IN-CONTEXT SEMI-SUPERVISED LEARNING VIA DISCRETE DIFFUSION

Anonymous authors
Paper under double-blind review

ABSTRACT

In-context learning (ICL), which enables a frozen model to adapt to new tasks using labeled demonstrations, has recently attracted considerable attention. However, in many practical settings, labeled demonstrations are scarce and costly to obtain, whereas unlabeled data are abundant. This motivates in-context semi-supervised learning (IC-SSL), where a model leverages both labeled and unlabeled examples together within the context. We propose Diffusion In-Context Learner (DICL), which treats the unknown labels in IC-SSL as discrete latent variables and learns a conditional diffusion process over them. Specifically, at inference time, a bidirectional Transformer iteratively denoises missing labels while keeping the input features and observed labels fixed. During this process, DICL progressively commits its most confident predictions, allowing previously inferred labels to serve as pseudo-labeled context for subsequent predictions. Moreover, the denoising budget provides a natural inference-time compute knob, allowing a single frozen model to trade additional computation for progressively refined predictions. Experiments on in-context classification of Gaussian mixtures and in-context wireless symbol detection consistently demonstrate the effectiveness of DICL.

1 INTRODUCTION

In-context learning (ICL) enables a trained model to learn a new task from demonstrations provided in its input context, without updating its parameters (Brown et al., 2020). ICL was popularized by large language models and has since been studied as a learning mechanism in its own right (Garg et al., 2022). In many theoretical studies of ICL (Akyürek et al., 2023; Von Oswald et al., 2023; Bai et al., 2023; Shen et al., 2025), the problem is formulated as follows. Consider a model A and a prompt $P = \{\mathcal{D}, x_q\}$, where $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n_\ell}$ is a set of demonstrations and x_q is a query input. The inputs x_i and outputs y_i in the demonstrations are assumed to be related according to some underlying task structure. The goal of ICL is for A to infer this relationship from $\mathcal{D}$ entirely through its input context, without any parameter updates, and, when prompted with P, to produce an output y_q for x_q that is consistent with the relationship exhibited by the demonstrations. For example, numerous studies have shown that Transformers trained on synthetic mathematical tasks can learn to perform in-context regression, classification, and estimation (Garg et al., 2022; Akyürek et al., 2023; Von Oswald et al., 2023; Bai et al., 2023; Shen et al., 2025). More recently, Transformers have also been explored as in-context receivers for wireless communications, where the demonstrations consist of labeled pilot symbols and the model performs symbol detection conditioned on the observed pilots (Zecchin et al., 2024; Kunde et al., 2025). Despite differences in the underlying tasks, these formulations share a common feature: the context consists of a set of *labeled* input–output pairs.

In many practical settings, however, labels are the expensive part of a context, while unlabeled inputs from the same task are often available for free: examples include other queries in the same test batch, unannotated samples from a new domain, or payload symbols observed by a receiver through the same channel as its pilots. Semi-supervised learning (SSL) has long shown that such unlabeled inputs can be informative when their distribution reveals structure relevant to the labeling task (Castelli & Cover, 1996). This motivates *in-context semi-supervised learning* (IC-SSL; Fan et al., 2025a; Li et al., 2025; Liu & Yang, 2026), in which the context contains a small number of labeled examples together with a larger set of unlabeled examples, and a frozen model must leverage both to infer the missing labels.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

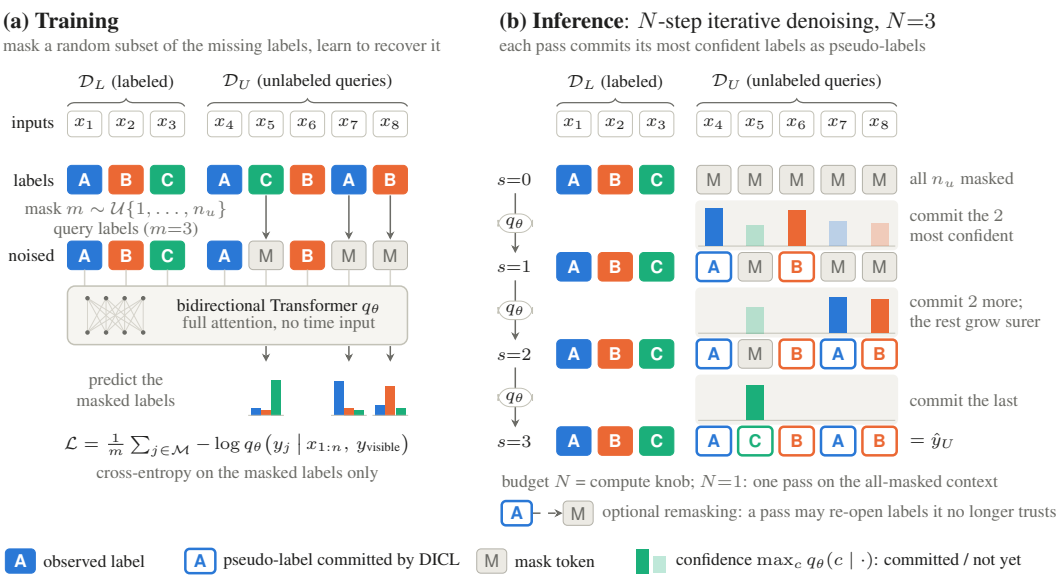

Figure 1: **How DICL works.** A context holds labeled examples $\mathcal{D}_L$ and unlabeled queries $\mathcal{D}_U$; every example is one token that carries its input x_i and either its label or a mask token. **(a)** Training: in a context whose labels are all known, a random number $m \sim \mathcal{U}\{1, \ldots, n_u\}$ of the query labels is masked, and a bidirectional Transformer q_θ learns to predict them from all inputs and all remaining labels (Eq. (1)). **(b)** Inference: every query label starts masked. Each of the N passes of the same frozen q_θ scores the masked positions and commits the $\lceil \text{remaining}/\text{steps left} \rceil$ most confident predictions (full bars) as pseudo-labels, which all later passes condition on, so the remaining predictions become more certain as the context fills in. N=1 is one-shot transductive prediction, so N trades compute for accuracy; optional remasking re-opens committed labels that a later pass no longer trusts (Algorithm 1). Confidences are illustrative.

However, current ICL frameworks each have their own limitations in accommodating this IC-SSL setting (See Table 1). For example, *Labeled-only ICL* discards the unlabeled examples, making it impossible to fully exploit the information contained in those unlabeled examples. *Sequential decision feedback ICL* (DEFINED; Fan et al., 2025d;b) reads the unlabeled examples in order with a causal model and feeds its own hard decisions back as labels, however, in this setting, early predictions never see later examples, errors propagate forward, and inference takes one pass per example. *One-shot transductive IC-SSL* (OICSSL) typically trains a bidirectional model to attend to all examples and predict all missing labels in a single forward pass (Fan et al., 2025a). Although this setting makes more effective use of in-context information than Labeled-only ICL and DEFINED, its key limitation is that the model must resolve all couplings simultaneously. This not only makes it potentially suboptimal for tasks that inherently benefit from iterative reasoning or refinement, but also prevents the model, especially on challenging tasks, from leveraging additional test-time computation to progressively improve its predictions, thereby limiting its ability to achieve test-time scaling. Classical SSL usually resolves the coupling *iteratively*, for example, EM alternates between inferring labels and refitting a model (Dempster et al., 1977; Nigam et al., 2000), and self-training commits confident predictions as pseudo-labels (Lee, 2013), however it needs a hand-specified model of every task.

We propose the **Diffusion In-Context Learner (DICL)**, which moves this iterative inference into the forward passes of a single frozen model. DICL treats the unknown labels as discrete latent variables and learns a conditional masked diffusion process over them (Austin et al., 2021; Sahoo et al., 2024; Shi et al., 2024). During training, a bidirectional Transformer sees contexts in which a random subset of the unlabeled examples' labels is masked and learns to reconstruct them from all inputs and all remaining labels. At inference, every missing label starts masked and DICL denoises them in N steps, committing its most confident predictions at each step so that they serve as pseudo-labeled context for the rest (Figure 1). The budget N is an inference-time compute knob: N=1 is one-shot transductive prediction, and larger N spends more forward passes to refine the predictions. An optional remasking step (Wang et al., 2026) lets DICL revisit earlier commitments.

Our main contributions can be summarized as follows:

- We formulate missing labels in IC-SSL as discrete latent variables and propose DICL, which trains a time-free conditional masked-diffusion model over the missing labels.
- With confidence-ordered iterative denoising, DICL removes the positional asymmetry of causal decision feedback in DEFINED (Fan et al., 2025b) and exposes an inference-time compute axis without changing model weights.
- We provide controlled comparisons against labeled-only ICL, sequential decision feedback, one-shot transductive IC-SSL, and classical semi-supervised methods. Results on two general classification families and two wireless systems consistently favor DICL.

Table 1: What each in-context learner conditions on when it predicts a missing label y_j, $j \in U$ in IC-SSL. $\hat{y}$ are the model's own earlier decisions and $\mathcal{M} \subseteq U$ is the set of labels still masked.

Method	prediction of y_j conditions on	passes
Labeled-only ICL	$\mathcal{D}_L$, x_j	n_u
DEFINED (Fan et al., 2025b)	$\mathcal{D}_L$, $x_{n_l+1:j}$, $\hat{y}_{n_l+1:j-1}$	n_u (sequential)
OICSSL	$\mathcal{D}_L$, $\mathcal{D}_U$	1
DICL (ours)	$\mathcal{D}_L$, $\mathcal{D}_U$, $\hat{y}_{U\setminus\mathcal{M}}$	$1 \le N \le n_u$

2 Related work

In-context learning. A Transformer trained on many tasks can fit a new task from the examples in its context, with no weight update (Brown et al., 2020; Min et al., 2022). On synthetic function classes this ability has been studied in detail (Garg et al., 2022). Trained Transformers can implement gradient descent, ridge regression and algorithm selection in their forward pass (Akyürek et al., 2023; Von Oswald et al., 2023; Bai et al., 2023; Zhang et al., 2024; Shen et al., 2025). Their predictions can also be read as Bayesian inference under the pretraining task distribution (Xie et al., 2022; Panwar et al., 2024).

In-context learning with unlabeled data. Recently, several works have begun to investigate in-context learning with unlabeled data. In language models, many-shot ICL (Agarwal et al., 2024) shows that unsupervised ICL, whose prompts contain only inputs, can already help. Pseudo-labeling methods first label extra demonstrations and then place them in the prompt (Chen et al., 2025c;b). The pseudo-labels there come from the same model in a separate stage, while in DICL they are produced and consumed inside the learned sampler. On the theory side, Li et al. (2025) prove that one-layer linear attention cannot use unlabeled demonstrations. Deeper or looped models, in contrast, can implement EM-style estimators whose error decreases with the number of unlabeled points. Liu & Yang (2026) proved that, with chain-of-thought (CoT) prompting, a multi-layer transformer can effectively emulate an EM algorithm. However, both Li et al. (2025) and Liu & Yang (2026) are primarily theoretical works, showing that looped or chain-of-thought (CoT) Transformers can mimic the EM algorithm. For example, a model trained following the CoT training scheme in Liu & Yang (2026) can only emulate the EM algorithm through a CoT-style iterative procedure. Therefore, in our experiments, we do not compare against the IC-SSL method of Liu & Yang (2026). Fan et al. (2025a) train a Transformer end to end for in-context semi-supervised learning: it builds a context-dependent representation from the unlabeled points and predicts all missing labels in one bidirectional pass. In the causal direction, decision-feedback ICL (Fan et al., 2025b) feeds a detector's own past decisions back as context. This is our DEFINED baseline. It pays one full pass per example and can only use unlabeled examples that come earlier. Recent theoretical works have explored transformers for unsupervised Gaussian mixture models. He et al. (2025a) showed that transformers can implement PCA for clustering two-component GMMs, while He et al. (2025b) demonstrated their ability to approximate Lloyd's algorithm. More recently, Chen et al. (2025a) provided a more efficient construction for approximating EM and showed that transformers can exactly implement cubic tensor power iterations used in spectral methods for GMMs.

Discrete diffusion. Masked (absorbing-state) discrete diffusion (Austin et al., 2021; Campbell et al., 2022; Lou et al., 2024; Sahoo et al., 2024; Shi et al., 2024; Gat et al., 2024) has scaled to diffusion

language models (Nie et al., 2025). Its training objective is an average of any-order autoregressive likelihoods (Uria et al., 2014; Hoogeboom et al., 2022; Ou et al., 2025). Its optimal denoiser does not depend on time (Zheng et al., 2025). ReMDM (Wang et al., 2026) adds remasking so that the number of steps becomes an inference-time compute knob. CARD (Han et al., 2022) also diffuses labels, but for supervised prediction of one output given one input, with no context.

Learned receivers and semi-blind detection. In wireless receivers, the pilot symbols are labeled examples and the payload symbols are unlabeled ones. Transformers trained over channel distributions detect symbols in context from pilots (Zecchin et al., 2024; 2025; Kunde et al., 2025). Detection improves further with decision feedback over the payload (Fan et al., 2025d;b) or with chain-of-thought-style intermediate steps (Fan et al., 2025c).

3 Diffusion In-Context Learner

Problem formulation. In this work, we consider in context classification tasks. A task $\tau \sim P(\tau)$ specifies a distribution P_τ over inputs $x \in \mathcal{X}$ and labels $y \in [C] = \{1, \dots, C\}$. A context contains n_l labeled examples $\mathcal{D}_L = \{(x_i, y_i)\}_{i \in L}$ and n_u unlabeled examples $\mathcal{D}_U = \{x_j\}_{j \in U}$ drawn i.i.d. from P_τ, with $L = \{1, \dots, n_l\}$, $U = \{n_{l+1}, \dots, n\}$ and $n = n_l + n_u$. An in-context learner f_θ is trained once on contexts of many tasks and then frozen; given a new context it predicts the missing labels y_U. We report the query error rate $\mathbb{E}[\frac{1}{n_u} \sum_{j \in U} \mathbf{1}\{\hat{y}_j \neq y_j\}]$ on a grid of (n_l, n_u).

Denoiser. We use a bidirectional masked-label Transformer as the denoiser. Each example is one token, $z_i = W_x x_i + e_i + p_i$, where $e_i = W_y \,\mathrm{onehot}(y_i)$ if y_i is visible and $e_i = e_{[\mathrm{M}]}$, a learned mask embedding, otherwise, and p_i is a learned position embedding. A bidirectional Transformer encoder (Vaswani et al., 2017; Xiong et al., 2020) maps $(z_1, \dots, z_n)$ to logits over $[C]$ at every position. Writing $V \supseteq L$ for the set of visible labels, the output at a masked position $j \notin V$ defines $q_\theta(y_j \mid x_{1:n}, y_V)$. The inputs $x_{1:n}$ and the labels in $\mathcal{D}_L$ are never corrupted.

Forward process and training objective. The forward process is absorbing-state masking of the missing labels only. For a training context whose labels are all known, we draw a masked count $m \sim \mathcal{U}\{1, \dots, n_u\}$ and a uniformly random subset $\mathcal{M} \subseteq U$ of size m, and minimize

$$\mathcal{L}_{\mathrm{DICL}}(\theta) = \mathbb{E}_{\text{context}}\, \mathbb{E}_{m,\mathcal{M}} \Big[\frac{1}{m} \sum_{j \in \mathcal{M}} - \log q_\theta\big(y_j \,\big|\, x_{1:n}, y_{L \cup (U \setminus \mathcal{M})}\big) \Big]. \tag{1}$$

Sampling the count rather than a per-position masking rate makes the training states exactly the states the sampler below can reach, for every budget. One-shot IC-SSL (OICSSL) is the same network trained on $m = n_u$.

Algorithm 1 DICL inference: N-step denoising of the missing labels (optional remasking)

Require: context $(\mathcal{D}_L, \mathcal{D}_U)$, denoiser q_θ, budget $N \leq n_u$, remask fraction $R \in [0, 1)$
$\mathcal{M} \leftarrow U$, $r_0 \leftarrow n_u$ ▷ every missing label starts masked
for $s = 1, \dots, N$ **do**
  $r_s \leftarrow r_{s-1} - \lceil r_{s-1}/(N - s + 1) \rceil$ ▷ labels still masked after step s; $r_N = 0$
  $p_j \leftarrow q_\theta(\cdot \mid x_{1:n}, y_L, \hat{y}_{U \setminus \mathcal{M}})$ for all $j \in U$ ▷ one forward pass
  $\rho \leftarrow \min(\mathrm{round}(R\,(n_u - r_{s-1})),\, r_s)$ ▷ $\rho = 0$ if $R = 0$ or $s = N$
  $\mathcal{R} \leftarrow$ the ρ committed $j \in U \setminus \mathcal{M}$ with the smallest $p_j(\hat{y}_j)$
  $\mathcal{C} \leftarrow$ the $r_{s-1} - r_s + \rho$ masked $j \in \mathcal{M}$ with the largest $\max_c p_j(c)$
  $\hat{y}_j \leftarrow \arg\max_c p_j(c)$ for $j \in \mathcal{C}$; $\mathcal{M} \leftarrow (\mathcal{M} \setminus \mathcal{C}) \cup \mathcal{R}$
end for
return $\hat{y}_U$

Inference. Algorithm 1 (Figure 1b) starts with every missing label masked and, in each of N steps, runs one forward pass and commits the $\lceil \text{remaining}/\text{steps left} \rceil$ masked labels the model is most confident about, in the spirit of confidence-ordered decoding for masked models (Ghazvininejad et al., 2019; Chang et al., 2022; Kim et al., 2025). The committed labels are visible in every later pass. N=1 is a single pass on the all-masked context and $N = n_u$ commits one label per step. Each step costs one pass over the n tokens of the context, so DICL costs N passes whatever n_u is, whereas decision feedback needs n_u sequential passes. Commitments are irreversible in the plain sampler. With remask fraction $R > 0$, a ReMDM-style step (Wang et al., 2026) re-opens the ρ committed

labels that the current pass believes least and commits ρ extra ones, so that each step still lands on the schedule. $R = 0$ is the plain sampler (In most of our experiments, unless otherwise specified, we set ($R = 0$), meaning that no remasking technique is used. This is because, in our current experiments, remasking appears to provide only limited benefits (see Appendix D.5).), the last step never remasks, and remasking is inert for $N \leq 2$.

4 EXPERIMENTAL SETUP

We use four settings: two families of in-context Gaussian-mixture classification, where well-specified classical estimators exist, and two multiple-input multiple-output (MIMO) symbol-detection problems, where pilots are labeled examples and payload observations are unlabeled ones. All details are in Appendices A–C.

GMM ($C = 8$, $D = 20$). Class prior $\pi \sim \mathrm{Dir}(2 \cdot \mathbf{1}_8)$; class means on a sphere of random radius (separation $\sim \mathcal{U}[4, 8]$ within-class scales); one anisotropic Gaussian per class with random orientation. Labeled and unlabeled examples are i.i.d. from the mixture, so a query's class has no labeled example with probability 27.7, 11.3 and 3.2% at $n_l = 8, 16, 32$; no in-context learner can name such a class. Grid $n_l \in \{8, 16, 32\} \times n_u \in \{1, 2, 4, 8, 16, 32\}$ (18 cells).

SubGMM ($C = 8$, $D = 20$). Each class is a mixture of $M_c \sim \mathcal{U}\{1, 2, 3\}$ anisotropic Gaussian sub-classes around a class center (separation $\sim \mathcal{U}[3, 6]$); the sub-class of an example is never revealed. Every labeled set contains each class at least once, but 32.4, 18.7 and 8.6% of the queries ($n_l = 8, 16, 32$; $n_u = 32$) lie in a sub-class without a labeled example, whose class has to be propagated through unlabeled data. Grid $n_l \in \{8, 16, 32\} \times n_u \in \{1, 2, \ldots, 64\}$ (21 cells).

MIMO 2×2 and MIMO 4×4. A transmitter with A antennas sends to a receiver with A antennas ($A = 2$ and 4) over a block of $n_l + n_u$ channel uses. At use i every antenna sends a unit-power QPSK symbol, so the transmitted vector is one of $C = 4^A$ joint symbols ($C = 16$ and 256), and the receiver observes $r_i = Hs_i + w_i \in \mathbb{C}^A$. The channel $H \in \mathbb{C}^{A\times A}$ has i.i.d. $\mathcal{CN}(0, 1)$ entries (Rayleigh fading), is redrawn for every block and stays fixed within it; the noise is $w_i \sim \mathcal{CN}(0, \sigma^2 I)$ with $\mathrm{SNR} = 1/\sigma^2 \sim \mathcal{U}[10, 20]$ dB per block. In IC-SSL terms, the first n_l uses are pilots, whose symbols are known to the receiver (the labeled examples), and the remaining n_u are payload symbols to be detected (the unlabeled examples, which are also the queries). Given H, the inputs of a block form an equal-weight mixture of C Gaussians with centers Hs and shared covariance $\frac{\sigma^2}{2} I$, so the payload reveals the cluster geometry and a few pilots suffice to name the clusters. We evaluate the pilot-starved regime $n_l \in \{2, 4\} \leq A$ with $n_u \in \{1, 2, \ldots, 64\}$ (14 cells per array size). We also test at fixed SNRs of 5 to 25 dB, partly outside the training range, at the reference cell $(n_l, n_u) = (4, 32)$.

Methods. The four learned methods of Table 1 have identical depth, width and heads within a setting: 4 layers (3.2M parameters) for classification and 8 layers (6.4M) for detection, $d = 256$, 8 heads; ICL and DEFINED use a GPT-2-style causal decoder (Radford et al., 2019) over interleaved (x_i, y_i) tokens. DICL is reported at $N{=}1$, at a fixed budget $N{=}T^\star$, and as DICL*, with N chosen per cell on validation. The classical per-context baselines, tuned on validation, are LogReg and QDA on $\mathcal{D}_L$ and semi-supervised MAP-EM with a Gaussian mixture per class ($K^\star$ iterations; EM* chooses K and the number of components per cell) for classification, and LS+ML (a least-squares channel estimate from the pilots, then joint maximum-likelihood detection over all C joint symbols) and EM-ML (semi-blind re-estimation of the channel and noise level from the payload; $K^\star$ iterations, or per cell as EM-ML*) for detection.

Protocol. Within a setting, every learned method trains on one shared set of 5×10^5 (classification) or 10^6 (detection) contexts with n_l, n_u drawn uniformly over the training ranges; OICSSL and DICL receive identical grids (Appendix C). Everything that is selected is selected on validation: each method's checkpoint by its geometric-mean error over the validation grid, $T^\star$ and $K^\star$ by a knee rule (the smallest budget within 0.5% or 2% of the best), and the per-cell choices of DICL*, EM* and EM-ML*. This gives $T^\star = 8, 8, 4, 4$ and $K^\star = 16, 8, 4, 32$ for GMM, SubGMM, MIMO 2×2 and MIMO 4×4. Each cell is scored on 4×10^4 test contexts (every cell a prefix of the same test blocks).

5 RESULTS

Figures 2 and 3 plot the test error of every method against the number of unlabeled examples n_u. The per-cell values are in Tables 2 (SubGMM) and 3 (MIMO 4×4) below, and in Tables 6 (GMM) and 7 (MIMO 2×2) in Appendix D. We call blocks with $n_u \leq 4$ *short* and blocks with $n_u \geq 8$ *long*, and "DICL" refers to the fixed budget $N{=}T^\star$ unless stated otherwise.

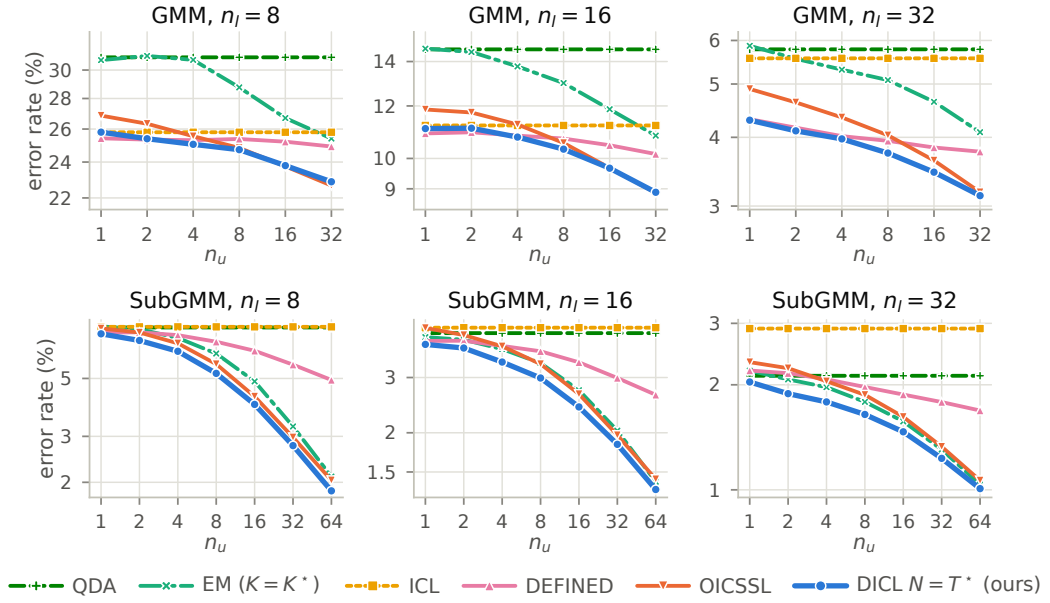

Figure 2: In-context classification: query error (%, log scale) vs. the number of unlabeled examples n_u. GMM (top row, $C{=}8$ Gaussian classes) and SubGMM (bottom row, classes made of 1–3 hidden sub-classes). DICL is shown at the fixed budget $N{=}T^\star{=}8$. The per-cell values, together with LogReg, EM*, DICL $N{=}1$ and DICL*, are in Table 2 (SubGMM) and Table 6 in the appendix (GMM).

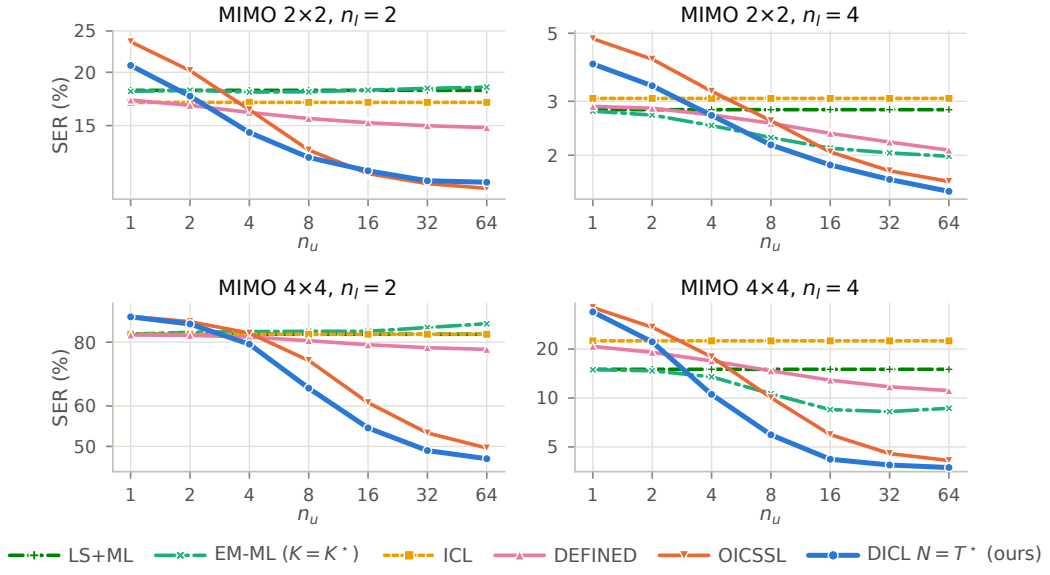

Figure 3: In-context MIMO detection: payload symbol error rate (%, log scale) vs. the number of unlabeled payload symbols n_u. MIMO 2×2 (top row, $C{=}16$ joint symbols) and MIMO 4×4 (bottom row, $C{=}256$). DICL is shown at $N{=}T^\star{=}4$. The per-cell values, together with EM-ML*, DICL $N{=}1$ and DICL*, are in Table 3 (MIMO 4×4) and Table 7 in the appendix (MIMO 2×2).

Table 2: SubGMM: query error (%) per cell (test, 4×10^4 tasks per cell). LogReg, QDA and ICL do not depend on n_u: one value per n_l (the mean over its n_u). Bold: best in the row. At $n_u = 1$ every DICL column is the same one-shot pass.

n_l	n_u	LogReg	QDA	EM K=8	EM*	ICL	DEFINED	OICSSL	DICL N=1	DICL N=8	DICL*
8	1	8.458	7.839	7.917	7.702	7.884	7.493	7.773	**7.412**	**7.412**	**7.412**
	2			7.720	7.554		7.511	7.501	7.089	**6.989**	**6.989**
	4			7.149	7.021		7.326	6.817	6.529	**6.355**	**6.355**
	8			6.230	6.230		6.912	5.675	5.689	**5.228**	**5.228**
	16			4.871	4.871		6.382	4.265	4.724	3.976	**3.963**
	32			3.273	3.232		5.646	2.974	3.704	**2.765**	2.767
	64			2.104	2.055		4.933	2.036	2.863	**1.855**	**1.855**
16	1	6.194	4.155	4.035	4.002	4.324	3.923	4.315	**3.825**	**3.825**	**3.825**
	2			3.959	3.976		3.930	4.097	3.730	**3.725**	3.730
	4			3.696	3.708		3.783	3.773	3.453	**3.361**	**3.361**
	8			3.335	3.335		3.637	3.312	3.142	**2.989**	**2.989**
	16			2.732	2.732		3.354	2.657	2.712	2.418	**2.403**
	32			2.031	2.018		2.989	1.966	2.229	1.837	**1.812**
	64			1.397	1.379		2.642	1.423	1.775	1.319	**1.297**
32	1	3.570	2.123	2.195	2.208	2.898	2.198	2.322	**2.038**	**2.038**	**2.038**
	2			2.073	2.096		2.159	2.231	1.935	**1.886**	**1.886**
	4			1.969	1.985		2.074	2.048	1.826	**1.786**	**1.786**
	8			1.787	1.787		1.971	1.872	1.698	**1.644**	**1.644**
	16			1.571	1.568		1.875	1.619	1.536	1.467	**1.459**
	32			1.305	1.296		1.785	1.330	1.343	1.231	**1.214**
	64			1.036	1.036		1.688	1.064	1.158	1.008	**0.9856**

Table 3: MIMO 4×4: SER (%) per cell (test, 4×10^4 blocks per cell). LS+ML and ICL do not depend on n_u: one value per n_l (the mean over its n_u). Bold: best in the row. At $n_u = 1$ every DICL column is the same one-shot pass.

n_l	n_u	LS+ML	EM-ML K=32	EM-ML*	ICL	DEFINED	OICSSL	DICL N=1	DICL N=4	DICL*
2	1	82.91	82.94	**82.64**	82.91	**82.64**	89.87	89.65	89.65	89.65
	2		83.65	82.78		**82.49**	87.74	87.63	86.83	86.83
	4		83.88	82.91		81.88	83.34	83.27	**79.27**	**79.27**
	8		84.02	83.54		80.54	73.66	72.83	65.01	**61.35**
	16		84.01	84.73		79.07	60.90	63.70	54.34	**51.04**
	32		85.46	83.99		78.03	53.13	58.39	49.07	**48.06**
	64		86.95	84.29		77.44	49.65	56.06	47.30	**46.71**
4	1	15.02	**14.93**	14.97	22.47	20.77	36.01	33.86	33.86	33.86
	2		**14.70**	14.85		19.10	27.27	25.21	22.15	22.15
	4		13.50	13.64		16.91	17.92	16.12	**10.53**	**10.53**
	8		10.61	10.69		14.70	10.05	9.962	5.938	**5.581**
	16		8.482	8.482		12.88	5.953	6.844	4.201	**4.032**
	32		8.251	8.162		11.72	4.553	5.561	3.871	**3.834**
	64		8.646	7.975		11.10	4.124	5.145	**3.734**	**3.734**

5.1 DICL MAKES THE BEST USE OF THE UNLABELED EXAMPLES

Methods that condition on $\mathcal{D}_L$ only (LogReg, QDA, LS+ML and ICL) have flat curves in n_u. Every method that reads $\mathcal{D}_U$ improves as n_u grows, and DICL turns the unlabeled examples into the lowest error: it has the lowest average error of all methods in all four settings, and once the context holds a few unlabeled examples it is the best method in most cells.

The advantage is largest where the unlabeled examples carry the most information. On SubGMM, many queries fall in a sub-class that has no labeled example, and their class can only be inferred through unlabeled examples; here DICL is the best method in every cell (Table 2). In detection with $n_l = 2$ pilots, which cannot identify the channel on their own, the semi-blind EM-ML receiver gains nothing from the payload, while DICL keeps improving as n_u grows (on MIMO 4×4 at $n_u = 64$, 47.3% SER against 84.3% for EM-ML*; Table 3). On GMM the margins are smaller, because much

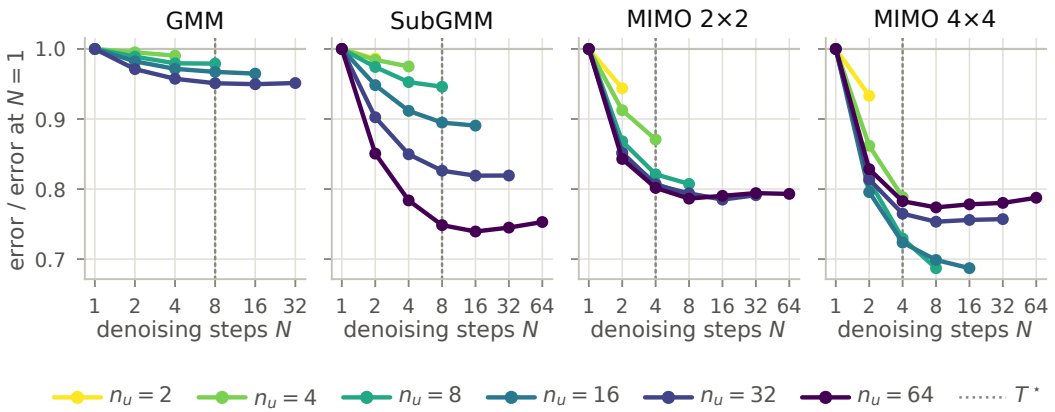

Figure 4: Denoising budget as an inference-time compute knob. DICL error at N steps relative to $N{=}1$ (test), geometric mean over n_l, one line per $n_u \geq 2$ ($N \leq n_u$, so short blocks have short curves); dotted: the validation-selected $T^\star$. Per-n_l curves: Figure 5.

of the error comes from queries whose class has no labeled example, which no in-context learner can name. DICL is not always the best method. With only a handful of unlabeled examples there is little to exploit, and a causal learner or a classical detector is often better; OICSSL, the other bidirectional model, is weaker still in this regime. In a few long blocks, OICSSL is marginally ahead.

Decision feedback uses the unlabeled examples, but less effectively. Unlike ICL, DEFINED improves with n_u, and with one or two unlabeled examples it is often the best learned method. As n_u grows, however, it falls increasingly behind OICSSL and DICL. This follows from its structure (Table 1): each prediction sees only the unlabeled examples that come before it, so early predictions never see later ones, and its hard decisions are fed back without revision, so an early error propagates to every later prediction. OICSSL and DICL let every prediction attend to all of $\mathcal{D}_U$. DICL, like DEFINED, conditions on its own decisions, but it commits them in order of confidence rather than position.

5.2 The gain of DICL comes from iterative denoising

OICSSL and DICL share the architecture, the training data and the training budget; they differ only in the training objective and in inference. DICL $N{=}1$ against OICSSL therefore isolates the objective, and DICL $N{=}T^\star$ against $N{=}1$ isolates iteration with the weights held fixed (Tables 2 and 3, and Tables 6 and 7 in the appendix).

The objective alone does not explain DICL's lead: in a single pass DICL is on par with OICSSL, slightly better on short blocks and slightly worse on long ones, whose all-masked state receives only a small share of DICL's training. Iteration makes the difference. With the same weights, $N{=}T^\star$ improves on $N{=}1$ in every cell with $n_u \geq 2$ in all four settings, and turns the one-shot tie with OICSSL into a lead in every setting.

Figure 4 shows the denoising budget acting as an inference-time compute knob. The error drops fastest over the first few steps and flattens by $N \approx 4$–8; further steps change little. The benefit is larger when there are more unlabeled examples to couple: on SubGMM with many unlabeled examples and on long detection blocks, iteration removes roughly a fifth to a quarter of the one-shot error. A single budget chosen on validation suffices, as choosing N per cell (DICL*) improves on it by at most 1.9%. Since each step is one forward pass, this costs $T^\star$ passes, far fewer than the n_u sequential passes of DEFINED (Table 9).

5.3 Generalization to SNRs outside the training range

All learned receivers are trained only on SNRs in $[10, 20]$ dB; Tables 4 and 5 test them from 5 to 25 dB. Inside the training range DICL is the best receiver at every SNR. Outside it, none of the learned receivers breaks down: their errors change smoothly across the boundary and their relative

Table 4: MIMO 2×2 at $(n_l, n_u) = (4, 32)$: SER (%) vs. SNR. Training range $[10, 20]$ dB.

SNR (dB)	LS+ML	EM-ML K=4	ICL	DEFINED	OICSSL	DICL N=1	DICL N=4
5	28.84	**21.29**	31.23	27.14	23.95	26.96	22.83
10	7.874	4.819	8.577	6.149	5.109	6.121	**4.670**
15	1.968	1.488	2.137	1.551	1.207	1.475	**1.113**
20	0.9393	0.9526	0.9995	0.7473	0.6361	0.7359	**0.5888**
25	1.024	1.132	0.9991	0.8530	**0.6902**	0.8146	0.7358

Table 5: MIMO 4×4 at $(n_l, n_u) = (4, 32)$: SER (%) vs. SNR. Training range $[10, 20]$ dB.

SNR (dB)	LS+ML	EM-ML K=32	ICL	DEFINED	OICSSL	DICL N=1	DICL N=4
5	60.97	**25.65**	61.72	58.77	35.80	44.74	26.94
10	28.22	9.519	33.64	19.64	6.995	9.998	**4.663**
15	13.14	8.124	21.19	10.35	4.076	4.963	**3.859**
20	10.15	7.822	17.30	8.711	3.761	4.310	**3.670**
25	10.01	7.702	16.39	8.485	3.728	4.231	**3.647**

order is largely preserved. At 5 dB, DICL is still the best learned receiver and close to EM-ML, which re-estimates the noise level from each block and is the best receiver there. At 25 dB, DICL remains the best receiver on MIMO 4×4. On MIMO 2×2, every receiver that estimates the channel from the block stops improving above 20 dB, including LS+ML and EM-ML, which involve no training. This points to an identifiability floor (blocks whose pilot matrix is rank-deficient) rather than a failure to extrapolate; at this SNR OICSSL edges slightly ahead of DICL.

6 Conclusion and Future Work

We cast IC-SSL as inference over discrete latent labels and proposed DICL, a masked diffusion model that infers the missing labels in a context through confidence-ordered iterative denoising. Across Gaussian-mixture classification and MIMO detection, DICL makes more effective use of unlabeled examples than labeled-only ICL, sequential decision-feedback ICL, one-shot transductive IC-SSL, and tuned classical estimators, while offering the denoising budget as a simple knob for trading off inference-time computation and performance. One limitation of our current work is that our current evaluation focuses on four synthetic tasks with known generative structure, which provide controlled testbeds but remain substantially simpler than real-world learning problems. Scaling DICL to real-world datasets, larger and more structured label spaces, and foundation-model backbones is therefore a natural next step. More broadly, understanding how iterative latent-label inference behaves under model misspecification, distribution shift, and increasingly complex context structures may help establish diffusion-style inference as a more general paradigm for in-context semi-supervised learning.

AI use statement

In this work, we used generative AI tools for language editing and polishing, as well as with the generation and modification of experimental code. We have reviewed all AI-assisted content. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

Section 3 and Algorithm 1 specify the model, objective and sampler. Appendices A–C describe the task generators, the classical baselines with their tuning, the architectures, and the selection rules. The code is available in the Supplementary Material.

REFERENCES

Rishabh Agarwal, Avi Singh, Lei M Zhang, Bernd Bohnet, Luis Rosias, Stephanie Chan, Biao Zhang, Ankesh Anand, Zaheer Abbas, Azade Nova, et al. Many-shot in-context learning. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? Investigations with linear models. In *International Conference on Learning Representations*, 2023.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Tom Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Vittorio Castelli and Thomas M Cover. The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Transactions on information theory*, 42(6): 2102–2117, 1996.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022.

Zhiheng Chen, Ruofan Wu, and Guanhua Fang. Transformers as unsupervised learning algorithms: A study on gaussian mixtures. *arXiv preprint arXiv:2505.11918*, 2025a.

Zihan Chen, Song Wang, Xingbo Fu, Chengshuai Shi, Zhenyu Lei, Cong Shen, and Jundong Li. From cross-task examples to in-task prompts: A graph-based pseudo-labeling framework for in-context learning. In *Conference on Empirical Methods in Natural Language Processing*, 2025b.

Zihan Chen, Song Wang, Zhen Tan, Jundong Li, and Cong Shen. MAPLE: Many-shot adaptive pseudo-labeling for in-context learning. In *International Conference on Machine Learning*, 2025c.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39(1):1–38, 1977.

Jiashuo Fan, Paul Rosu, Aaron T Wang, Zeyu Michael Li, Lawrence Carin, and Xiang Cheng. In-context semi-supervised learning. *arXiv preprint arXiv:2512.15934*, 2025a.

Li Fan, Wei Shen, Jing Yang, and Cong Shen. Decision feedback in-context learning for wireless symbol detection. *arXiv preprint arXiv:2503.16594*, 2025b.

Li Fan, Peng Wang, Jing Yang, and Cong Shen. Chain-of-thought enhanced shallow transformers for wireless symbol detection. In *GLOBECOM 2025-2025 IEEE Global Communications Conference*, pp. 417–422. IEEE, 2025c.

Li Fan, Jing Yang, and Cong Shen. Decision feedback in-context symbol detection over block-fading channels. In *ICC 2025-IEEE International Conference on Communications*, pp. 3785–3790. IEEE, 2025d.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? A case study of simple function classes. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T Q Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, 2024.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing*, pp. 6112–6121, 2019.

Xizewen Han, Huangjie Zheng, and Mingyuan Zhou. CARD: Classification and regression diffusion models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Yihan He, Yuan Cao, Hong-Yu Chen, Dennis Wu, Jianqing Fan, and Han Liu. Learning spectral methods by transformers. *arXiv preprint arXiv:2501.01312*, 2025a.

Yihan He, Hong-Yu Chen, Yuan Cao, Jianqing Fan, and Han Liu. Transformers versus the em algorithm in multi-class clustering. *arXiv preprint arXiv:2502.06007*, 2025b.

Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. In *International Conference on Learning Representations*, 2022.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *International Conference on Machine Learning*, 2025.

Vishnu Teja Kunde, Vicram Rajagopalan, Chandra Shekhara Kaushik Valmeekam, Krishna Narayanan, Jean-Francois Chamberland, Dileep Kalathil, and Srinivas Shakkottai. Transformers are provably optimal in-context estimators for wireless communications. In *International Conference on Artificial Intelligence and Statistics*, 2025.

Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop on Challenges in Representation Learning*, 2013.

Yingcong Li, Xiangyu Chang, Muti Kara, Xiaofeng Liu, Amit Roy-Chowdhury, and Samet Oymak. When and how unlabeled data provably improve in-context learning. *arXiv preprint arXiv:2506.15329*, 2025.

Renpu Liu and Jing Yang. Unlabeled data can provably enhance in-context learning of transformers. *Advances in Neural Information Processing Systems*, 38:91895–91932, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *International Conference on Machine Learning*, 2024.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. MetaICL: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 2791–2809, 2022.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2):103–134, 2000.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *International Conference on Learning Representations*, 2025.

Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the Bayesian prism. In *International Conference on Learning Representations*, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Wei Shen, Ruida Zhou, Jing Yang, and Cong Shen. On the training convergence of transformers for in-context classification of gaussian mixtures. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=iSyB2yYaMx.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

David Tse and Pramod Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *International Conference on Machine Learning*, pp. 467–475, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174, 2023.

Guanghan Wang, Yair Schiff, Subham Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. *Advances in Neural Information Processing Systems*, 38: 147282–147339, 2026.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit Bayesian inference. In *International Conference on Learning Representations*, 2022.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533, 2020.

Matteo Zecchin, Kai Yu, and Osvaldo Simeone. In-context learning for mimo equalization using transformer-based sequence models. In *2024 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1573–1578. IEEE, 2024.

Matteo Zecchin, Tomer Raviv, Dileep Kalathil, Krishna Narayanan, Nir Shlezinger, and Osvaldo Simeone. In-context learning for gradient-free receiver adaptation: Principles, applications, and theory. *arXiv preprint arXiv:2506.15176*, 2025.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024.

Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. In *International Conference on Learning Representations*, 2025.

APPENDIX

This appendix provides additional details and results of our experiments. Appendix A specifies the four task families: how GMM and SubGMM tasks and MIMO blocks are generated, the training, validation and test splits, and how the evaluation cells are built. Appendix B describes the classical baselines (LogReg, QDA and semi-supervised EM for classification; LS+ML and EM-ML for detection) and how they are tuned. Appendix C gives the architectures, training objectives and optimization of the four learned methods, their training budgets, and the validation protocol that selects checkpoints, $T^\star$, $K^\star$ and the per-cell choices. Appendix D reports additional results: the per-cell errors on GMM and MIMO 2×2, the value of the unlabeled examples, the denoising-budget curves for every labeled-set size, the inference cost, and remasking.

A TASK FAMILIES

GMM and SubGMM. A task draws $\pi \sim \mathrm{Dir}(2 \cdot \mathbf{1}_C)$ and a separation $s \sim \mathcal{U}[s_{\min}, s_{\max}]$, places the class centers at $\mu_c = s\, u_c$ with u_c uniform on the unit sphere in $\mathbb{R}^D$, and gives every Gaussian component a covariance $\Sigma = \kappa^2 R\,\mathrm{diag}(\lambda)R^\top$ with R Haar-distributed and $\log\lambda \sim \mathcal{N}(0, 0.6^2)$ renormalized to unit geometric mean. All lengths are in units of the within-cluster scale and multiplied by one global constant so that each feature has unit variance. GMM uses one component per class ($\kappa = 1$, $[s_{\min}, s_{\max}] = [4, 8]$), so semi-supervised Gaussian EM is correctly specified. SubGMM draws $M_c \sim \mathcal{U}\{1, 2, 3\}$ sub-classes per class with centers at distance 1.5 from μ_c in random directions, weights $\sim \mathrm{Dir}(2 \cdot \mathbf{1}_{M_c})$ and $\kappa = 0.7$ ($[s_{\min}, s_{\max}] = [3, 6]$). In GMM the labeled examples are i.i.d., and a query's class has no labeled example with probability $\mathbb{E}\sum_c \pi_c(1-\pi_c)^{n_l} = 3360/((14+n_l)(15+n_l)(16+n_l))$ for $C = 8$, i.e. 27.7, 11.3 and 3.2% at $n_l = 8, 16, 32$. Since the class names are exchangeable, no in-context learner can do better than guess among unseen names for such a query; the Bayes rule with the true parameters (error below 0.2%) is therefore not an attainable floor. In SubGMM the first C labeled examples are a random permutation of the classes and the rest are i.i.d., so every class is named, while 32.4, 18.7 and 8.6% of the queries at $n_l = 8, 16, 32$ ($n_u = 32$) lie in a sub-class without a labeled example. Training contexts draw $n_l \sim \mathcal{U}\{8..32\}$ and $n_u \sim \mathcal{U}\{1..32\}$ (GMM) or $\mathcal{U}\{1..64\}$ (SubGMM); 5×10^5 training, 2×10^4 validation and 4×10^4 test tasks.

MIMO detection. $H \in \mathbb{C}^{A\times A}$ has i.i.d. $\mathcal{CN}(0, 1)$ entries and is redrawn per block (Tse & Viswanath, 2005); each antenna sends a unit-power QPSK symbol, pilots and payload alike, and $w \sim \mathcal{CN}(0, \sigma^2 I)$ with $\mathrm{SNR} = 1/\sigma^2 \sim \mathcal{U}[10, 20]$ dB per block. The label is the index of the joint symbol among $C = 4^A$. Training blocks draw $n_l \sim \mathcal{U}\{2..8\}$ and $n_u \sim \mathcal{U}\{1..64\}$; 10^6 training, 2×10^4 validation and 4×10^4 test blocks, plus fixed-SNR test sets at 5, 10, 15, 20 and 25 dB for the reference cell. With $n_l \le A$ the pilot matrix is often singular; the least-squares moments are accumulated in double precision, and labels are stored in 16-bit integers for $C = 256$.

Every cell is a prefix of the same blocks. Validation and test contexts are generated at the maximum length, and cell (n_l, n_u) uses the first n_l examples as labeled and the next n_u as unlabeled, so all methods and all cells are scored on the same tasks.

B CLASSICAL BASELINES

Classification. All estimators are fitted per context, batched on the GPU, in exact fp32. LogReg is ℓ_2-regularized multinomial logistic regression on $\mathcal{D}_L$. QDA fits one Gaussian per class on $\mathcal{D}_L$ under a normal-inverse-Wishart prior. EM is semi-supervised MAP-EM over a Gaussian mixture (labeled points keep their class, unlabeled points receive soft responsibilities); $K = 0$ iterations reproduces QDA exactly. The ridge and prior hyper-parameters of all three are tuned on the validation split. EM's iteration count $K^\star$ is the knee (0.5% tolerance) of its validation curve; EM* chooses K and, on SubGMM, the number of components per class $M \in \{1, 2, 3\}$ per cell on validation. On SubGMM, $M = 1$ with a strong covariance prior is the best fixed model; more components, or the true sub-class counts, are worse. Its covariance-prior strength sits at the edge of the tuned grid ($\nu_0 = 1000$), but widening the grid from 160 changed the validation error by only 0.8%.

Detection. LS+ML estimates $\hat{H}$ by least squares from the pilots (minimum-norm when the pilot matrix is singular) and detects each payload symbol by $\arg\min_s \|y - \hat{H}s\|^2$ over the C joint symbols, the strongest detector for a given channel estimate. EM-ML alternates an E-step over the C joint symbols at every payload position with an M-step that refits $\hat{H}$ from the pilots and the payload's soft moments and re-estimates σ^2, then detects by joint ML; soft and decision-directed E-steps are both run, $K = 0$ reproduces LS+ML, and σ^2 is never given (it is initialized from the pilot residual). $K^\star$ is the knee (2% tolerance) at the reference cell; EM-ML* chooses K and the E-step per cell on validation.

C Architectures, training and selection

Architectures. OICSSL and DICL share one pre-LN Transformer encoder (Xiong et al., 2020) with one token per example (Section 3), learned position embeddings, GELU MLPs of width $4d$ and a final layer norm; ICL and DEFINED share a GPT-2 decoder (Radford et al., 2019) over interleaved tokens $(x_1, y_1, \ldots, x_n, y_n)$ whose prediction for y_i is read at the x_i token, so it never sees its own label. The two backbones have the same depth, width, heads and MLP ratio (4 layers for classification, 8 for detection; $d = 256$, 8 heads), i.e. 3.2M and 6.4M parameters.

Objectives. ICL: causal cross-entropy on positions $1..n_l{+}1$ of each block, i.e. every labeled-prefix length. DEFINED: causal cross-entropy on every position with true labels fed back (teacher forcing), optionally followed by a decision-feedback phase in which the query labels are replaced by the model's own decisions (two fixed-point sweeps; loss $0.7\,\mathrm{CE}_{\text{fed back}} + 0.3\,\mathrm{CE}_{\text{true}}$); at inference the queries are read in order and each hard decision is fed back before the next is read. OICSSL: cross-entropy on all query positions with all query labels masked. DICL: Eq. (1).

Optimization. AdamW (Loshchilov & Hutter, 2019) ($\beta = (0.9, 0.95)$, no weight decay, gradient clipping at 1), batch 512, linear warm-up and cosine decay to 5% of the peak learning rate, bf16 autocast for training and fp32 for every evaluation. Each run keeps its final checkpoint and its best checkpoint on one validation cell; both compete in selection.

Budgets. Within a setting the methods receive comparable GPU time rather than equal steps, because their per-step costs differ several-fold. OICSSL and DICL always receive identical grids. On the classification settings, where 5×10^5 stored tasks are few enough for OICSSL and ICL to overfit, the grids include short runs whose cosine schedule anneals fully, so that an early-stopped baseline is available. On MIMO $4{\times}4$, ICL received a second round of longer runs; its validation error bottoms out at 60k steps in every run and then rises, i.e. it is limited by the shared training set, not by its step budget.

Selection. Each method's checkpoint minimizes the geometric-mean validation error over the full evaluation grid (4000 validation contexts); selecting on a single cell instead would favour checkpoints that specialize in one block length. Budgets and per-cell choices use all 2×10^4 validation contexts. $T^\star$ is the knee of the validation geometric mean over the cells with $n_u \geq 8$, capped at 8, with 0.5% tolerance (classification), or the knee at the reference cell $(4, 32)$ with 2% tolerance (detection). The per-cell choices of DICL* (validation argmin of N), EM* and EM-ML* (validation knee of K) are all made on validation; none of them looks at test data.

D Additional results

This appendix collects results that support Section 5: the per-cell results of the two settings that are not tabulated in the main text (D.1), how much each method gains from the unlabeled examples (D.2), the denoising-budget curves for every labeled-set size (D.3), the inference cost (D.4) and remasking (D.5).

D.1 Per-cell results on GMM and MIMO $2{\times}2$

Tables 6 and 7 complement Tables 2 and 3 in the main text. They list the per-cell errors behind the GMM panels of Figure 2 and the MIMO $2{\times}2$ panels of Figure 3, together with the columns that are not plotted there (LogReg, EM* or EM-ML*, DICL $N{=}1$ and DICL*). On GMM, DICL is the best method in most cells; the exceptions are DEFINED with one or two unlabeled examples and

Table 6: GMM: query error (%) per cell (test, 4×10^4 tasks per cell). LogReg, QDA and ICL do not depend on n_u: one value per n_l (the mean over its n_u). Bold: best in the row. At $n_u = 1$ every DICL column is the same one-shot pass.

n_l	n_u	LogReg	QDA	EM K=16	EM*	ICL	DEFINED	OICSSL	DICL N=1	DICL N=8	DICL*
8	1	31.29	30.94	30.73	30.15	25.80	**25.42**	26.86	25.81	25.81	25.81
	2			31.05	31.05		**25.34**	26.34	25.57	25.40	25.40
	4			30.75	30.75		25.31	25.55	25.20	**25.05**	25.08
	8			28.76	28.90		25.38	24.84	24.91	**24.73**	**24.73**
	16			26.70	26.67		25.22	**23.75**	24.12	23.80	**23.75**
	32			25.41	25.37		24.92	**22.67**	23.45	22.88	22.81
16	1	15.83	14.60	14.64	14.64	11.21	**10.91**	11.85	11.09	11.09	11.09
	2			14.47	14.38		**10.95**	11.73	11.12	11.11	11.11
	4			13.76	13.76		10.82	11.25	10.92	**10.77**	10.81
	8			12.99	12.99		10.71	10.55	10.56	**10.33**	10.34
	16			11.86	11.86		10.47	9.676	10.00	**9.667**	**9.667**
	32			10.82	10.86		10.15	**8.876**	9.413	8.892	8.892
32	1	7.122	5.781	5.867	5.835	5.568	4.315	4.897	**4.297**	**4.297**	**4.297**
	2			5.557	5.526		4.165	4.630	4.167	**4.110**	**4.110**
	4			5.310	5.282		4.020	4.348	4.011	**3.974**	**3.974**
	8			5.082	5.059		3.942	4.034	3.875	**3.745**	**3.745**
	16			4.641	4.641		3.833	3.631	3.641	3.457	**3.425**
	32			4.087	4.087		3.767	3.180	3.358	3.134	**3.105**

Table 7: MIMO 2×2: SER (%) per cell (test, 4×10^4 blocks per cell). LS+ML and ICL do not depend on n_u: one value per n_l (the mean over its n_u). Bold: best in the row. At $n_u = 1$ every DICL column is the same one-shot pass.

n_l	n_u	LS+ML	EM-ML K=4	EM-ML*	ICL	DEFINED	OICSSL	DICL N=1	DICL N=4	DICL*
2	1	18.14	18.04	17.58	**17.00**	17.22	23.59	20.75	20.75	20.75
	2		18.15	17.03		**16.73**	20.19	18.73	17.59	17.59
	4		17.98	15.92		16.10	16.34	16.72	**14.45**	**14.45**
	8		18.00	16.60		15.58	13.14	15.13	12.63	**12.27**
	16		18.16	17.09		15.23	11.58	13.97	11.75	**11.45**
	32		18.34	17.42		14.99	**10.96**	13.26	11.13	10.98
	64		18.45	17.46		14.83	**10.69**	12.97	11.05	10.91
4	1	2.816	**2.785**	2.790	3.066	2.885	4.800	3.970	3.970	3.970
	2		2.705	**2.639**		2.846	4.116	3.551	3.370	3.370
	4		2.500	**2.368**		2.708	3.235	3.076	2.698	2.698
	8		2.283	**2.049**		2.544	2.593	2.676	2.161	2.150
	16		2.109	1.984		2.358	2.050	2.398	1.859	**1.812**
	32		2.036	2.036		2.208	1.779	2.155	1.667	**1.641**
	64		1.984	1.938		2.078	1.642	2.020	**1.524**	**1.524**

OICSSL, by less than 1%, in a few long blocks. On MIMO 2×2, a causal learner or EM-ML is best with one or two payload symbols, EM-ML* remains best with $n_l = 4$ pilots up to $n_u = 8$, OICSSL is marginally ahead in the two longest blocks with $n_l = 2$, and DICL is best in all other cells.

D.2 Value of the unlabeled examples

Table 8 measures how much each method gains from the unlabeled examples: the ratio of its error at $n_u = 1$ to its error at the largest n_u, for every labeled-set size. Methods that ignore $\mathcal{D}_U$ stay at 1 by construction. The bidirectional learners (OICSSL and DICL) gain the most, EM gains comparably on the classification families but nothing at $n_l = 2$ in detection (ratios below 1), and DEFINED gains much less than the bidirectional learners in every setting. The ratio rewards a poor start as much as a good finish: OICSSL often has the largest ratio because its one-shot error at $n_u = 1$ is the highest, so the ratio should be read together with the absolute errors in the per-cell tables.

Table 8: Value of the unlabeled block: error at $n_u = 1$ divided by error at the largest n_u (32 for GMM, 64 otherwise), per labeled-set size. Labeled-only methods stay at 1.

Setting	n_l	QDA / LS+ML	EM / EM-ML ($K^\star$)	ICL	DEFINED	OICSSL	DICL $N{=}1$	DICL $N{=}T^\star$
GMM	8	1.00	1.21	1.00	1.02	1.18	1.10	1.13
	16	1.00	1.35	1.00	1.07	1.34	1.18	1.25
	32	1.00	1.44	1.00	1.15	1.54	1.28	1.37
SubGMM	8	1.00	3.76	1.00	1.52	3.82	2.59	4.00
	16	1.00	2.89	1.00	1.48	3.03	2.15	2.90
	32	1.00	2.12	1.00	1.30	2.18	1.76	2.02
MIMO $2{\times}2$	2	1.00	0.98	1.00	1.16	2.21	1.60	1.88
	4	1.00	1.40	1.00	1.39	2.92	1.97	2.60
MIMO $4{\times}4$	2	1.00	0.95	1.00	1.07	1.81	1.60	1.90
	4	1.00	1.73	1.00	1.87	8.73	6.58	9.07

Table 9: Per-symbol wall-clock cost and SER at $(n_l, n_u) = (4, 32)$, measured while scoring the test grid on one GPU (useful as ratios, not as deployment latency). DICL* and EM-ML* are billed at the budget chosen for this cell.

	MIMO $2{\times}2$		MIMO $4{\times}4$	
Receiver	μs/sym.	SER (%)	μs/sym.	SER (%)
LS+ML	0.059	2.816	0.106	15.02
EM-ML ($K{=}K^\star$)	0.164	2.036	1.805	8.251
EM-ML*	0.164	2.036	0.538	8.162
ICL	4.011	3.066	4.016	22.47
DEFINED	52.147	2.208	52.134	11.72
OICSSL	0.418	1.779	0.423	4.553
DICL $N{=}1$	0.421	2.155	0.432	5.561
DICL $N{=}T^\star$	1.651	1.667	1.688	3.871
DICL*	3.291	1.641	3.363	3.834

D.3 Denoising budget for every labeled-set size

Figure 5 breaks Figure 4 down by labeled-set size and plots the absolute error instead of the ratio to $N{=}1$. The behavior described in Section 5.2 holds for every n_l: the error falls over the first few steps and then flattens, the drop is larger for longer blocks, and the validation-selected $T^\star$ (dotted) lies at the start of the flat region. Beyond the best budget, a few of the longest blocks rise slightly (at most $1.044\times$), as later commitments condition on more of the model's own guesses.

D.4 Inference cost

Table 9 reports the per-symbol wall-clock cost of every receiver at the reference cell $(n_l, n_u) = (4, 32)$, together with its SER. DICL at $N{=}T^\star$ costs 3.9–$4.0\times$ as much as OICSSL, i.e. roughly one pass per step, and is 31–$32\times$ cheaper than DEFINED, whose sequential feedback needs one full-block pass per symbol. The comparison with the classical receivers depends on the array size: joint ML costs 4^A metric evaluations per symbol, so EM-ML is $10.1\times$ cheaper than DICL at $2{\times}2$ but already $1.07\times$ more expensive at $4{\times}4$, while the cost of a network pass is essentially unchanged (0.418 vs. $0.423\,\mu$s per symbol for OICSSL).

D.5 Remasking

With a remask fraction $R > 0$, Algorithm 1 re-opens at every step the committed labels that the current pass trusts least (Section 3). We sweep $R \in \{0.1, 0.2, 0.3, 0.5\}$ (and $R = 0.05$ for detection) jointly with $N \in \{4, 8, 16\}$ on the long blocks. Table 10 reports DICL's error relative to the plain sampler ($R = 0$) at the same N, as a geometric mean over the swept cells of each setting.

Figure 6 shows the same ratios separately for every labeled-set size, as a function of R with one line per budget N.

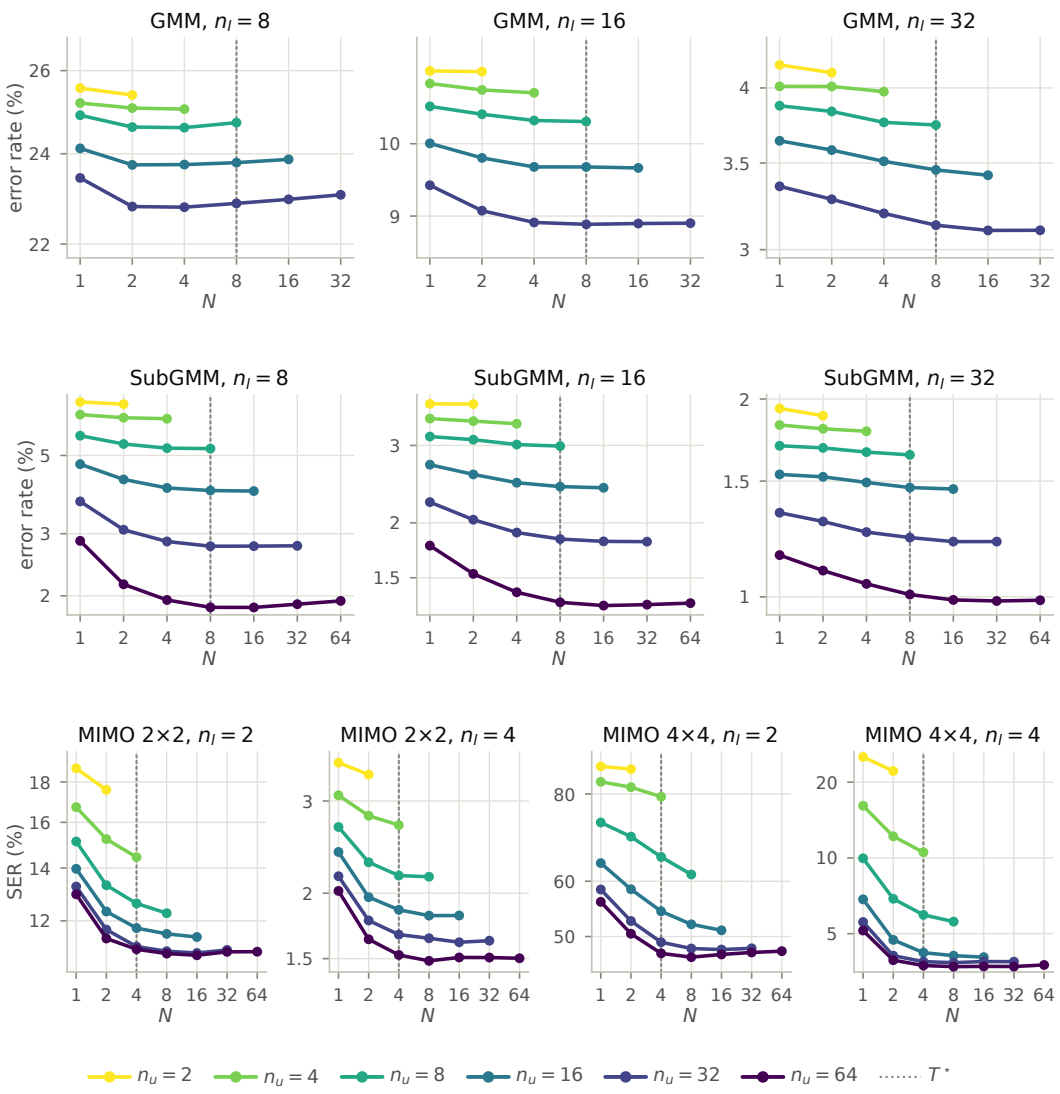

Figure 5: DICL error vs. the number of denoising steps N for every labeled-set size (test), one line per $n_u \geq 2$ ($N \leq n_u$); dotted: $T^\star$.

Table 10: Remasking (test): DICL error at remask fraction R over the plain sampler at the same N, geometric mean over the swept cells with $n_u \geq 8$ in which $N \leq n_u$ ("cells"). Bold: best R per N. The detection sweep also contains $R = 0.05$.

Setting	N	cells	R=0.05	R=0.1	R=0.2	R=0.3	R=0.5
GMM	4	9	–	0.997	0.994	**0.993**	1.005
	8	9	–	0.997	0.998	0.995	**0.992**
	16	6	–	0.995	0.994	**0.993**	**0.993**
SubGMM	4	12	–	0.986	0.979	**0.975**	1.029
	8	12	–	0.989	0.999	0.989	**0.988**
	16	9	–	0.990	0.983	**0.982**	0.986
MIMO 2×2	4	6	0.998	**0.995**	**0.995**	0.997	1.013
	8	6	0.997	0.996	0.999	0.998	**0.992**
	16	4	0.997	0.997	0.999	0.992	**0.986**
MIMO 4×4	4	6	0.994	0.991	0.989	**0.987**	0.995
	8	6	0.995	0.992	0.988	0.987	**0.984**
	16	4	0.998	0.997	0.990	0.985	**0.982**

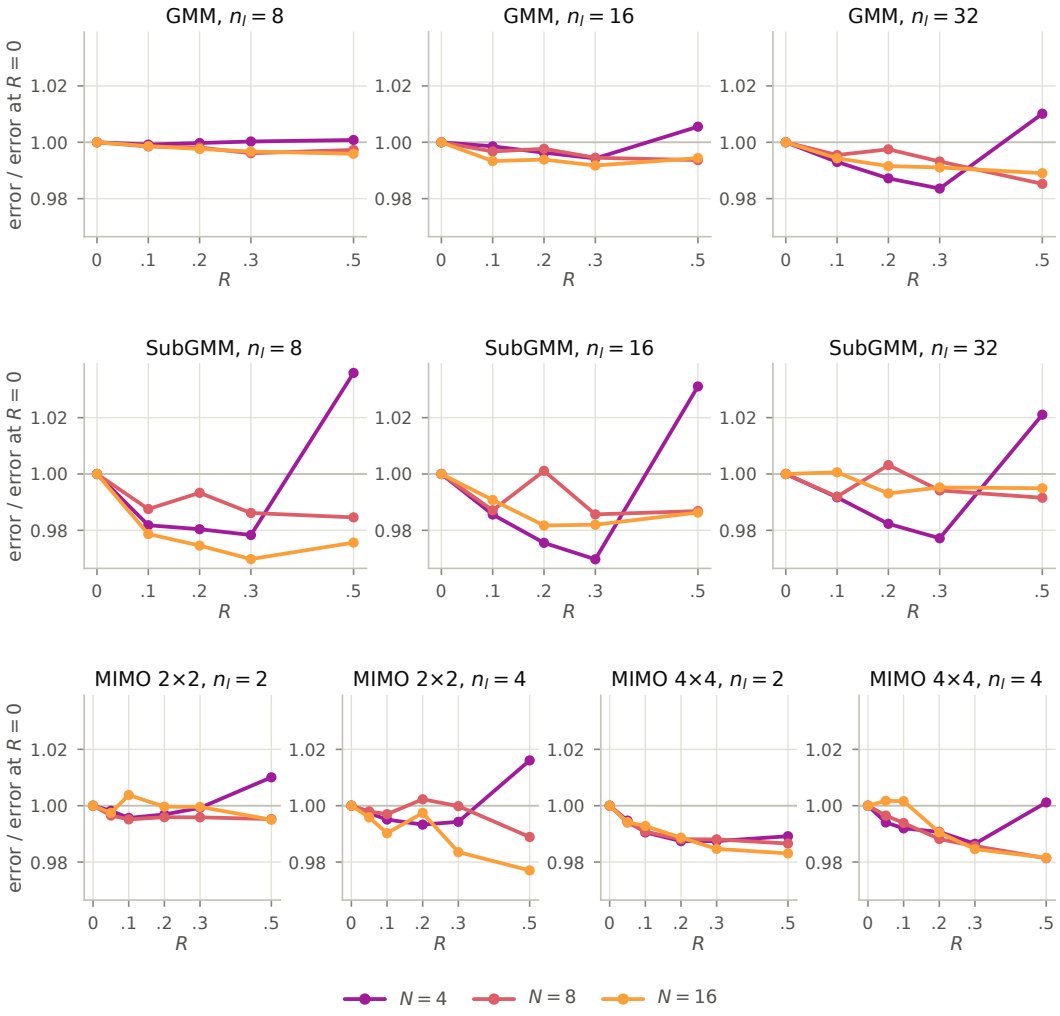

Figure 6: Remasking for every labeled-set size: error at remask fraction R over the plain sampler at the same N (test), geometric mean over the swept $n_u \geq 8$.

The effect of remasking is small everywhere, between 0.975 and $1.029\times$ over the whole sweep. The best (N, R) gains 0.8–2.5%, and large fractions hurt at small budgets, where too few steps remain to settle the re-opened labels (SubGMM, $N{=}4$, $R{=}0.5$: $1.029\times$). Selected on validation, remasking improves DICL on the long blocks by $1.008\times$ on GMM ($N^\star{=}8$, $R^\star{=}0.5$; better in 6 of 9 cells) and by 1.2% on SubGMM ($N^\star{=}8$, $R^\star{=}0.5$). For detection, where (N, R) is selected per cell against the best plain budget, it helps in 3 of 6 cells on MIMO $2{\times}2$ (median ratio 1.000) and in 6 of 6 on MIMO $4{\times}4$ (median 0.985). The plain sampler's commitments are rarely wrong in a way a later pass can detect, so re-opening them buys little; all other numbers in this paper use $R = 0$.