Advancing Image Classification with Discrete Diffusion Classification Modeling

Supplementary Material

A. Background on Transition Matrices

In this section, we present the theoretical background and intuition underlying the transition rate matrices used in our diffusion processes. These matrices form the core of our framework, governing the temporal evolution of probabilities through a continuous-time Markov process.

A.1. Theoretical Analysis of Transition Rates

We begin by generalizing our diffusion framework to an arbitrary target probability vector. Let $q_0 \in \mathbb{R}^K$ denote the target distribution, and let $q_t \in \mathbb{R}^K$ represent its time-dependent evolution for $t \in [0, 1]$. Assume that q_t evolves according to the linear ordinary differential equation,

$$\frac{dq_t}{dt} = R_t q_t \ , \tag{12}$$

where $R_t \in \mathbb{R}^{K\times K}$ is the transition rate matrix.

Following Anderson [1] and Campbell et al. [5], the transition rate matrix is defined by,

$$R_t(i, j) := \lim_{\Delta t\to 0} \frac{q_{t|t-\Delta t}(j|i) - \delta_{i,j}}{\Delta t} \ , \tag{13}$$

with $q_{t|t-\Delta t}(j|i)$ representing the infinitesimal probability of transitioning from state i at time $t - \Delta t$ to state j at time t. For $i \neq j$, $R_t(i, j)$ thus quantifies the instantaneous rate of transition from state i to state j; higher values correspond to faster expected transitions.

By construction, $R_t(i, j) \geq 0$ for $i \neq j$ and $R_t(j, j) = -\sum_{i\neq j} R_t(i, j) \leq 0$, ensuring that each column of R_t sums to zero. Therefore, R_t preserves the total probability mass when applied to q_t and defines a valid Markov process generator.

Suppose R_t takes the simple structured form $R_t = \sigma_t R$, with $R \in \mathbb{R}^{K\times K}$ being a time-invariant transition rate matrix and σ_t is a strictly decreasing non-negative scaling function satisfying $\sigma_0 > 0$ and $\sigma_1 = 0$. Under this assumption, Equation (12) admits a closed-form solution. Below is a proof.

Theorem 1 (Closed-Form Solution for the Discrete Markovian Process)**.** *Let $q_t \in \mathbb{R}^K$ satisfy Equation (12) and let $R = U\Lambda U^{-1}$ be the eigendecomposition of R, where $U \in \mathbb{R}^{K\times K}$ denotes the matrix of eigenvectors and $\Lambda \in \mathbb{R}^{K\times K}$ is the diagonal matrix of eigenvalues. Define $\overline{\sigma}_t := \int_0^t \sigma_s \, ds$. Then,*

$$q_t = U \exp\left(\overline{\sigma}_t \Lambda\right) U^{-1} q_0 \ . \tag{14}$$

Proof. Assume $q_t = \exp(\overline{\sigma}_t R)q_0$. We first show that this form satisfies the given differential equation.

Differentiating with respect to t,

$$\frac{dq_t}{dt} = \frac{d}{dt}\left(\exp(\overline{\sigma}_t R)\right) q_0 \tag{15}$$

$$= \exp(\overline{\sigma}_t R)\frac{d}{dt}(\overline{\sigma}_t R)q_0 \tag{16}$$

$$= \exp(\overline{\sigma}_t R)\sigma_t R q_0 = \exp(\overline{\sigma}_t R)R_t q_0 \ . \tag{17}$$

Since the matrices $\exp(\overline{\sigma}_t R)$ and R_t are polynomials in R, they are diagonalizable and therefore commute. It then follows that,

$$\frac{dq_t}{dt} = \exp(\overline{\sigma}_t R)R_t q_0 = R_t \exp(\overline{\sigma}_t R)q_0 = R_t q_t \ , \tag{18}$$

thus confirming the assumption made,

$$q_t = \exp(\overline{\sigma}_t R)q_0 \ . \tag{19}$$

Additionally, Appendix E of Campbell et al. [5] establishes that,

$$\exp(\overline{\sigma}_t R) = \sum_{k=0}^{\infty} \frac{1}{k!} \left(\overline{\sigma}_t R\right)^k \tag{20}$$

$$= \sum_{k=0}^{\infty} \frac{1}{k!} \left(U\Lambda U^{-1}\overline{\sigma}_t\right)^k \tag{21}$$

$$= \sum_{k=0}^{\infty} \frac{1}{k!} U \left(\Lambda\overline{\sigma}_t\right)^k U^{-1} \tag{22}$$

$$= U \left\{\sum_{k=0}^{\infty} \frac{1}{k!} \left(\Lambda\overline{\sigma}_t\right)^k\right\} U^{-1} \tag{23}$$

$$= U \exp\left(\overline{\sigma}_t\Lambda\right) U^{-1} \ . \tag{24}$$

Therefore, the stated solution follows. □

When R_t does not adopt the form $\sigma_t R$, for instance, in the case of the reversal rate matrix $\overline{R}_t$ introduced in Section 4.2, an approximate solution can be obtained via Euler discretization with sufficiently small time increment $\Delta t > 0$. In this approach, the infinitesimal transition matrix is defined as $Q_t := I + R_t\Delta t$. The probability distribution is then updated according to $q_{t+\Delta t} \approx Q_t q_t$, where each entry satisfies $Q_t(i, j) \geq 0$, and the columns of Q_t sum to one, thereby ensuring conservation of total probability mass when applied to q_t. Below is a proof for this solution.

Theorem 2 (Approximated Solution for the Discrete Markovian Process)**.** *Let $q_t \in \mathbb{R}^K$ satisfy Equation (12).*

For sufficiently small time increment $\Delta t > 0$, the first-order Euler approximation yields

$$q_{t+\Delta t} = Q_t q_t + \mathcal{O}(\Delta t^2)\,, \tag{25}$$

where $\mathcal{O}(\Delta t^2)$ denotes a remainder term that vanishes quadratically as $\Delta t \to 0$.

Proof. By the definition of the time derivative, we have,

$$\frac{dq_t}{dt} = \lim_{\Delta t \to 0} \frac{q_{t+\Delta t} - q_t}{\Delta t} = R_t q_t.$$

Consequently, for sufficiently small $\Delta t > 0$,

$$\frac{q_{t+\Delta t} - q_t}{\Delta t} = R_t q_t + \mathcal{O}(\Delta t), \tag{26}$$

$$q_{t+\Delta t} - q_t = R_t q_t\, \Delta t + \mathcal{O}(\Delta t^2), \tag{27}$$

$$q_{t+\Delta t} = \big(I + R_t \Delta t\big) q_t + \mathcal{O}(\Delta t^2) \tag{28}$$

$$q_{t+\Delta t} = Q_t q_t + \mathcal{O}(\Delta t^2)\,. \tag{29}$$

□

A.2. Intuition of the Forward Transition Rates

The forward process introduced in Section 4.1 relies on a specific structure of the transition rate matrix R_t, defined as $R_t = \sigma_t R$, where $R := \mathbf{1}\mathbf{1}^T - KI \in \mathbb{R}^{K\times K}$. This form, referred to as the uniform transition rate matrix [1, 5], and can be written explicitly as

$$R_t = \sigma_t \begin{bmatrix} 1-K & 1 & \cdots & 1 \\ 1 & 1-K & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1-K \end{bmatrix}. \tag{30}$$

Intuitively, at each time t, this matrix transitions the probability vector to the uniform distribution, controlled by the scale factor σ_t.

For illustration, consider $K = 3$, $\sigma_t = 0.2$, and $q_t = [0.8, 0.1, 0.1]^T$. Then,

$$\frac{dq_t}{dt} = R_t q_t = \begin{bmatrix} -0.4 & 0.2 & 0.2 \\ 0.2 & -0.4 & 0.2 \\ 0.2 & 0.2 & -0.4 \end{bmatrix} \begin{bmatrix} 0.8 \\ 0.1 \\ 0.1 \end{bmatrix} = \begin{bmatrix} -0.28 \\ 0.14 \\ 0.14 \end{bmatrix}. \tag{31}$$

Now, increasing σ_t to 2.5 yields,

$$\frac{dq_t}{dt} = R_t q_t = \begin{bmatrix} -5 & 2.5 & 2.5 \\ 2.5 & -5 & 2.5 \\ 2.5 & 2.5 & -5 \end{bmatrix} \begin{bmatrix} 0.8 \\ 0.1 \\ 0.1 \end{bmatrix} = \begin{bmatrix} -3.5 \\ 1.75 \\ 1.75 \end{bmatrix}. \tag{32}$$

From these computations, we observe that larger values of σ_t lead to faster transitions, causing the probability mass to spread more quickly across the discrete states. The uniform structure of R_t guarantees equal transition potential among states, meaning that states with higher source probabilities will experience proportionally higher transition rates than others.

A.3. Intuition of the Reverse Transition Matrices

Recalling the reverse process introduced in Equation (3),

$$\frac{dp_{1-t}}{dt} = \overline{R}_{1-t}\, p_{1-t}, \quad \text{s.t} \quad \overline{R}_t := S_t \odot R_t - \mathrm{diag}\big(\mathbf{1}^T (S_t \odot R_t)\big)\,, \tag{33}$$

where, for simplicity, we denote $p(\mathrm{c}_t|\mathbf{y})$ as p_t. For a sufficiently small time step $\Delta t > 0$, applying the Euler approximation from Theorem 2 yields,

$$p_{t-\Delta t} \approx \overline{Q}_t p_t\,, \tag{34}$$

where $\overline{Q}_t := I + \overline{R}_t \Delta t$ represents the infinitesimal transition matrix.

Consider a simple example where $\overline{R}_t = \sigma_t R$ is a uniform transition rate matrix with parameters $K = 3$, $\sigma_t = 0.2$, $\Delta t = 0.1$, and $p_t = [0.8, 0.1, 0.1]^T$. Then,

$$p_{t-\Delta t} \approx (I + 0.1\,\overline{R}_t)\, p_t = \begin{bmatrix} 0.96 & 0.02 & 0.02 \\ 0.02 & 0.96 & 0.02 \\ 0.02 & 0.02 & 0.96 \end{bmatrix} \begin{bmatrix} 0.8 \\ 0.1 \\ 0.1 \end{bmatrix} = \begin{bmatrix} 0.774 \\ 0.113 \\ 0.113 \end{bmatrix}. \tag{35}$$

If we increase σ_t to 2.5, we obtain

$$p_{t-\Delta t} \approx (I + 0.1\,\overline{R}_t)\, p_t = \begin{bmatrix} 0.5 & 0.25 & 0.25 \\ 0.25 & 0.5 & 0.25 \\ 0.25 & 0.25 & 0.5 \end{bmatrix} \begin{bmatrix} 0.8 \\ 0.1 \\ 0.1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.25 \\ 0.25 \end{bmatrix}. \tag{36}$$

A larger σ_t therefore induces stronger state transitions, driving the distribution more rapidly toward uniformity in a single step. In general, the scaling factor σ_t governs the sharpness of the reverse diffusion: small values yield smoother, more stable refinements of the probabilities, while large values accelerate convergence but risk oversmoothing finer distinctions in p_t.

B. Ablation Study of Label Selection Strategies in DiDiCM-CP

In Section 5.1, we introduced DiDiCM-CP, our diffusion-based method over class probabilities, summarized in Algorithm 2. This method takes advantage of the tractable prior distribution $P(\mathrm{c})$ in classification settings, enabling

Method	Label Selection Method	Metric	Training Ratio - 25%			Training Ratio - 50%			Training Ratio - 100%		
			56	112	224	56	112	224	56	112	224
Ours: DiDiCM-CP $1/\Delta t = 8$	ArgMax	Top-1	56.58	63.35	69.17	65.91	72.12	77.00	69.41	76.20	79.46
		Top-5	80.99	85.89	88.61	87.48	90.75	92.85	89.89	92.88	94.72
	Sampling	Top-1	58.89	65.33	69.78	**68.34**	73.38	**77.62**	**71.86**	76.84	80.27
		Top-5	**81.98**	86.74	89.24	**88.80**	91.51	**93.79**	**90.97**	93.53	95.20
	ArgMin	Top-1	**59.05**	**68.83**	**72.27**	65.75	**73.67**	77.01	70.80	**77.89**	**80.40**
		Top-5	**81.93**	**88.76**	**91.00**	86.72	**91.63**	93.62	89.69	**93.75**	**95.29**

Table 2. Ablation study of label selection strategies in DiDiCM-CP under varying uncertainty conditions.

the computation of the complete score matrix S_t^θ within a single model iteration. This is achieved by normalizing the Concrete Score prediction, defined as $s_\theta(\mathbf{y}, j, t) \approx q(\mathrm{c}_t|\mathbf{y})/q(\mathrm{c}_t = j|\mathbf{y})$, where $j \in \{1, \ldots, K\}$ represents the noisy class label corresponding to noise level t, and $\mathbf{y}$ denotes the input to be classified.

By construction, any label j can be used in this process. However, we find empirically that the choice of j can influence model performance. In this section, we therefore examine several strategies for selecting j.

We evaluate three label selection strategies:

1. **ArgMax**: Select the class label that *maximizes* the current noisy posterior distribution, $j = \arg\max p_\theta(\mathrm{c}_t|\mathbf{y})$.
2. **Sampling**: Sample the class label from the current noisy posterior distribution, $j \sim p_\theta(\mathrm{c}_t|\mathbf{y})$.
3. **ArgMin**: Select the class label that *minimizes* the current noisy posterior distribution, $j = \arg\min p_\theta(\mathrm{c}_t|\mathbf{y})$.

Table 2 reports the performance of DiDiCM-CP under these three label selection methods. For each, we evaluate nine uncertainty conditions, varying image corruption and data scarcity levels, consistent with the experimental setup presented in Section 6.

Interestingly, across most uncertainty conditions, the ArgMin strategy yields the best performance. We hypothesize that this approach, which selects the least confident class label under the current noisy posterior $p_\theta(\mathrm{c}_t|\mathbf{y})$, appears to encourage the model to more thoroughly explore the space of possible classes and to refine its decision boundaries based on perceptual features rather than prior confidence.

The Sampling strategy performs second best, occasionally surpassing ArgMin in certain moderate-uncertainty scenarios, but showing less stability under high-uncertainty conditions, when only 25% of the training data is available.

Finally, the ArgMax strategy consistently performs worst. We attribute this to the bias it introduces: by repeatedly reinforcing the most confident class label, the model may overfit to its initial predictions and fail to adequately re-examine alternative interpretations of the input image.

C. Visualizing DiDiCM Using Misclassified Images

In this section, we visualize the top-5 class probabilities for randomly selected images on which both DiDiCM and the standard classifier produce incorrect Top-1 predictions. The corresponding illustrations are provided in Figure 6.

Figure 6 presents 10 randomly sampled images in which both the classification approaches bring incorrect prediction. For each image, we display the sorted top-5 class probabilities obtained from DiDiCM (based on DiDiRN-50) and from a standard classifier (based on ResNet-50). Both models were trained using our Strong Aug training recipe, detailed in Appendix F, which represents the state-of-the-art training procedure for ResNet-50 [46].

We observe that in most cases, the standard classifier exhibits overconfident Top-1 probability estimates for the incorrect class labels. In contrast, DiDiCM produces a more balanced probability distribution across plausible class options. For instance, in image #8, depicting a spider web with a car mirror in the background, the true label is "spider web". While the standard classifier assigns nearly 100% confidence to the "car mirror" label, DiDiCM appropriately allocates substantial probability mass to the "spider web" class as well, reflecting a more calibrated representation of the posterior distribution.

D. Empirical Study of DiDiRN Model Size

In this section, we provide additional experimental results evaluating the performance of DiDiCM compared to standard classification models across various model sizes, leveraging the DiDiRN architectural design based on ResNet models.

The proposed DiDiRN architecture is built on top of the ResNet framework to enable direct comparison with corresponding baseline models, as described in Section 6.1 and illustrated in Figure 3.

Table 3 summarizes the full results, while Figure 5 visualizes the corresponding accuracy gains. We focus on the most challenging uncertainty scenario from our empirical study in Section 6, in which the input image resolution is

Method	Model	Params (M)	Top-1 (%)	Top-5 (%)
Standard Classifiers	ResNet-18	11.7	43.46	67.70
	ResNet-34	21.8	47.41	71.65
	ResNet-50	25.6	53.77	76.03
	ResNet-101	44.5	54.33	76.35
	ResNet-152	60.2	54.20	76.00
Ours: DiDiCM-CP $1/\Delta t = 8$	DiDiRN-18	12.1	**49.13**	**74.01**
	DiDiRN-34	22.6	**52.50**	**77.09**
	DiDiRN-50	27.7	**59.05**	**81.93**
	DiDiRN-101	49.0	**61.86**	**83.95**
	DiDiRN-152	66.6	**62.95**	**84.61**

Table 3. **ImageNet Top-1 and Top-5 accuracy across model sizes.** Comparison of DiDiCM-CP (8 steps) and standard classifiers at input resolution 56, trained on 25% of the available data. DiDiCM consistently outperforms standard classifiers across all model sizes.

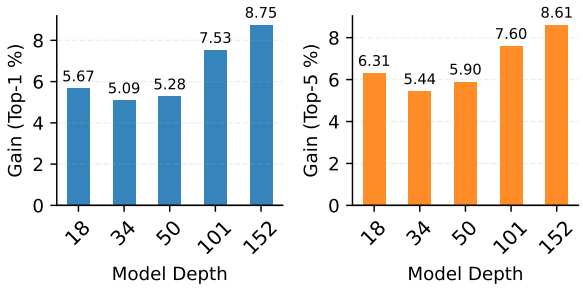

Figure 5. **ImageNet Top-1 and Top-5 accuracy gains across model sizes**. Comparison of DiDiCM-CP (8-step) and standard classifiers at 56 input resolution, trained on 25% of the data.

56 and only 25% of the data are available for training. Under this setting, we train DiDiRN models corresponding to standard ResNet sizes, i.e., DiDiRN-18 through DiDiRN-152. We then evaluate and compare the Top-1 and Top-5 accuracies of the DiDiRN variants with their baseline ResNet counterparts.

As shown in Table 3, DiDiRN consistently outperforms standard classifiers across all model sizes. Figure 5 further indicates that accuracy gains in Top-1 and Top-5 metrics are approximately consistent for each model scale. The largest observed improvement occurs with DiDiRN-152, which achieves an accuracy gain of ∼8.7% over ResNet-152 in both Top-1 and Top-5 metrics. This finding suggests that scaling up DiDiRN architectures has the potential to yield higher performance than conventional classification models.

Hyperparameter	Weak Aug	Strong Aug
Optimization		
Epochs	600	600
Warmup epochs	5	5
Batch size	1024	2048
Optimizer	LAMB	LAMB
Learning rate	2×10^{-3}	3.5×10^{-3}
Weight decay	0.1	0.1
Mixed precision	v	v
Image Augmentation		
Repeated augmentation	-	3
Horizontal flip	v	v
Random resized crop	v	v
RandAugment	-	7 / 0.5
Color jitter	0.4	-
Mixup α	-	0.2
CutMix α	-	1.0
Label smoothing ϵ	-	0.1
Evaluation		
Test crop ratio	0.875	0.875

Table 4. Details of the two training recipes used for the DiDiCM experiments. Both are variants of the A1 recipe from ResNet-SB [46]. **Weak Aug** incorporates standard image augmentations as implemented in PyTorch [37], while **Strong Aug** follows the original A1 procedure without modifications.

E. Empirical Study on ImageNet-C

We evaluate the robustness of the benefits from our DiDiCM-CP on the ImageNet-C benchmark [25], which comprises a diverse set of algorithmically generated corruptions spanning noise, blur, weather, and digital distortions. We report Top-1 and Top-5 classification accuracy across all corruption types.

All models are trained on 25% of the ImageNet training set at a resolution of 56×56, using the Strong Aug training policy (see Appendix F). For each corruption type, the training data is corrupted accordingly at severity level 5 (See Hendrycks and Dietterich [25] for more details), and we compare against a ResNet-50 baseline trained under the same protocol. Importantly, we do not perform corruption-specific hyperparameter tuning, although such optimization could further improve performance.

As shown in Table 5, DiDiCM-CP consistently outperforms standard classifiers across all corruption types, demonstrating robustness to common corruptions. On clean (uncorrupted) data, our method achieves a Top-1 accuracy of 59.05%, which serves as an approximate upper bound for performance under corrupted settings.

Notably, we observe Top-1 accuracy gains of up to +13.61% (e.g., Impulse Noise), and even more pronounced

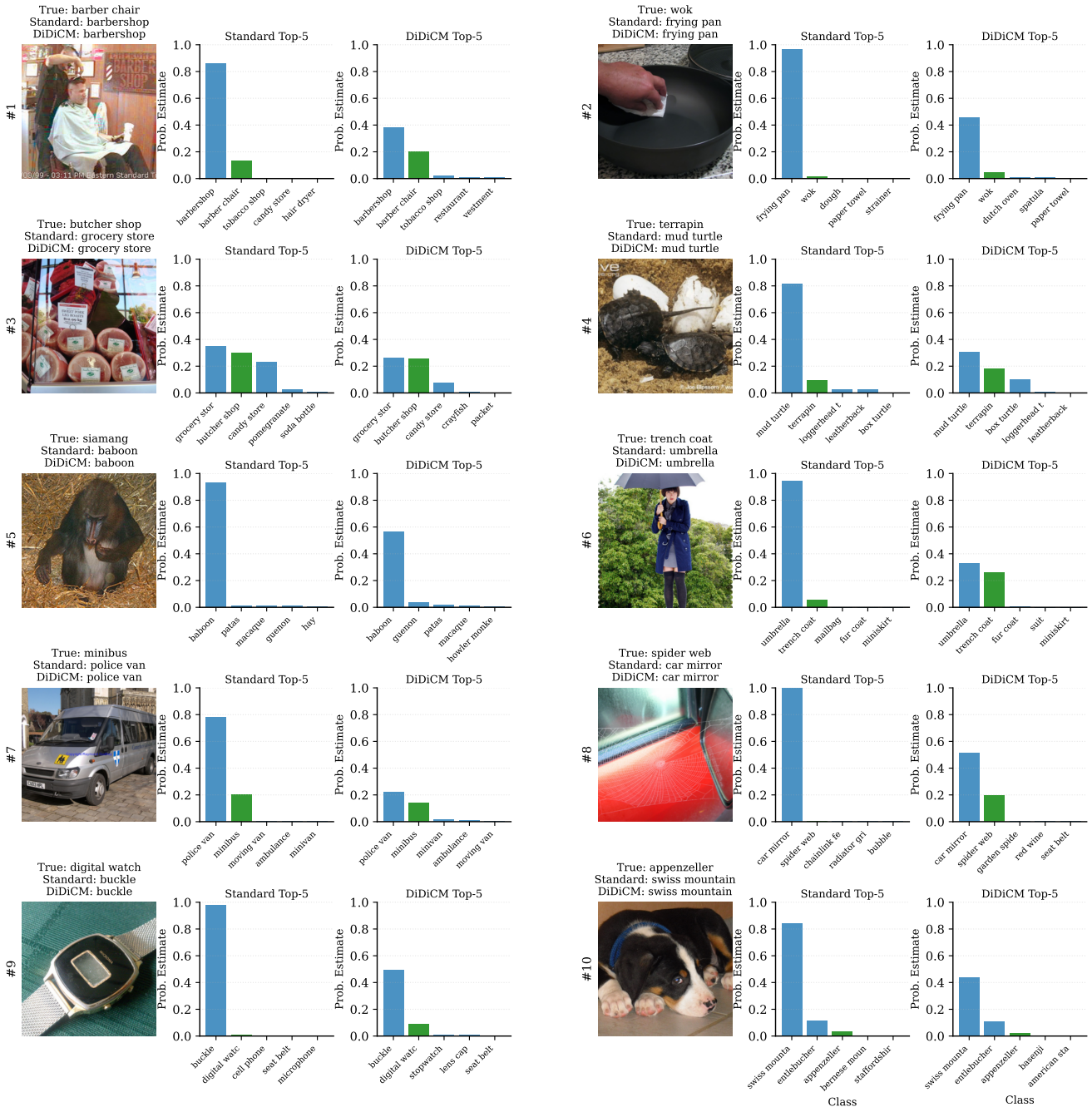

Figure 6. Random images of misclassifications by both DiDiCM (DiDiRN-50) and the standard classifier (ResNet-50), trained with the Strong Aug recipe. The results indicate that while the standard classifier tends toward overconfidence, DiDiCM yields more balanced prediction probabilities.

improvements in Top-5 accuracy, with improvements of up to +17.05% (e.g., Impulse Noise). Across all corruption types, Top-5 accuracy improves consistently, indicating that DiDiCM-CP not only increases prediction confidence but also preserves correct labels among the top predictions under severe degradation.

F. Technical Training Details

In this section we provide additional technical details of all the experiments in this paper. In Table 4, we summarize the training details relevant only to our DiDiCM experiments.

Architecture. Most of our experiments are conducted

Method	Metric	Corruption Types									
		No Corruption	Brightness	Fog	Glass Blur	Pixelate	Spatter	Contrast	Frost	Impulse Noise	Saturate
Standard Classifiers	Top-1	53.77	47.95	41.25	21.89	36.12	43.23	45.12	42.78	30.98	45.39
	Top-5	76.03	70.54	63.90	40.13	58.61	66.21	68.36	66.03	52.62	68.56
Ours: DiDiCM-CP $1/\Delta t = 8$	Top-1	**59.05**	**55.11**	**43.95**	**24.36**	**40.23**	**53.89**	**49.98**	**49.09**	**44.59**	**54.23**
	Top-5	**81.93**	**78.88**	**69.27**	**45.35**	**64.76**	**78.04**	**75.30**	**73.84**	**69.67**	**78.34**
		Speckle Noise	Defocus Blur	Gaussian Blur	JPEG Compression	Shot Noise	Zoom Blur	Elastic Transform	Gaussian Noise	Motion Blur	Snow
Standard Classifiers	Top-1	30.71	25.57	22.82	28.87	23.78	46.65	32.39	22.81	33.95	39.07
	Top-5	51.82	45.68	41.22	49.99	43.24	69.32	53.65	41.81	56.04	61.76
Ours: DiDiCM-CP $1/\Delta t = 8$	Top-1	**37.46**	**29.48**	**26.34**	**31.08**	**26.71**	**51.26**	**37.04**	**25.44**	**38.10**	**46.52**
	Top-5	**61.42**	**52.90**	**48.18**	**54.48**	**49.04**	**75.76**	**60.86**	**47.20**	**62.70**	**71.17**

Table 5. **ImageNet Top-1 and Top-5 Accuracy:** DiDiCM (DiDiRN-50) vs. standard classifiers (ResNet-50) on ImageNet-C [25] corruption benchmarks. For each corruption type, the model is trained on 25% of the training data with corruption severity level 5 at 56×56 resolution using the Strong Aug training policy (see Appendix F); best Top-1 and Top-5 results are bolded. DiDiCM consistently outperforms standard classifiers across all corruption types.

using ResNet-50. For DiDiCM, we construct the corresponding DiDiRN-50 model. In Appendix D, we extend our empirical study with additional experiments across all standard ResNet model sizes.

Dataset. All experiments were conducted on the challenging ImageNet-1k (ILSVRC-2012) dataset [40]. Image normalization was applied consistently during both training and evaluation. For training, we applied the image augmentations described below. For evaluation, we used the standard center crop ratio of 0.875.

Image Augmentations. We evaluate model performance under two augmentation regimes. The first, referred to as *Weak Aug*, consists of the standard image augmentations used for the ImageNet dataset as implemented in PyTorch [37]. These include random horizontal flipping, random resized cropping, and ColorJitter. The second, denoted by *Strong Aug*, employs a more advanced, state-of-the-art augmentation pipeline for ResNet models, following ResNet-SB [46]. It includes RandAugment [9] with repeated augmentation enabled, Mixup [52] and CutMix [51] with label smoothing, and the standard random horizontal flipping and random resized cropping.

Optimization. All models were trained using the LAMB optimizer [49] with a cosine annealing learning rate schedule and a weight decay of 0.1. Training was conducted for 600 epochs, with the first 5 epochs designated for linear warmup. For experiments employing Weak Aug, we used a batch size of 1024 and a base learning rate of 2×10^{-3}. The learning rate was scaled according to the training data ratio using the square-root scaling rule, $\text{lr} = \text{base} \times \sqrt{1/\text{ratio}}$, to mitigate overfitting. For standard classification experiments with Strong Aug, we followed ResNet-SB [46] by using a batch size of 2048 and a base learning rate of 5×10^{-3}. The learning rate was scaled according to the same policy described above. For DiDiCM experiments with Strong Aug, we also used a batch size of 2048 but with a base learning rate of 3.5×10^{-3}. For these experiments, learning rate scaling based on the data ratio was omitted, as we found it to be non-beneficial for performance in contrast for standard classifiers performance.

G. The Quality-Efficiency Tradeoff

In this section, we evaluate the top-1 and top-5 accuracy of DiDiCM under efficient configuration settings, considering both DiDiCM-CP and DiDiCM-CL. Specifically, we set the diffusion steps to $\frac{1}{\Delta t} = 2$ and, for DiDiCM-CL, the number of class labels to be averaged to $N = 16$. These settings correspond to 2 NFEs for DiDiCM-CP and 32 NFEs for DiDiCM-CL. Notably, DiDiCM-CL is more memory-efficient than DiDiCM-CP, providing a viable alternative in resource-constrained scenarios. Full results are reported in Table 6.

Consistent with our observations using 8 diffusion steps

Method	NFEs	Memory	Metric	Training Ratio - 25%			Training Ratio - 50%			Training Ratio - 100%		
				56	112	224	56	112	224	56	112	224
Standard Classifiers	1	-	Top-1	53.77	66.51	71.85	60.51	72.31	**77.05**	64.71	75.96	**80.42**
			Top-5	76.03	85.41	88.72	81.74	89.81	92.64	85.32	92.30	94.60
DiDiCM-CL	$N\frac{1}{\Delta t}$	$\mathcal{O}(K+N)$	Top-1	56.96	67.15	70.78	64.51	72.58	76.20	68.74	76.86	79.77
			Top-5	72.27	81.61	83.83	79.57	85.60	88.58	83.17	88.89	91.51
DiDiCM-CP	$\frac{1}{\Delta t}$	$\mathcal{O}(K^2)$	Top-1	**58.95**	**68.71**	**72.10**	**65.57**	**73.57**	76.78	**69.84**	**77.63**	80.15
			Top-5	**81.80**	**88.64**	**90.89**	**86.60**	**91.57**	**93.56**	**89.21**	**93.81**	**95.23**

Table 6. **Quality-Efficiency Analysis**. ImageNet top-1 and top-5 accuracy of DiDiCM under efficient configurations ($1/\Delta t = 2$, $N = 16$ for DiDiCM-CL). Results compare DiDiCM-CP (2 NFEs) and DiDiCM-CL (32 NFEs) against standard classification (1 NFE).

in Table 1, DiDiCM-CP consistently outperforms standard classifiers in top-5 accuracy, with performance gains increasing under higher uncertainty. For top-1 accuracy, DiDiCM also surpasses standard classifiers in settings that challenge uncertainty estimation. In low-uncertainty conditions (resolution 224, full and half training data), standard classifiers slightly outperform DiDiCM configured under this efficient setting.

While DiDiCM-CL is more memory-efficient than DiDiCM-CP, we find that its efficient configuration with 32 NFEs achieves performance comparable to DiDiCM-CP, with an accuracy difference of $\sim$1%. This suggests that DiDiCM-CL can serve as a practical, memory-conscious alternative without substantial sacrifice in accuracy.