Passive-Aggressive Algorithms

John Hurwitz
University of Maryland, Baltimore County
j216@umbc.edu

Edward Raff
CrowdStrike
University of Maryland, Baltimore County
Syracuse University

Charles Nicholas
University of Maryland, Baltimore County

Abstract

We present an algorithm for margin-based online learning using lossless compressors (like zip) via the Minimum Description Length principle. The technique, which we call *MDL-PA*, can be viewed as an analogue of Passive-Aggressive algorithms for probability distributions (equivalently viewed as lossless compression schemes) where we consider a *code length margin*. Separate models are maintained per class, and the update rule is an information projection correcting the true-class model's prediction on a margin-violating sample. To bridge theory and practice, we demonstrate a practical approximation to this online learning setup via adaptive Lempel-Ziv-style compression dictionaries to classify sequences in the text and malware domains.

1 Introduction

Lossless compression and prediction are fundamentally connected [16]. In order to compress well, a model must discover regularities in data, and predictive models require the same. This connection is formalized by the Minimum Description Length (MDL) principle [19, 11], which selects the model that gives the shortest description of data, counting the size of the model itself. This yields a convenient way to use lossless compression programs for classification: maintain separate compression models per class whose compressor states are initialized from the class-specific data, and then select the class model which best compresses the given sample. This idea (along with related but alternative means of using compression for prediction [15]) has motivated a long line of (*lossless*) compression-based learning methods that operate directly on raw sequences, avoiding hand-designed features, and applying naturally across domains such as text, biological sequences, and malware binaries. Despite numerous advances in the application of lossless compression to machine learning tasks to the offline learning setting, a clear application in the online learning setting remains understudied. Indeed, compression programs based on Lempel-Ziv schemes practically implement a form of online learning by continually updating an internal compressor state based on previously seen symbols, gradually improving its ability to compress a sequence containing patterns over time. This suggests a natural question: *can we formulate lossless compression-based classification as an online learning problem?* We answer this question by introducing *MDL-PA*, an online Passive-Aggressive-style algorithm for MDL classification. As classic Passive-Aggressive learning operates in Euclidean weight space, our MDL variant operates in distribution space. Our work crucially utilizes the correspondence between probability assignments and ideal code lengths established by Shannon's source coding framework [22]. In contrast to previous works which perform Passive-Aggressive updates on distributions over *parameters* [23, 8, 9, 25], our approach performs updates on distributions over *data*. The compressed length of a sample using some category-specific

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

model corresponds to a likelihood assignment via information theory: a probability p corresponds to code length $-\log p$. As obtaining explicit distributions over the entire input space of a practical machine learning task is intractable, we additionally instantiate a practical version of MDL-PA using adaptive Lempel-Ziv-style compression dictionaries. Here, the update rule triggers on rounds violating a *code-length* margin, and inserts the bytes from that sample into the true-class compression dictionary so as to reduce its code length under the correct class and yield correct prediction. We empirically evaluate MDL-PA on text and malware classification tasks, showing effectiveness in both domains and improved performance compared to alternative offline compression-based learning approaches.

Our contributions are: **(1)** We formulate MDL classification as an online learning problem and introduce *MDL-PA*, a theoretical Passive-Aggressive algorithm that operates in distribution space with closed-form update (Theorem 1). **(2)** We prove a cumulative model-movement bound for MDL-PA under a distribution-space realizability assumption (Theorem 2) and, under an additional uniform slack condition and a positive margin, a finite mistake bound (Corollary 1). **(3)** We develop a practical surrogate for MDL-PA based on adaptive Lempel-Ziv-style compression dictionaries that operate directly on byte sequences and thus applicable to arbitrary sequences. We empirically evaluate this practical approach on text and malware classification tasks, demonstrating online effectiveness and improved held-out test performance over compression-similarity baselines when using validation-selected margins.

This work is organized as follows: we provide background and related work in compression-based learning, MDL, online learning and Passive-Aggressive algorithms in §2, introduce MDL-PA in §3, describe our practical instantiation with adaptive Lempel-Ziv compression dictionaries in §4, report experimental results in text and malware tasks in §5, provide a theoretical analysis in §6 establishing mistake bounds under a reference-distribution assumption, and finally conclude in §7.

2 Background and related work

MDL and Compression-Based Learning The *Minimum Description Length* (MDL) principle [19, 11] is an approach to the model selection problem in inductive inference that explicitly ties learning to compression, as both can be seen as identifying regularities in the data. Here, a model is considered best if it provides the shortest description of the data, counting the size of the model itself. Given a dataset D and a set of candidate hypotheses $\mathcal{H}$, the MDL principle selects the hypothesis $H \in \mathcal{H}$ that minimizes the total description length $L(D) = L(H) + L(D|H)$ where $L(H)$ represents the length in bits of the description of the model, and $L(D|H)$ is the length of the data when encoded with the help of the model. Therefore the selected model is $\arg\min_i L(H_i) + L(D|H_i)$. Using the well-known correspondence between probability and code lengths that an event with probability p can be encoded in $-\log_2 p$ bits, we see that the model which minimizes description length is equivalently the maximum a posteriori (MAP) estimate from Bayesian inference. The MAP rule selects the hypothesis which maximizes the posterior probability of the hypothesis given the data: $\arg\min_H L(H) + L(D|H) = \arg\max_H P(H|D)$. There are primarily two distinct methods by which lossless compression programs have been used for machine learning tasks. The first is via the Normalized Compression Distance (NCD) [15], which uses compression ratios to obtain a compression distance for distance-based learners. The second, which we concern ourselves with for this work, uses the MDL principle with class-conditional compression models, used in a number of works since the early 2000s for sequence classification tasks [10, 2, 24, 17, 3].

Passive-Aggressive online learning Passive-Aggressive (PA) algorithms [7] are a class of online margin-based learning methods for binary classification which update the model only when the prediction violates a margin constraint. The introduction of this margin extended the classical perceptron [20] which updated only on mistakes. Consider the binary classification setting, and denote the sample at round t by x_t associated with label $y_t \in \{+1, -1\}$. Consider weight vector $\boldsymbol{w}_t \in \mathbb{R}^n$ and classification function sign($\boldsymbol{w_t} \cdot \boldsymbol{x_t}$). We refer to $y_t(\boldsymbol{w}_t \cdot \boldsymbol{x}_t)$ as the *margin*. A positive margin indicates a correct prediction. When the margin is greater than or equal to 1, no loss is incurred; otherwise the hinge loss is $\ell(\boldsymbol{w}; (\boldsymbol{x}, y)) = \max\{0, 1 - y(\boldsymbol{w} \cdot \boldsymbol{x})\}$. The simplest variant of PA updates the weight vector as $\boldsymbol{w}_{t+1} = \arg\min_{w \in \mathbb{R}^n} \frac{1}{2}||\boldsymbol{w} - \boldsymbol{w_t}||^2$ s.t. $\ell(\boldsymbol{w}; (\boldsymbol{x}_t), y_t)) = 0$, which has a closed form solution $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \tau_t y_t \boldsymbol{x}_t$, where $\tau_t = \frac{\ell_t}{||\boldsymbol{x}_t||^2}$. Other choices of τ_t result in variants of PA.

Passive-Aggressive learning has been substantially extended beyond its original Euclidean projection formulation. Closest in spirit to our work is Bayesian Passive-Aggressive learning (BayesPA) [23], which replaces point-valued parameter updates with I-projections in a space of posterior distributions over model parameters. Other PA variants such as Confidence-Weighted learning (CW) [9], Adaptive Regularization of Weight Vectors (AROW) [8], and Soft Confidence-Weighted learning (SCW) [25] maintain Gaussian uncertainty over model parameters and perform confidence-aware large-margin online updates, expressible as I-projections. These works show that PA-style online learning naturally extends from point estimates to distributions. MDL-PA is similar in the sense of employing KL geometry over distributions, but differs fundamentally in placing the distribution on the data side (corresponding to class-conditional compressors) rather than the parameter side (uncertainty over weights). For a comprehensive survey of online learning, see [12].

3 MDL-PA: Passive-Aggressive updates in distribution space

Standard online Passive-Aggressive algorithms optimize weight vectors in Euclidean space. In MDL classification, learners are class-conditional lossless compressors, which correspond to probability distributions over the instance space. Therefore, in order to achieve online learning in the MDL paradigm, we formulate a Passive-Aggressive update directly on distributions: Euclidean distance is replaced by KL divergence, the linear margin by a *code-length margin*, and the Euclidean projection by an information projection (I-projection). We call the resulting scheme ***MDL-PA***.

Let $\mathcal{X}$ be a finite instance space $\mathcal{X}$ with $|\mathcal{X}| \geq 2$. Let $\Delta_{\mathcal{X}}$ denote the probability simplex on $\mathcal{X}$ and $\Delta_{\mathcal{X}}^{\circ}$ its relative interior, the full-support distributions, so $\Delta_{\mathcal{X}} = \{P : \mathcal{X} \to [0,1] \mid \sum_{x\in\mathcal{X}} P(x) = 1\}$ and $\Delta_{\mathcal{X}}^{\circ} = \{P \in \Delta_{\mathcal{X}} : P(x) > 0 \text{ for all } x \in \mathcal{X}\}$. Let $\mathcal{Y} = \{1, ..., K\}$ be the class set, with $K \geq 2$. By the standard correspondence between probability assignments and ideal code lengths [22, 6], a full-support distribution P induces code lengths $L_P(x) = -\log P(x)$ nats (all logarithms are natural).

At round t, the learner maintains class-conditional models $P_t^y \in \Delta_{\mathcal{X}}$, initialized at prior models $P_1^y \in \Delta_{\mathcal{X}}^{\circ}$. Given the instance x_t, the learner predicts the class whose model assigns the shorter code length (equivalently, higher likelihood):

$$\hat{y}_t = \arg\min_{y\in\mathcal{Y}} L_{P_t^y}(x_t) = \arg\max_{y\in\mathcal{Y}} P_t^y(x_t). \tag{1}$$

with ties broken with a fixed rule.

After the true label y_t is revealed, write $p_t = P_t^{y_t}(x_t)$ and $q_t = \max_{y\neq y_t} P_t^y(x_t)$ for the true-class and best competing-class probabilities. We define the *code-length margin*

$$\gamma_t = \log(p_t/q_t) \tag{2}$$

measuring the code-length advantage of the true-class compressor over the competitor on the current example. The prediction is correct when $\gamma_t > 0$, incorrect when $\gamma_t < 0$, and $\gamma_t = 0$ is a tie. Given a margin parameter $\rho \geq 0$, we say that the current example satisfies the margin if $\gamma_t \geq \rho$; here the learner is *passive*. Otherwise, an *aggressive* update is triggered whenever the margin is not satisfied, i.e. $\gamma_t < \rho$, equivalently $p_t < e^{\rho} q_t$. Let $a_t = e^{\rho} q_t$. The update is the following I-projection:

$$P_{t+1}^{y_t} = \arg\min_{Q\in\Delta_{\mathcal{X}}} D_{\mathrm{KL}}(Q\|P_t^{y_t}) \quad \text{s.t.} \quad Q(x_t) \geq a_t. \tag{3}$$

All competing models remain unchanged: $P_{t+1}^y = P_t^y$ for $y \neq y_t$.

The update in (3) is directly analogous to the classical Passive-Aggressive update, $\boldsymbol{w}_{t+1} = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}_t\|^2$ s.t. $y_t(\boldsymbol{w} \cdot \boldsymbol{x}_t) \geq \rho$. In this view, the Euclidean distance is replaced by KL divergence, and the linear margin constraint is replaced by a code-length margin constraint. Thus, the MDL update can be understood as a Passive-Aggressive projection in KL geometry over the probability simplex. The competing models are unchanged.

Theorem 1 (Closed-form update)**.** *Fix an update round, so* $0 < p_t < a_t$*, and suppose* $a_t < 1$*. Then the unique solution to* (3) *is*

$$P_{t+1}^{y_t}(x) = \begin{cases} a_t & x = x_t, \\ P_t^{y_t}(x) \cdot \dfrac{1-a_t}{1-p_t} & x \neq x_t \end{cases} \tag{4}$$

This update rule is interpretable as increasing the probability assigned to the margin-violating example to exactly the value which satisfies the desired code-length margin, while the remaining probability mass is renormalized. Proof in Appendix A.1.

3.1 Comparison with Classical PA via Bregman projections

Recall that a Bregman projection finds the closest point in a constraint set under a Bregman divergence. The update in (3) is a Bregman projection in KL geometry, directly paralleling the classical PA update, which is a Bregman projection in Euclidean geometry. Concretely, (3) is the Bregman projection of $P_t^{y_t}$ onto the half-space $\{Q \in \Delta_{\mathcal{X}} : Q(x_t) \geq e^{\rho} q_t\}$ under D_{KL}. This situates MDL-PA and classical PA as two distinct instantiations of the more general Bregman projection, with differing choice of geometries. We compare our formulation with classical PA in Table 1. The choice of a convex mirror map ψ induces a Bregman divergence.

Table 1: Classical Passive-Aggressive learning and MDL-PA as Bregman projection methods. MDL-PA replaces Euclidean geometry with KL geometry on the probability simplex $\Delta_{\mathcal{X}}$.

	Classical PA	MDL-PA
Domain	$\mathbb{R}^d$	$\Delta_{\mathcal{X}}$
Mirror map ψ	$\frac{1}{2}\|\mathbf{w}\|^2$	$\sum_x P(x) \log P(x)$ (negative entropy)
Induced divergence	$\frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|^2$	$D_{\mathrm{KL}}(Q \| P_t)$
Margin constraint	$y_t \langle \mathbf{w}, \mathbf{x}_t \rangle \geq \rho$	$Q(x_t) \geq e^{\rho} q_t$
Projection	Euclidean projection	Information projection

4 Practical approximation with adaptive Lempel-Ziv dictionaries

The MDL-PA update in Equation 4 is formulated over explicit probability distributions. While it is feasible in general to convert between distributions and lossless codes, considering the distribution over the entire input space for real-world machine learning tasks is not feasible. Indeed, practical lossless compression algorithms based on Lempel-Ziv schemes do not maintain explicit probability distributions but rather search for byte subsequence repetitions via maintaining some history of previously-seen data (a lookback window).

To bridge the theoretical MDL-PA update with practical compression programs, we propose an approximation based on adaptive class-conditional compression dictionaries. We maintain one compressor D_t^y for each class y. Prediction still follows the MDL rule: $\hat{y}_t = \arg\min_{y \in \mathcal{Y}} L_{D_t^y}(x_t)$, where $L_{D_t^y}(x_t)$ denotes the compressed length of sample x_t under the compressor associated with class y at round t. The classification task predicts the label (e.g., malicious vs. benign) of the entire sequence.

For a labeled example (x_t, y_t), define the multiclass code-length margin $\gamma_t = \min_{y \neq y_t} L_{D_t^y}(x_t) - L_{D_t^{y_t}}(x_t)$. A positive margin indicates that the true-class compressor yields the shortest description length (and thus the classification decision is correct). Given a margin parameter $\rho \geq 0$, we update only the true-class compressor by appending the bytes of the current sample to its dictionary state: $D_{t+1}^{y_t} = D_t^{y_t} \parallel x_t$, where $\parallel$ denotes byte concatenation, while all competing class compressors remain unchanged: $D_{t+1}^{y} = D_t^{y}$, $y \neq y_t$. This yields a practical passive-aggressive update. The learner is *passive* when the current margin is sufficiently large, and *aggressive* when the margin is violated by inserting the example's bytes directly into the true-class compression dictionary state.

This update can be interpreted as an approximation of the theoretical I-projection in (3). In the theoretical update, probability mass assigned to x_t under the true class is increased while the remaining mass is re-normalized proportionally. In our practical approach with adaptive compression dictionaries, appending x_t to dictionary D decreases its own code length since we provide all of its bytes to the lookback window to provide candidate matches. This will also reduce the code length of future samples from the same class which also contain matches to subsequences within x_t. The correspondence between code lengths and probabilities $L_D(x) \approx -\log P_D(x)$ means that reducing the code length is a practical surrogate for increasing probability assigned to x_t and to samples similar to x_t.

Algorithm 1 Practical online MDL-PA

Initialize empty compressor D^y for each class $y \in \mathcal{Y}$.
for each labeled sample (x_t, y_t) **do**
  Compute lengths $\{L_{D^y}(x_t)\}_{y \in \mathcal{Y}}$.
  $\hat{y}_t \leftarrow \arg\min_y L_{D^y}(x_t)$ ▷ Predict
  $\gamma_t \leftarrow \min_{y \neq y_t} L_{D^y}(x_t) - L_{D^{y_t}}(x_t)$ ▷ Margin
  if $\gamma_t < \rho$ **or** $\hat{y}_t \neq y_t$ **then**
    $D^{y_t} \leftarrow D^{y_t} \| x_t$

In our implementation, we instantiate each class compressor using `zstandard` [5] raw content compression dictionaries. Such a dictionary can simply be viewed as a long byte sequence which acts as a simulated lookback window for Lempel-Ziv compression. The resulting method requires no gradient computation or explicit feature representation, and operates directly in byte sequence space, making it applicable to arbitrary data including text and malware binaries. We present pseudocode for this procedure in Algorithm 1.

As opposed to historical works using lossless compressors and the MDL rule for classification and which generally use all of the available class data to condition the class compression model [10, 24, 3, 17], our approach allows for the model to simply ignore patterns from examples that are already correctly classified, potentially allowing for a higher "signal-to-noise" ratio when considering the bytes present in each class-conditioned dictionary. This also allows the dictionary size to grow at a more reasonable rate, if memory constraints are a consideration.

5 Experiments

5.1 Setup

We evaluate the practical MDL-PA algorithm in an online prequential setting, wherein each training example is streamed one at a time. At each round, each class-conditional compressor compresses the sample in order to obtain a compressed length, and then the label is predicted according to the MDL rule: assign the corresponding label to the class compressor which achieved shortest code length. After the label is revealed, we perform the practical MDL-PA update by inserting the bytes of a margin-violating sample into the true-class compressor dictionary. The prequential error is the cumulative fraction of mistakes made during this online pass. After the online pass is complete, we evaluate the resulting compressors on a held-out test set. We report two quantities for MDL-PA: prequential error, measuring online performance, and final test error, measuring the quality of the learned compressors after processing the entire training stream and allowing for comparison with offline approaches. LZJD serves as a representative compression-based nearest-neighbor baseline which uses compression principles to obtain similarity scores for offline prediction.

Table 2: Dataset statistics. "Mean size" is the per-sample byte length averaged over train and test.

Dataset	# train	# test	# classes	Mean size
AG News	120,000	7,600	4	193 B
20 Newsgroups	11,314	7,532	20	1.9 KB
Drebin TAR	4,191	466	20	1.77 MB
Drebin APK	4,197	467	20	1.30 MB

We evaluate MDL-PA in the multiclass setting on text classification datasets AG News and 20newsgroups, and on Android malware from the Drebin corpus. Following [1], we evaluate malware family classification for two variants: APK, the default Android package which is zipped, and TAR, an uncompressed TAR of the APK contents. We note dataset statistics in Table 2. All experiments were run on a MacBook Pro M1 Max and require only CPU operations. Processing the entire online stream for MDL-PA (per margin) took roughly 15 minutes on AG News, 30 seconds on 20 newsgroups, and 90 minutes on Drebin, which were swept across multiple margin values.

5.2 Main results

We report in Figure 1 prequential error rate trajectories across a range of code-length margin ρ values for the text and malware classification tasks. We observe that larger margins consistently improve performance in text classification, while having limited or no effect in malware classification. This

suggests that the utility of large-margin updates depends on the underlying structure of the data. For malware files at the megabyte scale, it appears that updates triggered on mistakes only already result in good class separability in compression space.

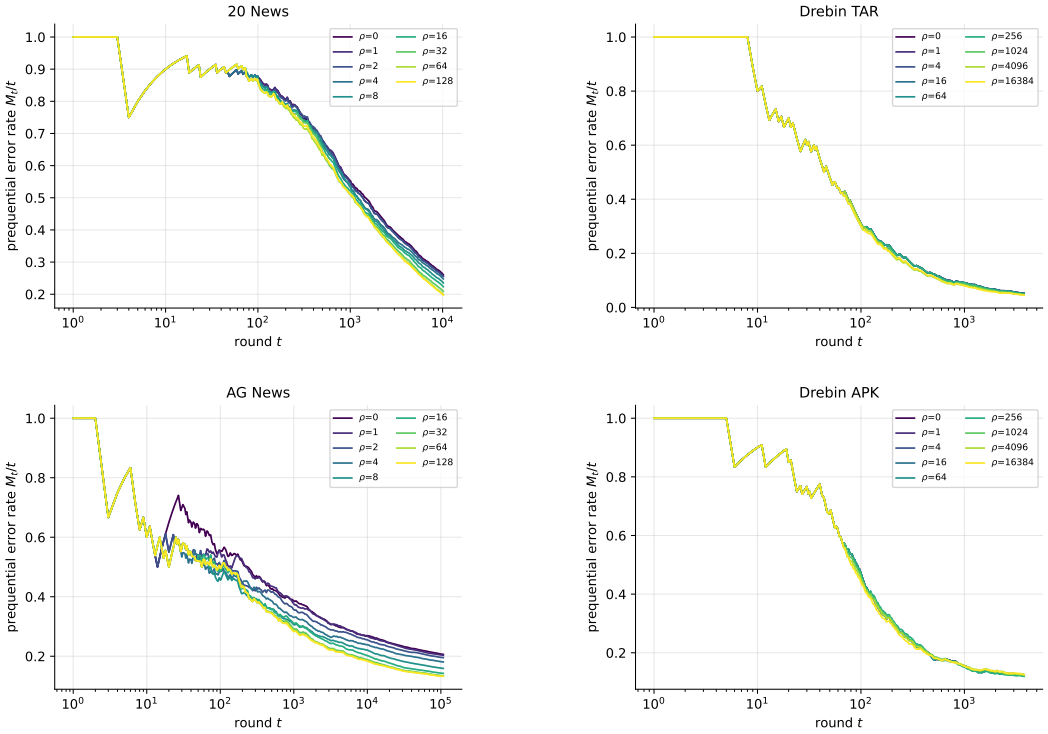

Figure 1: Running prequential error trajectories (number of mistakes so far, M_t, divided by the current round t) across code-length margin values ρ on text and malware classification tasks. For text, increasing code-length margins substantially improves prequential error rate. For malware (Drebin), the choice of code-length margin has minimal effect.

We report for all datasets final prequential error in Table 3 and held-out offline test error in Table 4, along with a comparison to two compression-based similarity approaches. The first is LZJD [18], a method of comparing byte sequences which converts each sequence into an alternative representation based on a Lempel Ziv compression operation, and then performs a Jaccard distance between those representations. Scalars from LZJD are used as the distance metric for kNN. The second is the NCD [15], which we instantiate via `gzip`, whose scalars are also used for kNN, as re-popularized in the modern era in [13], however this flavor of compression-similarity approaches date back a quarter-century [14, 15, 4, 17, 26, 21]. In both cases, we set $k = 1$.

Table 3: Final prequential (online) error M_T/T for MDL-PA across all evaluated datasets. The best (lowest) value in each column is bolded. The margin-tuned $\rho^\star$ was chosen by evaluating against a validation set: AG News: $\rho^\star = 128$, Drebin TAR: $\rho^\star = 0$, Drebin APK: $\rho^\star = 4096$, 20 News: $\rho^\star = 128$.

Method	AG News	Drebin TAR	Drebin APK	20 News
MDL-PA, $\rho = 0$	0.2062	**0.0517**	**0.1221**	0.2604
MDL-PA, $\rho = \rho^\star$	**0.1325**	**0.0517**	0.1239	**0.1972**

Across domains, MDL-PA achieves strong performance, outperforming the alternative compression-similarity baselines in all settings. This highlights the effectiveness of online adaptation via margin-triggered updates compared to static compression-similarity-based approaches. We observe that tuning the margin on a validation set can significantly improve performance over the zero-margin case,

but the effect is domain-dependent. On AG News, increasing the margin from $\rho = 0$ to $\rho^\star = 128$ reduces prequential error from 0.2062 to 0.1325 and test error from 0.1861 to 0.1204. In contrast, for Drebin TAR, the optimal margin is $\rho^\star = 0$, indicating that updating only on mistakes suffices for this dataset. The class-conditional byte structure appears sufficiently separable that additional updates on low-margin but correctly classified examples do not improve balanced test error.

Table 4: Held-out test error for MDL-PA (frozen dictionaries after the online run), the offline LZJD + 1-NN baseline, and the gzip + 1-NN NCD baseline. The best (lowest) value in each column is bolded. Test error for Drebin TAR, Drebin APK is balanced test error; other Test columns are plain test error. Margin-tuned $\rho^\star$ chosen by validation-error argmin: AG News: $\rho^\star = 128$, Drebin TAR: $\rho^\star = 0$, Drebin APK: $\rho^\star = 4096$, 20 News: $\rho^\star = 128$.

Method	AG News	Drebin TAR	Drebin APK	20 News
MDL-PA, $\rho = 0$	0.1861	**0.0310**	**0.1047**	0.2935
MDL-PA, $\rho = \rho^\star$	**0.1204**	**0.0310**	0.1093	**0.2284**
LZJD + kNN	0.7154	0.1286	0.1211	0.9275
gzip + kNN (NCD)	0.1417	N/A	N/A	0.3947

5.3 Dictionary growth and update dynamics

We now examine how the compression dictionaries evolve over time. Figure 2 shows dictionary growth over time and cumulative updates across margin values for text and malware, and Table 5 lists selected margin values and their associated final prequential error, test error, update rate (number of rounds which triggered an update), and mean dictionary size across classes.

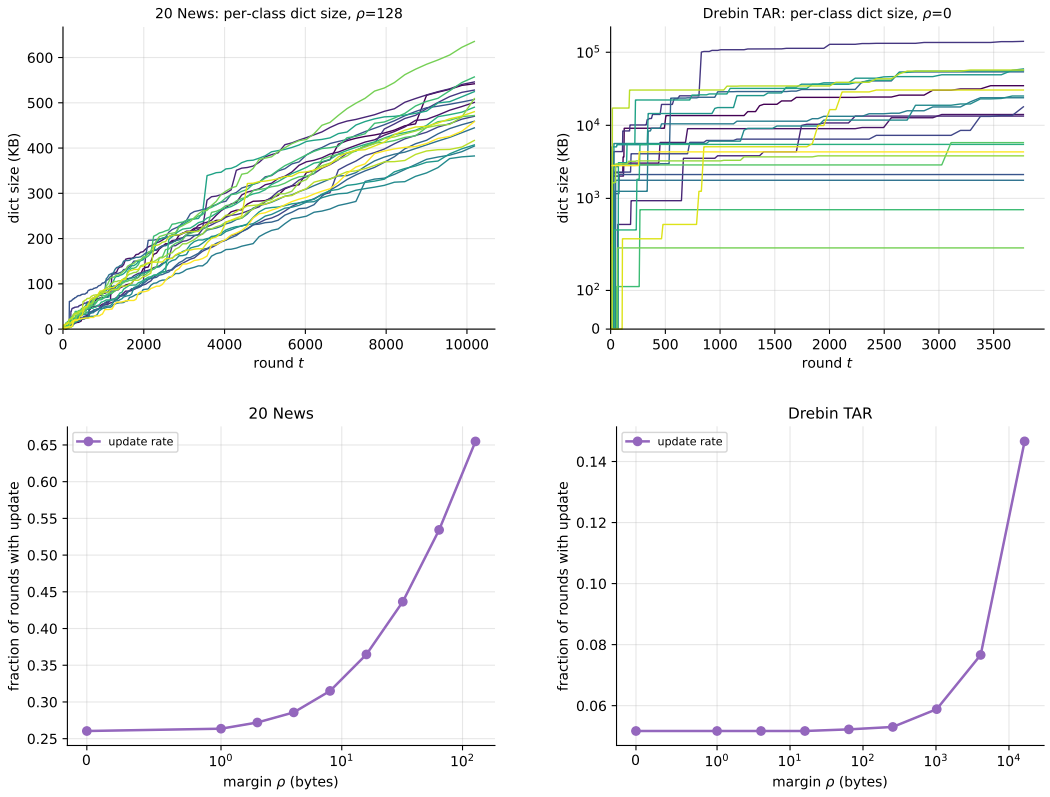

Figure 2: Dictionary growth (top) and cumulative updates (bottom) for representative text and malware datasets. Larger margins induce more frequent updates.

Table 5: Prequential error, test error, the fraction of samples that triggered an update, and mean dictionary size across selected code-length margin values in bytes. As margin increases so does the number of rounds which triggered an update and thus leads to higher byte counts in the dictionary.

Dataset	ρ (bytes)	Preq. error	Test error	Update rate	Mean dict size / class
AG News	0	0.2062	0.1861	0.206	1004.9 KB
	64	0.1331	**0.1187**	0.958	4.67 MB
	128 ($\rho^\star$)	0.1325	0.1204	0.998	4.96 MB
Drebin TAR	0	0.0517	**0.0310**	0.052	26.92 MB
Drebin APK	0	0.1221	0.1047	0.122	50.11 MB
	64	0.1252	**0.1018**	0.135	55.96 MB
	4096 ($\rho^\star$)	0.1239	0.1093	0.208	79.36 MB
20 News	0	0.2604	0.2935	0.260	196.1 KB
	64	0.1995	0.2402	0.534	382.6 KB
	128 ($\rho^\star$)	0.1972	**0.2284**	0.655	486.8 KB

In Figure 3 we visualize the trade-off between final dictionary size and test error across margin values for each dataset. We view the effect that larger dictionaries generally correspond with better prequential and test performance to be analogous to the notion of model complexity in machine learning. As the compression dictionary increases in size, it is able to store more relevant patterns and improve its predictive performance. In approximating the MDL-PA update, appending bytes from misclassified samples to the model's look-back window shortens the model's corresponding code lengths, and thus analogously increases the model's likelihood assigned to that sample and improving the model's predictions.

5.4 Data reordering

To investigate the effect that stream order has on both error and dictionary growth, we select 20 newsgroups and run 10 seeds across ρ values of 0, 64, and 128, noting mean and standard deviation in Table 6. We observe that the improvement from larger margins is highly consistent across random stream orderings. In particular, increasing the margin from $\rho = 0$ to $\rho = 128$ reduces mean held-out test error from 0.2966 ± 0.0042 to 0.2270 ± 0.0029, with low variance across all 10 seeds.

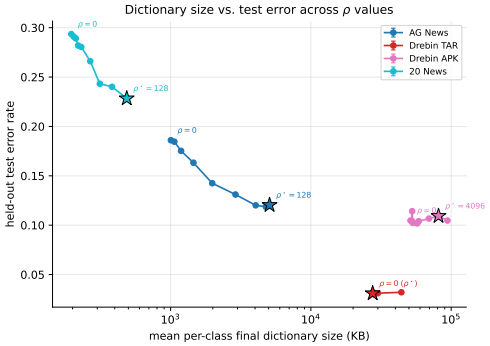

Figure 3: Dictionary-size vs. performance trade-off induced by the code length margin ρ. Each point corresponds to one swept margin value, and the starred point denotes the highest-scoring margin on a validation set.

6 Theoretical Analysis

Our analysis compares the algorithm's models against fixed reference models $R^y \in \Delta_{\mathcal{X}}$, one for each class $y \in \mathcal{Y}$, through the potential

$$\Phi_t := \sum_{y \in \mathcal{Y}} D_{\mathrm{KL}}(R^y \| P_t^y),$$

Table 6: 20 newsgroups seed variance across 10 seeds. We report mean $\pm$ 1 standard deviation across seeds.

ρ (bytes)	Preq. error	Test error	Update rate	Mean dict size / class (KB)
0	0.2610 ± 0.0015	0.2966 ± 0.0042	0.261 ± 0.001	207.6 ± 8.4
64	0.2022 ± 0.0024	0.2367 ± 0.0039	0.531 ± 0.002	386.3 ± 5.7
128 ($\rho^\star$)	0.1974 ± 0.0026	0.2270 ± 0.0029	0.655 ± 0.001	488.3 ± 4.6

which measures how far the current models are from the references.

Write $r_t = R^{y_t}(x_t)$, and let $\mathcal{U}_T = \{t \le T : \gamma_t < \rho\}$ denote the update rounds. For $a, b \in (0,1)$, define

$$D_{\mathrm{bin}}(a\|b) = a \log \frac{a}{b} + (1-a) \log \frac{1-a}{1-b}, \qquad \Lambda(a,b) = \log \frac{a(1-b)}{b(1-a)}.$$

Here, D_{bin} is the KL divergence between Bernoulli distributions, and $\Lambda(a,b) > 0$ whenever $a > b$.

Lemma 1 (Exact potential drop)**.** *On any update round with $0 < p_t < a_t < 1$,*

$$\Phi_t - \Phi_{t+1} = D_{\mathrm{bin}}(a_t\|p_t) + (r_t - a_t)\Lambda(a_t, p_t).$$

On non-update rounds, $\Phi_{t+1} = \Phi_t$.

Proof in Appendix A.2.

Lemma 2 (Monotonicity and feasibility)**.** *Suppose the stream is label-consistent, meaning that each observed instance has a single label, and the full-support priors satisfy*

$$\sup_{y,x} P_1^y(x) < e^{-\rho}.$$

Then for every round t: (i) $a_t < 1$, so every update is well defined and $P_t^y \in \Delta_{\mathcal{X}}^\circ$ for every class y; and (ii)

$$q_t \le \max_{j \ne y_t} P_1^j(x_t).$$

Proof in Appendix A.3.

Assumption 1 (ρ-separability with respect to the priors, slack $\delta \ge 0$)**.** *The stream is label-consistent, the full-support priors satisfy $\sup_{y,x} P_1^y(x) < e^{-\rho}$, and there exist reference models $R^y \in \Delta_{\mathcal{X}}$ for every class $y \in \mathcal{Y}$ such that for every instance–label pair (x, y) occurring in the stream,*

$$R^y(x) \ge e^{\rho} \max_{j \ne y} P_1^j(x) + \delta.$$

In words, there exists a fixed collection of class-conditional reference models whose probabilities dominate the margin targets determined by the initial competing models. By Lemma 2, these reference models also satisfy the post-update margin constraint on every update round. This assumption plays the role of a margin-separability condition in our analysis.

Theorem 2 (Cumulative model movement)**.** *Under Assumption 1 with $\delta = 0$,*

$$\sum_{t \in \mathcal{U}_T} D_{\mathrm{bin}}(a_t\|p_t) \le \Phi_1 = \sum_{y \in \mathcal{Y}} D_{\mathrm{KL}}(R^y\|P_1^y).$$

Proof. On each update round, Lemma 2 and Assumption 1 give

$$r_t \ge e^{\rho} \max_{j \ne y_t} P_1^j(x_t) \ge e^{\rho} q_t = a_t.$$

Since $\Lambda(a_t, p_t) > 0$, Lemma 1 yields

$$\Phi_t - \Phi_{t+1} \ge D_{\mathrm{bin}}(a_t\|p_t) \ge 0.$$

On all other rounds, $\Phi_{t+1} = \Phi_t$. Summing over $t \le T$ gives

$$\sum_{t \in \mathcal{U}_T} D_{\mathrm{bin}}(a_t\|p_t) \le \Phi_1 - \Phi_{T+1} \le \Phi_1,$$

since $\Phi_{T+1} \ge 0$. □

Proof in Appendix A.4

Corollary 1 (Finite mistake bound)**.** *Under Assumption 1 with slack $\delta > 0$ and margin $\rho > 0$, the number of prediction mistakes (rounds with $\hat{y}_t \ne y_t$) among the first T rounds is at most*

$$\frac{\Phi_1}{\delta\rho},$$

uniformly in T.

Proof in Appendix A.5.

The cumulative bound controls movement of the exact models: by (4),

$$D_{\mathrm{KL}}(P_{t+1}^{y_t} \| P_t^{y_t}) = D_{\mathrm{bin}}(a_t \| p_t)$$

on update rounds. It does not by itself bound the number of updates. The mistake bound additionally requires positive slack and may be loose because the reference probabilities and slack concern complete inputs. These guarantees apply to the exact distribution-space update; they do not automatically extend to the practical dictionary updates.

7 Conclusion

We introduce MDL-PA, a Passive-Aggressive-style online learner operating in probability distribution space. We derive its closed-form update, and instantiate a practical approximation using adaptive compression dictionaries operating directly on raw byte sequences. Experiments on text and malware classification demonstrate strong online adaptation. Our results suggest that compression-based learning admits a natural online learning formulation through information geometry.

Acknowledgments and Disclosure of Funding

This work received no external funding. The authors declare no competing interests.

References

[1] Lempel-ziv jaccard distance, an effective alternative to ssdeep and sdhash. *Digital Investigation*, 24. ISSN 1742-2876. doi: 10.1016/j.diin.2017.12.004. URL `http://dx.doi.org/10.1016/j.diin.2017.12.004`.

[2] Dario Benedetto, Emanuele Caglioti, and Vittorio Loreto. Language trees and zipping. *Physical Review Letters*, 88(4), January 2002. ISSN 1079-7114. doi: 10.1103/physrevlett.88.048702. URL `http://dx.doi.org/10.1103/PhysRevLett.88.048702`.

[3] Andrej Bratko, Gordon V Cormack, Bogdan Filipič, Thomas R Lynam, and Blaž Zupan. Spam filtering using statistical data compression models. *Journal of Machine Learning Research*, 7: 2673–2698, 2006.

[4] X. Chen, B. Francia, M. Li, B. McKinnon, and A. Seker. Shared information and program plagiarism detection. 50(7):1545–1551, 2004. ISSN 0018-9448. doi: 10.1109/TIT.2004.830793. URL `http://ieeexplore.ieee.org/document/1306552/`.

[5] Yann Collet and Murray Kucherawy. Zstandard Compression and the 'application/zstd' Media Type. RFC 8878, February 2021. URL `https://www.rfc-editor.org/info/rfc8878`.

[6] Thomas Cover and Joy Thomas. *Elements of Information Theory*. Wiley, 2nd edition edition, 2009.

[7] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585, December 2006. ISSN 1532-4435.

[8] Koby Crammer, Alex Kulesza, and Mark Dredze. Adaptive regularization of weight vectors. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009. URL `https://proceedings.neurips.cc/paper_files/paper/2009/file/8ebda540cbcc4d7336496819a46a1b68-Paper.pdf`.

[9] Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 264–271, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390190. URL `https://doi.org/10.1145/1390156.1390190`.

[10] Eibe Frank, Chang Chui, and Ian H. Witten. Text categorization using compression models. In *Proceedings of the Conference on Data Compression*, DCC '00, page 555, USA, 2000. IEEE Computer Society. ISBN 0769505929.

[11] Peter Grunwald. A tutorial introduction to the minimum description length principle, 2004. URL https://arxiv.org/abs/math/0406077.

[12] Steven C. H. Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey, 2018. URL https://arxiv.org/abs/1802.02871.

[13] Zhiying Jiang, Matthew Yang, Mikhail Tsirlin, Raphael Tang, Yiqin Dai, and Jimmy Lin. "low-resource" text classification: A parameter-free classification method with compressors. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6810–6828, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.426. URL https://aclanthology.org/2023.findings-acl.426.

[14] Ming Li, Jonathan H. Badger, Xin Chen, Sam Kwong, Paul Kearney, and Haoyong Zhang. An information-based sequence distance and its application to whole mitochondrial genome phylogeny. *Bioinformatics*, 17(2):149–154, 02 2001.

[15] Ming Li, Xin Chen, Xin Li, Bin Ma, and Paul MB Vitányi. The similarity metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, 2004.

[16] David J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, Cambridge, UK, 2003.

[17] Yuval Marton, Ning Wu, and Lisa Hellerstein. On compression-based text classification. In David E. Losada and Juan M. Fernández-Luna, editors, *Advances in Information Retrieval*, volume 3408 of *Lecture Notes in Computer Science*, pages 300–314. Springer, Berlin, Heidelberg, 2005. doi: 10.1007/978-3-540-31865-1_22. URL https://doi.org/10.1007/978-3-540-31865-1_22.

[18] Edward Raff and Charles Nicholas. An alternative to ncd for large sequences, lempel-ziv jaccard distance. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 1007–1015, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4887-4. doi: 10.1145/3097983.3098111. URL http://doi.acm.org/10.1145/3097983.3098111.

[19] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(78)90005-5. URL https://www.sciencedirect.com/science/article/pii/0005109878900055.

[20] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[21] D. Sculley and C.E. Brodley. Compression and machine learning: a new perspective on feature space vectors. In *Data Compression Conference (DCC'06)*, pages 332–341, 2006. doi: 10.1109/DCC.2006.13.

[22] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27: 379–423, 1948.

[23] Tianlin Shi and Jun Zhu. Online Bayesian passive-aggressive learning. *Journal of Machine Learning Research*, 18(33):1–39, 2017.

[24] W. J. Teahan and D. J. Harper. Using compression-based language models for text categorization. In *Proceedings of the Workshop on Language Modeling and Information Retrieval*, 2001.

[25] Jialei Wang, Peilin Zhao, and Steven C. H. Hoi. Exact soft confidence-weighted learning. In *Proceedings of the 29th International Conference on Machine Learning*, pages 121–128, 2012.

[26] Stephanie Wehner. Analyzing worms and network traffic using compression. *J. Comput. Secur.*, 15:303–320, 2005. URL https://api.semanticscholar.org/CorpusID:13391901.

A Proofs

A.1 Proof of Theorem 1

Proof. Write $P = P_t^{y_t}, p = P(x_t)$, $a = a_t$, and $s = Q(x_t)$. For $s < 1$, denote the conditional distributions on $\mathcal{X}\backslash\{x_t\}$ by Q_- and P_-. Define

$$d(s\|p) = s\log\frac{s}{p} + (1-s)\log\frac{1-s}{1-p}.$$

for $s \in [0,1]$, with zero-weight terms interpreted as zero. Expanding KL gives

$$D_{\mathrm{KL}}(Q\|P) = d(s\|p) + (1-s)D_{\mathrm{KL}}(Q_-\|P_-).$$

At $s = 1$, Q is a point mass at x_t, so $D_{KL}(Q||P) = d(1||p)$. For any fixed $s < 1$, the second term is uniquely minimized by $Q_- = P_-$. Since $d(s||p)$ is strictly increasing in s on $[p, 1]$ and feasibility requires $s \geq a > p$, the unique optimum is $s = a$ and $Q_- = P_-$.

□

A.2 Proof of Lemma 1

Lemma 1 (Exact potential drop)**.** *On any update round with* $0 < p_t < a_t < 1$*,*

$$\Phi_t - \Phi_{t+1} = D_{\mathrm{bin}}(a_t\|p_t) + (r_t - a_t)\Lambda(a_t, p_t).$$

On non-update rounds, $\Phi_{t+1} = \Phi_t$*.*

Proof. Only the true-class model changes, so with $a = a_t$, $b = p_t$, and $r = r_t$,

$$\begin{aligned}\Phi_t - \Phi_{t+1} &= \sum_x R^{y_t}(x)\log\frac{P_{t+1}^{y_t}(x)}{P_t^{y_t}(x)} \\ &= r\log\frac{a}{b} + (1-r)\log\frac{1-a}{1-b},\end{aligned}$$

since by (4) the ratio equals a/b at x_t and $(1-a)/(1-b)$ at every other point. Writing $r = a+(r-a)$ and $1-r = (1-a)-(r-a)$ and regrouping,

$$\begin{aligned}r\log\frac{a}{b} + (1-r)\log\frac{1-a}{1-b} &= D_{\mathrm{bin}}(a\|b) \\ &\quad + (r-a)\left[\log\frac{a}{b} - \log\frac{1-a}{1-b}\right],\end{aligned}$$

and the bracket is $\Lambda(a, b)$. □

A.3 Proof of Lemma 2

Lemma 2 (Monotonicity and feasibility)**.** *Suppose the stream is label-consistent, meaning that each observed instance has a single label, and the full-support priors satisfy*

$$\sup_{y,x} P_1^y(x) < e^{-\rho}.$$

Then for every round t*: (i)* $a_t < 1$*, so every update is well defined and* $P_t^y \in \Delta_{\mathcal{X}}^\circ$ *for every class* y*; and (ii)*

$$q_t \leq \max_{j \neq y_t} P_1^j(x_t).$$

Proof. Proceed by induction over rounds. Initially all models have full support. Suppose all preceding updates are well defined, and fix a competing class $j \neq y_t$. By label consistency, any earlier update to class j occurred at an instance different from x_t. By (4), each such update multiplied the probability of x_t under class j by $(1-a_s)/(1-p_s) \in (0,1)$; all other rounds left it unchanged. Therefore $P_t^j(x_t) \leq P_1^j(x_t)$, and taking the maximum over competing classes gives (ii). Consequently,

$$a_t = e^\rho q_t \leq e^\rho \max_{j \neq y_t} P_1^j(x_t) < 1.$$

An update therefore preserves full support by (4); otherwise all models remain unchanged. This closes the induction. □

A.4 Proof of Theorem 2

Proof. On each update round, Lemma 2 and Assumption 1 give

$$r_t \geq e^{\rho} \max_{j \neq y_t} P_1^j(x_t) \geq e^{\rho} q_t = a_t.$$

Since $\Lambda(a_t, p_t) > 0$, Lemma 1 yields

$$\Phi_t - \Phi_{t+1} \geq D_{\mathrm{bin}}(a_t \| p_t) \geq 0.$$

On all other rounds, $\Phi_{t+1} = \Phi_t$. Summing over $t \leq T$ gives

$$\sum_{t \in \mathcal{U}_T} D_{\mathrm{bin}}(a_t \| p_t) \leq \Phi_1 - \Phi_{T+1} \leq \Phi_1,$$

since $\Phi_{T+1} \geq 0$. □

A.5 Proof of Corrolary 1

Proof. Let t be a mistake round. Then $p_t \leq q_t$, including any incorrectly broken tie, so $\gamma_t \leq 0 < \rho$ and an update occurs. Assumption 1 and Lemma 2 give $r_t \geq a_t + \delta$. Since $a_t > p_t$,

$$\begin{aligned}
\Lambda(a_t, p_t) &= \log \frac{a_t}{p_t} + \log \frac{1 - p_t}{1 - a_t} \\
&\geq \log \frac{a_t}{p_t} \\
&= \rho + \log q_t - \log p_t \\
&= \rho - \gamma_t \geq \rho.
\end{aligned}$$

Lemma 1 then gives

$$\Phi_t - \Phi_{t+1} \geq (r_t - a_t)\Lambda(a_t, p_t) \geq \delta\rho$$

on mistake rounds, while every round satisfies $\Phi_t - \Phi_{t+1} \geq 0$ as in Theorem 2. Telescoping,

$$(\#\text{mistakes})\, \delta\rho \leq \Phi_1 - \Phi_{T+1} \leq \Phi_1.$$

□

B Limitations

This work has several limitations. Firstly, the theoretical MDL-PA algorithm is formulated over explicit full-support probability distributions on a finite instance space, whereas the practical algorithm uses adaptive Lempel-Ziv-style compression dictionaries as a practical approximation. Thus, this practical method can only be viewed as a very rough surrogate for the I-projection update in Equation 3 rather than any sort of exact implementation.

Secondly, the mistake-bound analysis relies on a strong realizability assumption and additional uniform slack assumption. These assumptions may fail in real-world data streams.

Third, the empirical evaluation is limited to text and malware sequence classification tasks, so effectiveness of the method and behavior of code-length margins in other domains remains unexplored.

Finally, dictionary growth is controlled indirectly through margin-triggered updates in our experiments, but large datasets may necessitate memory caps and byte eviction policies within the dictionary as the compressor is unable to reference patterns within too large of a window.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The theoretical claims in the abstract and introduction are addressed in §3 while the empirical claims are verified by our empirical results in §5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We describe limitations in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provide assumptions and proofs for all referenced lemmas, corollaries, and theorems in Appendix A.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We implement exactly our algorithm listed in Algorithm Algorithm 1 using the `zstandard` compression library, as described in our experimental setup in §5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification:

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: §5 specifies the datasets, metrics, and hyperparameter selection via validation set for the margin hyperparameter ρ.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report variability across 10 independent random stream orderings in Table 6, where each entry is reported as mean $\pm$ 1 standard deviation across seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We note in §5 that our method is low-resource enough to run on a home computer on with a CPU only, and also report MDL-PA runtimes per dataset.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification:

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.