Contrastive MIM: A Contrastive Mutual Information Framework for Unified Generative and Discriminative Representation Learning

Micha Livne
NVIDIA
mlivne@nvidia.com

Abstract

Learning representations that generalize well to unknown downstream tasks is a central challenge in representation learning. Existing approaches such as contrastive learning, self-supervised masking, and denoising auto-encoders address this challenge with varying trade-offs. In this paper, we introduce the *contrastive Mutual Information Machine* (cMIM), a probabilistic framework that augments the Mutual Information Machine (MIM) with a novel contrastive objective. While MIM maximizes mutual information between inputs and latent variables and encourages clustering of latent codes, its representations underperform on discriminative tasks compared to state-of-the-art alternatives. cMIM addresses this limitation by enforcing global discriminative structure while retaining MIM's generative strengths.

We present two main contributions: (1) we propose cMIM, a contrastive extension of MIM that eliminates the need for positive data augmentation and is robust to batch size, unlike InfoNCE-based methods; (2) we introduce *informative embeddings*, a general technique for extracting enriched representations from encoder–decoder models that substantially improve discriminative performance without additional training, and which apply broadly beyond MIM.

Empirical results demonstrate that cMIM consistently outperforms MIM and InfoNCE in classification and regression tasks, while preserving comparable reconstruction quality. These findings suggest that cMIM provides a unified framework for learning representations that are simultaneously effective for discriminative and generative applications.

1 Introduction

Learning representations that remain effective across unknown downstream tasks is a central challenge in representation learning. Prominent approaches addressing this challenge include contrastive learning (*e.g.*, Chen et al. [2020], van den Oord et al. [2018]), self-supervised masking (*e.g.*, Devlin et al. [2018]), and denoising auto-encoders (*e.g.*, Bengio et al. [2013]).

In this work, we propose a new method, *cMIM* (Contrastive MIM), designed to learn representations that are broadly useful for downstream applications. cMIM integrates a contrastive learning objective with the Mutual Information Machine (MIM) framework introduced by Livne et al. [2019]. MIM is a probabilistic auto-encoder that maximizes mutual information between inputs and latent representations while clustering the latent codes. However, preliminary results suggest that MIM alone yields representations less effective for discriminative tasks compared to state-of-the-art alternatives. Our cMIM framework directly addresses this limitation by incorporating contrastive learning.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: accepted at NEGEL workshop @ NeurIPS 2025.

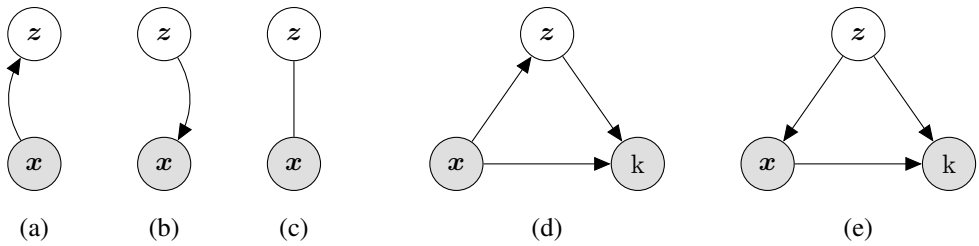


Figure 1: (Left) A MIM model learns two factorizations of a joint distribution (encoding/decoding) and an undirected joint. (Right) cMIM extends MIM with a binary variable k to encourage global discriminative structure while preserving local clustering.

Algorithm 1 Learning parameters θ of cMIM

Require: Samples from dataset $p(\boldsymbol{x})$
1: **while** not converged **do**
2: $\mathcal{D} \leftarrow \{\boldsymbol{x}_j, \boldsymbol{z}_j \sim q_\theta(\boldsymbol{z} \mid \boldsymbol{x})\, p(\boldsymbol{x})\}_{j=1}^{B}$ {Sample a batch}
3: $\hat{\mathcal{L}}_{\text{A-MIM}} = -\frac{1}{B}\sum_{i=1}^{B}\big(\log p_\theta(\boldsymbol{x}_i \mid \boldsymbol{z}_i) + \log p_{k=1}(\boldsymbol{x}_i, \boldsymbol{z}_i) + \frac{1}{2}(\log q_\theta(\boldsymbol{z}_i \mid \boldsymbol{x}_i) + \log p(\boldsymbol{z}_i))\big)$
4: $\theta \leftarrow \theta - \eta\, \nabla_\theta \hat{\mathcal{L}}_{\text{A-MIM}}$ {Reparameterized gradients}
5: **end while**

Figure 2: Training algorithm for cMIM.

Our main contributions are as follows:

1. We propose a contrastive extension to the Mutual Information Machine (MIM) that enables learning discriminative representations without requiring data augmentation (no explicit "positive" pairs) and with reduced sensitivity to the number of negative samples (typically determined by batch size).
2. We introduce *informative embeddings*, a generic method for extracting embeddings from encoder–decoder models. This technique improves discriminative downstream performance without additional training and applies broadly to pre-trained encoder–decoder architectures.

By combining generative modeling with contrastive objectives, cMIM provides a unified framework effective for both generative and discriminative tasks. Empirical results show that cMIM produces representations that retain MIM's generative capacity while significantly improving performance in downstream discriminative settings.

2 Formulation

Below we provide a summary of the cMIM formulation and training procedure. For full details and derivations see Section A.

Goal. We extend the Mutual Information Machine (MIM), a probabilistic auto-encoder that learns informative, clustered latent codes $\boldsymbol{z}$ (see Fig. 1a). MIM 's loss induces local similarity (nearby codes for similar samples), yet its global latent layout may be suboptimal for discriminative tasks. We therefore augment MIM with a contrastive term (see Fig. 1b) that pushes *dissimilar* samples apart in angle while MIM preserves *similarity* in Euclidean distance. This extended formulation remains a valid probabilistic model in its own right. This extended formulation remains a valid probabilistic model in its own right.

Notation. Observations are $\boldsymbol{x} \in \mathcal{X}$, latent codes $\boldsymbol{z} \in \mathcal{Z}$. Enc/dec distributions are $q_\theta(\boldsymbol{z} \mid \boldsymbol{x})$ and $p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$. We denote cosine similarity by $\mathrm{sim}(\mathbf{a}, \mathbf{b})$ and its temperature-scaled exponential by $\exp(\mathrm{sim}/\tau)$.

2.1 MIM and Contrastive Objective

The original MIM objective can be written as a mixture-model loss that maximizes mutual information between inputs and latent codes while promoting clustering. For A-MIM we sample $(\boldsymbol{x}, \boldsymbol{z})$ along the encoding path and evaluate cross-entropies under both factorizations. In practice [Livne et al., 2020] the empirical loss is approximated as

$$\hat{\mathcal{L}}_{\text{A-MIM}} = -\tfrac{1}{N} \sum_{i=1}^{N} \Big(\log p_\theta(\boldsymbol{x}_i \mid \boldsymbol{z}_i) + \tfrac{1}{2}(\log q_\theta(\boldsymbol{z}_i \mid \boldsymbol{x}_i) + \log p(\boldsymbol{z}_i)) \Big). \tag{1}$$

To introduce global discriminative structure, we add the contrastive probability term $p_{k=1}(\boldsymbol{x}_i, \boldsymbol{z}_i)$, defined as

$$p_{k=1}(\boldsymbol{x}_i, \boldsymbol{z}_i) = \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i)/\tau)}{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i)/\tau) + \frac{1}{B-1} \sum_{j \neq i} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}. \tag{2}$$

The resulting empirical cMIM loss is

$$\hat{\mathcal{L}}_{\text{cMIM}} = \hat{\mathcal{L}}_{\text{A-MIM}} - \tfrac{1}{N} \sum_{i=1}^{N} \log p_{k=1}(\boldsymbol{x}_i, \boldsymbol{z}_i). \tag{3}$$

This term separates dissimilar codes while MIM enforces clustering of similar codes, producing a latent space that is both generative and discriminative.

Training uses in-batch negatives; no positive data augmentation is required. Intuitively, MIM clusters nearby samples, while the contrastive term spreads clusters on the unit hypersphere, improving separability for downstream tasks. Moreover, because cMIM uses an expectation over negatives (approximated in-batch) rather than a B-way classification, its calibration is less sensitive to batch size. We provide pseudocode in Algorithm 2.

2.2 Relation to InfoNCE

The contrastive probability $p_{k=1}$ can be interpreted as a shifted version of the InfoNCE softmax. Specifically, the positive logit is offset by $\log(B-1)$, calibrating the baseline probability to $1/2$ instead of $1/B$ when all logits are equal. Unlike standard InfoNCE, cMIM does not require positive augmentations, since MIM already clusters codes locally. This difference makes cMIM more robust to batch size and easier to apply to modalities where augmentations are difficult to design (e.g., text).

Empirically, we find that this formulation maintains generative fidelity while improving classification and regression accuracy, highlighting the benefit of combining MIM's clustering with angular separation from the contrastive term.

2.3 Informative Embeddings

We extract "informative embeddings" $\boldsymbol{h}$ from decoder hidden states just before projecting to the parameters of $p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$. For sequences we mean-pool over time; for non-autoregressive decoders we use the final hidden representation. These embeddings enrich the latent code with decoder context, yielding stronger performance on discriminative tasks while preserving generative fidelity. See Fig. 3 for an illustration.

3 Experiments

To evaluate the proposed cMIM model, we conduct experiments on a 2D toy example, MNIST-like images, and on molecular property prediction tasks (MolMIM by Reidenbach et al. [2023]). The 2D toy example illustrates the impact of the proposed contrastive MIM loss. We then explore the qualitative nature of cMIM by running rigorous experiments on MNIST-like datasets. Finally, we compare the performance of cMIM with MIM, VAE, and AutoEncoder models trained on molecular data, assessing their reconstruction and effectiveness in downstream tasks.

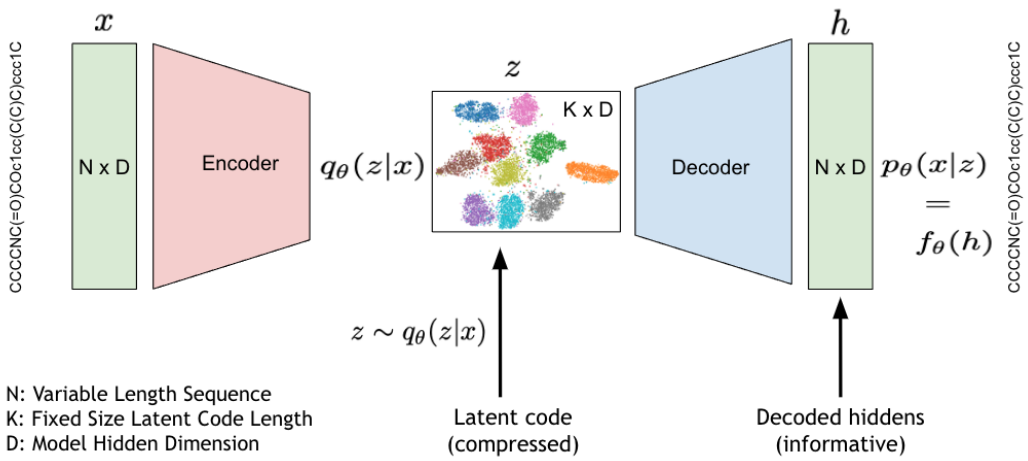


Figure 3: Informative embeddings $\boldsymbol{h}$ are taken from decoder hidden states before mapping to $p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$. For auto-regressive decoders we use teacher forcing.

#	Dataset	Train Samples	Test Samples	Categories	Description
1	MNIST	60,000	10,000	10	Handwritten digits
2	Fashion MNIST	60,000	10,000	10	Clothing images
3	EMNIST Letters	88,800	14,800	27	Handwritten letters
4	EMNIST Digits	240,000	40,000	10	Handwritten digits
5	PathMNIST	89,996	7,180	9	Colon tissue histology
6	DermaMNIST	7,007	2,003	7	Skin lesion images
7	OCTMNIST	97,477	8,646	4	Retinal OCT images
8	PneumoniaMNIST	9,728	2,433	2	Pneumonia chest X-rays
9	RetinaMNIST	1,600	400	5	Retinal fundus images
10	BreastMNIST	7,000	2,000	2	Breast tumor ultrasound
11	BloodMNIST	11,959	3,432	8	Blood cell microscopy
12	TissueMNIST	165,466	47,711	8	Kidney tissue cells
13	OrganAMNIST	34,581	8,336	11	Abdominal organ CT scans
14	OrganCMNIST	13,000	3,239	11	Organ CT, central slices
15	OrganSMNIST	23,000	5,749	11	Organ CT, sagittal slices

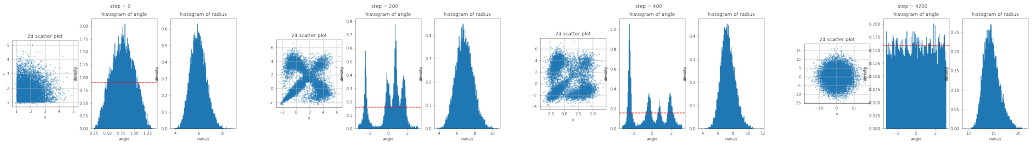
Table 1: **Image Classification:** Summary of train/test samples, categories, and descriptions for MNIST, EMNIST, and MedMNIST datasets.

3.1 Experiment Details and Datasets

All models are trained in an unsupervised manner. The checkpoint with the lowest validation loss is selected for evaluation. We avoid selecting any intermediate checkpoints, a common heuristic which does not scale well with complexity and model size. For downstream tasks, we freeze the encoder–decoder and train lightweight classifiers on the learned representations, evaluating them on the held-out test set. Importantly, classification accuracy is not monitored during training. This ensures that comparisons reflect the quality of unsupervised representations rather than reliance on checkpoint selection heuristics. Training continues until convergence, making comparisons fair across models.

2D Toy Example. A synthetic dataset of 1000 points in 2D space is initialized in the first quadrant. The task is to check the effect of the proposed contrastive MIM loss on their position, highlighting the effect of Eq. (6).

Image Classification on MNIST-like Datasets. We train MIM, cMIM, VAE, cVAE (VAE with the contrastive term), and InfoNCE to convergence on MNIST-like datasets. Comparisons include classification accuracy, batch size sensitivity, and reconstruction (except InfoNCE). Datasets include



(a) Step 0 **(b)** Step 200 **(c)** Step 400 **(d)** Step 4200

Figure 4: Effect of the contrastive MIM loss term in Eq. (6) on the 2D toy example. Each plot shows: latent space (left), histogram of latent code angles (middle), and histogram of latent radii (right). From **(a)** initialization in the first quadrant to **(d)** after 4200 training steps, the loss distributes points uniformly in angle while allowing radii to vary. This complements MIM's clustering by encouraging angular uniformity, which enhances separability and thus improves downstream discriminative performance.

MNIST Deng [2012], EMNIST (letters, digits) Cohen et al. [2017], and MedMNIST Yang et al. [2021]. All images are resized to 28×28 pixels and converted to black and white when needed. We used $\tau = 0.1$ (as in InfoNCE) following a small hyper-parameter search with $\tau \in \{0.1, 1\}$. The encoder is a Perceiver Jaegle et al. [2021] with 1 cross-attention layer, 4 self-attention layers, hidden size 16, projecting 784 pixels to 400 steps, followed by a projection to 64-dimensional latent codes. The decoder mirrors this design. Models are trained for 500k steps with batch sizes 2, 5, 10, 100, 200, using Adam with learning rate 10^{-3}, and WSD scheduler Hu et al. [2024]. Classifiers include KNN ($k = 5$; cosine and Euclidean metrics) and a one-hidden-layer MLP (size 400, Adam, 10^{-3}, 1000 steps).

Molecular Property Prediction. We use ZINC-15 Sterling and Irwin [2015] with SMILES Weininger [1988] sequences, following Reidenbach et al. [2023]. Properties include ESOL, FreeSolv, and Lipophilicity. Here $\tau = 1$. Both MIM and cMIM are trained for 250k steps. Embeddings are evaluated using SVM and MLP regressors, both with and without informative embeddings, and compared against CDDD Winter et al. [2019]. Architectural details appear in Appendix B.

3.2 Effects of cMIM Loss on 2D Toy Example

We study the effect of the proposed contrastive MIM loss term (Eq. (6)) on a 2D toy example. Training minimizes the negative log-likelihood associated with Eq. (6), learning optimal latent codes in two dimensions. Here we use $\tau = 1$.

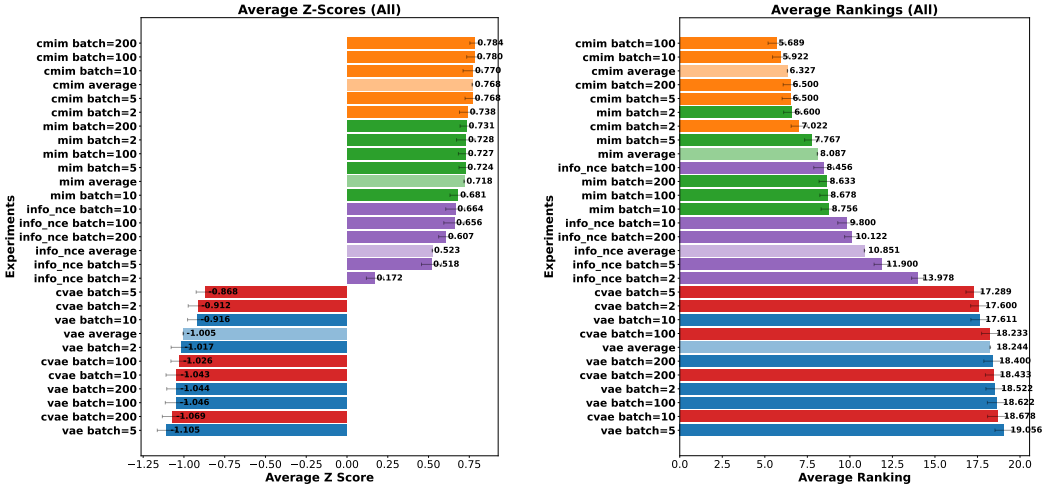
We expect the latent codes to distribute uniformly across all angles while maintaining variability in radii, as suggested by Wang and Isola [2020]. Fig. 4 shows the progression of the latent space during training. The results confirm that the contrastive term integrates smoothly with the MIM objective, preserving radial clustering while enforcing uniform angular distribution.

3.3 Classification Accuracy

We now analyze classification accuracy, as a proxy for the quality of the learned embeddings, and which was never used as a training signal.

Image Classification. We trained MIM, cMIM, VAE, cVAE (VAE + cMIM contrastive loss term), and InfoNCE across batch sizes $\{2, 5, 10, 100, 200\}$. All models share the same architecture, with InfoNCE consisting only of the encoder. Checkpoints with the lowest validation loss were evaluated on test sets. This design controls for architecture, optimizer, training steps, and dataset usage, isolating the effect of the objective.

We report results using KNN (cosine and Euclidean) and a one-hidden-layer MLP with 400 dimensions. We use Scikit-learn Pedregosa et al. [2011] with default values. Inputs to classifiers are either the mean encoding (standard embedding) or informative embeddings (Section 2.3). Together we evaluate 6 classification tasks per model and batch size. Performance is summarized with (1) average normalized accuracy (z-score across datasets and tasks) and (2) average ranking per dataset and task (Fig. 5).



(a) Z-scores with error bars **(b)** Rankings with error bars

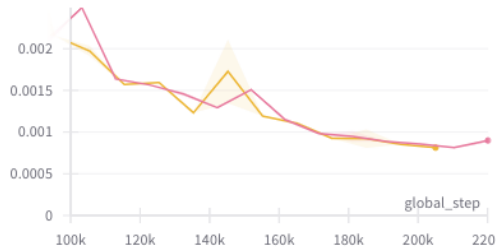
Figure 5: Classification accuracy across datasets and classifiers. Colors indicate model families: cMIM (orange), MIM (green), InfoNCE (purple), VAE (blue), cVAE (red). Light shades denote model averages. cMIM consistently outperforms all baselines across batch sizes and metrics.

Model (Latent K × D)	**ESOL**		**FreeSolv**		**Lipophilicity**		**Recon.**
	SVM	**MLP**	**SVM**	**MLP**	**SVM**	**MLP**	
MIM (1 × 512)	0.65	0.34	2.23	1.82	0.663	0.61	100%
cMIM (1 × 512)	0.47	0.19	2.32	1.67	0.546	0.38	100%
MIM (1 × 512) info emb	0.21	0.29	1.55	1.4	0.234	0.28	100%
cMIM (1 × 512) info emb	0.21	0.24	1.74	1.35	0.24	0.23	100%
CDDD (512)	**0.33**		**0.94**		**0.4**		
†Seq2seq (N × 512)	0.37	0.43	1.24	1.4	0.46	0.61	100%
†Perceiver (4 × 512)	0.4	0.36	1.22	1.05	0.48	0.47	100%
†VAE (4 × 512)	0.55	0.49	1.65	3.3	0.63	0.55	46%
MIM (1 × 512)	0.58	0.54	1.95	1.9	0.66	0.62	100%
Morgan fingerprints (512)	1.52	1.26	5.09	3.94	0.63	0.61	

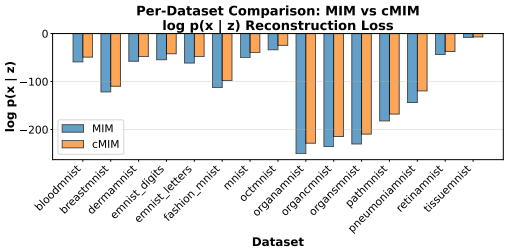
Table 2: Comparison of models on ESOL, FreeSolv, and Lipophilicity using SVM and MLP regressors, with reconstruction accuracy. Top: our results. Bottom: results from Reidenbach et al. [2023]. For †models, sequence representations were averaged to 512 dimensions. Bold: best non-MIM results. Highlighted: best among MIM-based models. Note that CDDD training included these classification tasks.

cMIM consistently outperformed all baselines across batch sizes and metrics, dominating the ranking plots. Both MIM and cMIM surpassed VAE, cVAE, and most InfoNCE runs in z-score. The only competitive non-MIM model was InfoNCE with batch size 100. Adding a contrastive term to VAE (cVAE) had little impact, indicating that cMIM's advantage arises from combining MIM's clustering with angular separation.

Molecular Property Prediction and Informative Embeddings. Table 2 compares MIM and cMIM on ESOL, FreeSolv, and Lipophilicity tasks. We evaluate SVM and MLP regressors trained on embeddings and informative embeddings. Baselines include CDDD Winter et al. [2019], Seq2seq,



(a) ZINC15 Training Validation Loss



(b) MNIST Dataset Comparison

Figure 7: Reconstruction performance of MIM and cMIM. **(a)** Validation reconstruction loss during training on molecular data (cMIM in yellow, MIM in pink) shows comparable behavior. **(b)** Per-dataset test reconstruction log-likelihood on MNIST-like datasets. Surprisingly, cMIM achieves better average reconstruction (-96.25) compared to MIM (-109.64), a +12.2% relative improvement, suggesting a beneficial regularization effect.

Perceiver, VAE, and Morgan fingerprints. We note that CDDD was trained with the classification tasks during training.

cMIM with informative embeddings outperformed vanilla MIM and was competitive with or superior to these baselines. This highlights the value of informative embeddings and the discriminative structure encouraged by cMIM.

3.4 Batch Size Sensitivity

Fig. 6 summarizes batch size sensitivity for MIM, cMIM, VAE, cVAE, and InfoNCE. For each model, we performed a linear regression of average z-score on batch size, across six evaluation settings (three classifiers $\times$ two embedding types). The slope of this fit serves as a measure of sensitivity: positive slope means higher accuracy with larger batches, while near-zero slope indicates robustness.

InfoNCE shows clear dependence on batch size, with positive slopes and low variance. MIM and cMIM both yield slopes near zero with small variance, confirming their robustness to batch size. By contrast, VAE and cVAE exhibit high variance in slopes, reflecting unstable performance and strong sensitivity to batch size changes.

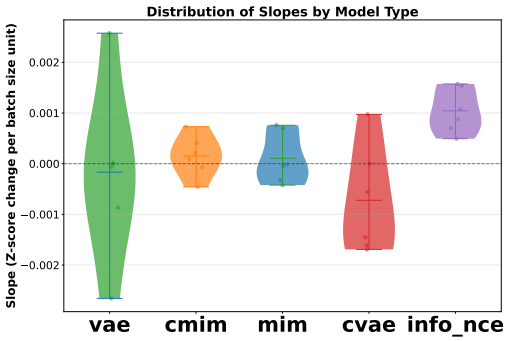


Figure 6: Distribution of slopes from linear fits of accuracy vs. batch size for different models. Each point corresponds to average z-score of a model trained on MNIST-like datasets.

3.5 Reconstruction

Fig. 7 compares reconstruction quality of MIM and cMIM. On molecular data (panel **a**), both models exhibit nearly identical reconstruction loss trajectories during training. On MNIST-like datasets (panel **b**), test log-likelihood reveals a consistent advantage for cMIM, which outperforms MIM by an average margin of 12.2%. This improvement was unexpected, as both methods share the same generative architecture. We conjecture that the gain to the implicit regularization can be attributed to the contrastive term, which could reduce overfitting while maintaining generative fidelity.

4 Related Work

Contrastive Learning. Contrastive learning has become a cornerstone of self-supervised representation learning, with methods such as CPC van den Oord et al. [2018], SimCLR Chen et al. [2020], and

MoCo He et al. [2020] demonstrating strong discriminative performance. These approaches typically rely on data augmentation to form positive pairs, making their success dependent on carefully chosen invariances. Augmentation-free contrastive methods, such as BYOL Grill et al. [2020] and SimSiam Chen and He [2021], avoid negatives but often require additional predictors or asymmetries for stability. Our work differs by integrating contrastive learning directly into a probabilistic framework, eliminating the need for augmentation or auxiliary networks.

Mutual Information Maximization. The Mutual Information Machine (MIM) Livne et al. [2019] and follow-up works Reidenbach et al. [2023] maximize mutual information between inputs and latent codes while encouraging latent clustering. Related approaches such as Deep InfoMax Hjelm et al. [2018] and InfoVAE Zhao et al. [2017] also maximize information-theoretic quantities, but typically lack a generative auto-encoding structure, or require various approximations and weighted losses which are hard to tune. Our method extends MIM with a contrastive component, addressing its limited discriminative power.

Informative Embeddings. Extracting hidden states from encoder–decoder models has proven effective in large language models Brown et al. [2020], Lee et al. [2024]. Similarly, representations from intermediate layers of auto-encoders or VAEs have been used for downstream prediction tasks Alemi et al. [2018]. We generalize this idea by introducing *informative embeddings*, a systematic method to leverage decoder hidden states in probabilistic auto-encoders, demonstrating significant gains in both image and molecular tasks.

Unifying Generative and Discriminative Learning. Bridging generative modeling with discriminative performance has been a longstanding goal, explored in frameworks such as β-VAE Higgins et al. [2017], InfoGAN Chen et al. [2016], and hybrid likelihood–contrastive models van den Oord et al. [2018]. Our work contributes to this line by showing that cMIM yields a single framework that maintains generative fidelity while significantly improving discriminative utility.

5 Limitations

While cMIM improves discriminative performance and robustness to batch size, several limitations remain. Our evaluation of generative capacity is limited to reconstruction, leaving open questions on sample quality, diversity, and controlled generation. Experiments are restricted to moderate-scale models and datasets, so scalability to larger architectures and high-dimensional modalities (e.g., video or long-context language) is unclear. Although cMIM avoids data augmentation, results may still depend on the similarity function and temperature τ. Finally, despite reduced batch-size sensitivity, performance can benefit from larger effective numbers of negatives, which may incur computational cost. Future work should explore scaling, broader modalities, and deeper analysis of generative behavior.

6 Conclusions

In this paper, we introduced cMIM, a contrastive extension of the MIM framework. Unlike conventional contrastive learning, cMIM does not require positive data augmentation and is less sensitive to batch size than InfoNCE. Our experiments demonstrate that cMIM learns more discriminative features than both MIM and InfoNCE, consistently outperforming MIM in classification and regression tasks. At the same time, cMIM preserves reconstruction quality, suggesting comparable generative performance, though further evaluation on broader generative tasks is warranted.

We also proposed *informative embeddings*, a simple method for extracting enriched representations from encoder–decoder models, which enhance downstream effectiveness without additional training.

Overall, cMIM advances the goal of unifying discriminative and generative representation learning. We hope this work provides a foundation for models that excel across a wide range of tasks and motivates further research in this direction. In particular, the augmentation-free contrastive design and informative embeddings indicate natural applicability to large-scale and multimodal domains, where designing positive augmentations is often challenging or ill-defined.

References

Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A. Saurous, and Kevin Murphy. Fixing a broken elbo. In *International Conference on Machine Learning (ICML)*, pages 159–168, 2018.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. *arXiv preprint arXiv:1305.6663*, 2013.

Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Inc., 2009.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2172–2180, 2016.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2021.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: an extension of MNIST to handwritten letters. *CoRR*, abs/1702.05373, 2017. URL `http://arxiv.org/abs/1702.05373`.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. doi: 10.1109/MSP.2012.2211477.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, 2020.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations (ICLR)*, 2017.

R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Khurram Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.

Ross Irwin, Spyridon Dimitriadis, Jiazhen He, and Esben Jannik Bjerrum. Chemformer: a pre-trained transformer for computational chemistry. *Machine Learning: Science and Technology*, 3(1):015022, 2022. doi: 10.1088/2632-2153/ac3ffb. URL `https://doi.org/10.1088/2632-2153/ac3ffb`.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 4651–4664. PMLR, 2021.

Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A. Shoemaker, Paul A. Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E. Bolton. Pubchem 2019 update: improved access to chemical data. *Nucleic Acids Research*, 47(D1):D1102–D1109, 2018. doi: 10.1093/nar/gky1033. URL https://doi.org/10.1093/nar/gky1033.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Oleksii Kuchaiev, Jason Li, Huyen Nguyen, Oleksii Hrinchuk, Ryan Leary, Boris Ginsburg, Samuel Kriman, Stanislav Beliaev, Vitaly Lavrukhin, Jack Cook, et al. Nemo: a toolkit for building ai applications using neural modules. *arXiv preprint arXiv:1909.09577*, 2019.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Nv-embed: Improved techniques for training llms as generalist embedding models. *arXiv preprint arXiv:2405.17428*, 2024.

Micha Livne, Kevin Swersky, and David J. Fleet. MIM: Mutual Information Machine. *arXiv preprint arXiv:1910.03175*, 2019.

Micha Livne, Kevin Swersky, and David J Fleet. Sentencemim: A latent variable language model. *arXiv preprint arXiv:2003.02645*, 2020.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011. URL http://jmlr.org/papers/v12/pedregosa11a.html.

Danny Reidenbach, Micha Livne, Rajesh K. Ilango, Michelle Gill, and Johnny Israeli. Improving small molecule generation using mutual information machine. *arXiv preprint arXiv:2208.09016*, 2023.

Teague Sterling and John J. Irwin. Zinc 15 – ligand discovery for everyone. *Journal of Chemical Information and Modeling*, 55(11):2324–2337, 2015. doi: 10.1021/acs.jcim.5b00559. URL https://doi.org/10.1021/acs.jcim.5b00559.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

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 (NeurIPS)*, volume 30, 2017.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 9929–9939. PMLR, 2020.

David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988. doi: 10.1021/ci00057a005. URL https://doi.org/10.1021/ci00057a005.

Robin Winter, Floriane Montanari, Frank Noé, and Djork-Arné Clevert. Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. *Chemical Science*, 10:1692–1701, 2019. doi: 10.1039/C8SC04175J. URL http://dx.doi.org/10.1039/C8SC04175J.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. *CoRR*, abs/2110.14795, 2021. URL https://arxiv.org/abs/2110.14795.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. Infovae: Information maximizing variational autoencoders. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 5885–5892, 2017.

A Extended Formulation

Background: Contrastive Learning

Contrastive learning maximizes similarity of positive pairs and minimizes similarity to negatives. With cosine similarity $\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j) = \frac{\boldsymbol{z}_i \cdot \boldsymbol{z}_j}{\|\boldsymbol{z}_i\| \, \|\boldsymbol{z}_j\|}$ and temperature τ, define $g(\boldsymbol{z}_i, \boldsymbol{z}_j) = \exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)$. The InfoNCE loss per sample is

$$\mathrm{InfoNCE}(\boldsymbol{x}_i, \boldsymbol{x}_i^+) = -\log\left(\frac{g(\boldsymbol{z}_i, \boldsymbol{z}_i^+)}{\sum_{j=1}^{B} g(\boldsymbol{z}_i, \boldsymbol{z}_j)}\right). \tag{4}$$

cMIM without Data Augmentation

We extend the MIM graphical model with a binary variable k and define the joint factorizations

$$q_\theta(\boldsymbol{x}, \boldsymbol{z}, \mathrm{k}) = q_\theta(\mathrm{k} \mid \boldsymbol{x}, \boldsymbol{z})\, q_\theta(\boldsymbol{z} \mid \boldsymbol{x})\, q_\theta(\boldsymbol{x}), \qquad p_\theta(\boldsymbol{x}, \boldsymbol{z}, \mathrm{k}) = p_\theta(\mathrm{k} \mid \boldsymbol{x}, \boldsymbol{z})\, p_\theta(\boldsymbol{x} \mid \boldsymbol{z})\, p_\theta(\boldsymbol{z}). \tag{5}$$

With $\boldsymbol{z}_i \sim q_\theta(\boldsymbol{z} \mid \boldsymbol{x}_i)$, the discriminator over k shares parameters in both paths and is Bernoulli with success probability

$$p_{k=1}(\boldsymbol{x}_i, \boldsymbol{z}_i) = \frac{g(\boldsymbol{z}_i, \boldsymbol{z}_i)}{g(\boldsymbol{z}_i, \boldsymbol{z}_i) + \mathbb{E}_{\boldsymbol{x}' \sim p(\boldsymbol{x}),\, \boldsymbol{z}' \sim q_\theta(\boldsymbol{z} \mid \boldsymbol{x}')}[g(\boldsymbol{z}_i, \boldsymbol{z}')]} \approx \frac{g(\boldsymbol{z}_i, \boldsymbol{z}_i)}{g(\boldsymbol{z}_i, \boldsymbol{z}_i) + \frac{1}{B-1} \sum_{\substack{j=1 \\ j \neq i}}^{B} g(\boldsymbol{z}_i, \boldsymbol{z}_j)}. \tag{6}$$

During training, $\mathrm{k} = 1$ because $\boldsymbol{z}_i$ is sampled given $\boldsymbol{x}_i$; the expectation is approximated in-batch.

Concentration. Since cosine similarity lies in $[-1, 1]$, $g \in [\mathrm{e}^{-1/\tau}, \mathrm{e}^{1/\tau}]$. Hoeffding's inequality implies the in-batch Monte Carlo estimate of the negative mean concentrates around its expectation:

$$\Pr\left(\left|\frac{1}{B-1} \sum_{j \neq i} g(\boldsymbol{z}_i, \boldsymbol{z}_j) - \mu\right| \geq \epsilon\right) \leq 2 \exp\left(-\frac{2(B-1)\epsilon^2}{(\mathrm{e}^{1/\tau} - \mathrm{e}^{-1/\tau})^2}\right). \tag{7}$$

Relation to InfoNCE

Let $s_{ij} = \mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau$ so $g(\boldsymbol{z}_i, \boldsymbol{z}_j) = \exp(s_{ij})$. Then from (6),

$$\begin{aligned} p_{k=1} &= \frac{\exp(s_{ii})}{\exp(s_{ii}) + \frac{1}{B-1} \sum_{j \neq i} \exp(s_{ij})} \\ &= \frac{\exp\big(s_{ii} + \log(B-1)\big)}{\exp\big(s_{ii} + \log(B-1)\big) + \sum_{j \neq i} \exp(s_{ij})}. \end{aligned} \tag{8}$$

Thus $-\log p_{k=1}$ equals an InfoNCE cross-entropy where the positive logit is shifted by $\log(B-1)$. Calibration: if all logits are equal, $p_{k=1} = 1/2$ (independent of B) versus $1/B$ in standard InfoNCE. With cosine similarity $s_{ii} = 1/\tau$ is constant; attraction comes from the MIM term.

cMIM Training Objective

Define the mixture model

$$\mathcal{M}_\theta(\boldsymbol{x}, \boldsymbol{z}, \mathrm{k}) = \tfrac{1}{2}\big(p_\theta(\mathrm{k} \mid \boldsymbol{z}, \boldsymbol{x})\, p_\theta(\boldsymbol{x} \mid \boldsymbol{z})\, p_\theta(\boldsymbol{z}) + q_\theta(\mathrm{k} \mid \boldsymbol{z}, \boldsymbol{x})\, q_\theta(\boldsymbol{z} \mid \boldsymbol{x})\, q_\theta(\boldsymbol{x})\big), \tag{9}$$

with sampling distribution $\mathcal{M}_S(\boldsymbol{x}, \boldsymbol{z}, \mathrm{k})$ as in MIM . The learning objective upper-bounds the negative mixture entropy:

$$\mathcal{L}_{\text{MIM}}(\theta) = \tfrac{1}{2}\Big(\mathrm{CE}(\mathcal{M}_S, q_\theta) + \mathrm{CE}(\mathcal{M}_S, p_\theta)\Big) \geq H_{\mathcal{M}_S}(\boldsymbol{x}, \mathrm{k}) + H_{\mathcal{M}_S}(\boldsymbol{z}) - I_{\mathcal{M}_S}(\boldsymbol{x}, \mathrm{k}; \boldsymbol{z}). \tag{10}$$

For A-MIM (sampling along the encoding path),

$$\begin{aligned} \mathcal{L}_{\text{A-MIM}}(\theta) = -\tfrac{1}{2}\, \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x}),\, \boldsymbol{z} \sim q_\theta(\boldsymbol{z} \mid \boldsymbol{x}),\, \mathrm{k}=1}\Big[&\log p_\theta(\mathrm{k} \mid \boldsymbol{z}, \boldsymbol{x}) + \log p_\theta(\boldsymbol{x} \mid \boldsymbol{z}) + \log p_\theta(\boldsymbol{z}) \\ &+ \log q_\theta(\mathrm{k} \mid \boldsymbol{z}, \boldsymbol{x}) + \log q_\theta(\boldsymbol{z} \mid \boldsymbol{x}) + \log q_\theta(\boldsymbol{x})\Big]. \end{aligned} \tag{11}$$

The empirical loss with anchor prior $p(\boldsymbol{z}) = \mathcal{N}(0, \mathbf{I})$ is

$$\hat{\mathcal{L}}_{\text{A-MIM}}(\theta; \mathcal{D}) = -\frac{1}{N} \sum_{i=1}^{N} \Big(\log p_\theta(\boldsymbol{x}_i \mid \boldsymbol{z}_i) + \log p_{k=1}(\boldsymbol{x}_i, \boldsymbol{z}_i) + \frac{1}{2}(\log q_\theta(\boldsymbol{z}_i \mid \boldsymbol{x}_i) + \log p(\boldsymbol{z}_i)) \Big). \tag{12}$$

B Experiment Training Details

B.1 Image Classification

We opted for a simple architecture.

- The encoder flattens the image to 784 dimensions, up-projects using a linear layer to $(784, 16)$ which is fed to a Perceiver encoder that projects it down to 400 steps $(400, 16)$. A linear layer projects the hidden dimension to 1, followed by a layer norm, and finally a linear projection from 400 to 64.
- The encoding distribution is a Gaussian with mean and variance predicted by linear layers from the encoder output.
- The decoder up-projects the 64 dimension latent code using a linear layer to $(64, 16)$ which is fed to a Perceiver encoder that projects it down to 400 steps $(400, 16)$. A linear layer projects the hidden dimension to 1, followed by a layer norm, and finally a linear projection from 400 to 784, which is reshaped back to $(28, 28)$ image dimensions.
- The decoding distribution is a conditional Bernoulli with logits predicted by a linear layer from the decoder output.
- The prior is a standard Gaussian.

All models were trained with Adam optimizer with learning rate $1e-3$ and WSD scheduler with 10% warmup steps and 10% decay steps, for a total of 500k steps (regardless of the batch size).

B.2 Molecualr Property Prediction

Dataset: All models were trained using a tranche of the ZINC-15 dataset [Sterling and Irwin, 2015], labeled as reactive and annotated, with molecular weight $\leq$ 500Da and logP ≤ 5. Of these molecules, 730M were selected at random and split into training, testing, and validation sets, with 723M molecules in the training set. We note that we do not explore the effect of model size, hyperparameters, and data on the models. Instead, we train all models on the same data using the same hyperparameters, focusing on the effect of the learning framework and the fixed-size bottleneck. For comparison, Chemformer was trained on 100M molecules from ZINC-15 [Sterling and Irwin, 2015] – 20X the size of the dataset used to train CDDD (72M from ZINC-15 and PubChem [Kim et al., 2018]). MoLFormer-XL was trained on 1.1 billion molecules from the PubChem and ZINC datasets.

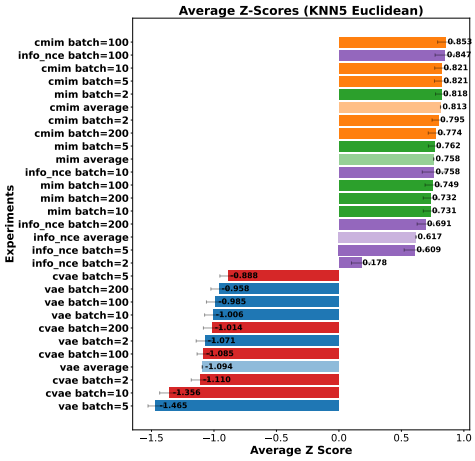
Data augmentation: Following Irwin et al. [2022], we used two augmentation methods: masking, and SMILES enumeration [Weininger, 1988]. Masking is as described for the BART MLM denoising objective, with 10% of the tokens being masked, and was only used during the training of MegaMolBART. In addition, MegaMolBART, PerBART, and MolVAE used SMILES enumeration where the encoder and decoder received different valid permutations of the input SMILES string. MolMIM was the only model to see an increase in performance when both the encoder and decoder received the same input SMILES permutation, simplifying the training procedure.

Model details: We implemented all models with NeMo Megatron toolkit [Kuchaiev et al., 2019]. We used a RegEx tokenizer with 523 tokens [Bird et al., 2009]. All models had 6 layers in the encoder and 6 layers in the decoder, with a hidden size of 512, 8 attention heads, and a feed-forward dimension of 2048. The Perceiver-based models also required defining K, the hidden length, which relates to the hidden dimension by $H = K \times D$ where H is the total hidden dimension, and D is the model dimension (Fig. 3). MegaMolBART had $58.9M$ parameters, PerBART had $64.6M$, and MolVAE and MolMIM had $65.2M$. We used greedy decoding in all experiments. We note that we trained MolVAE using the loss of β-VAE [Higgins et al., 2017] where we scaled the KL divergence term with $\beta = \frac{1}{D}$ where D is the hidden dimensions.

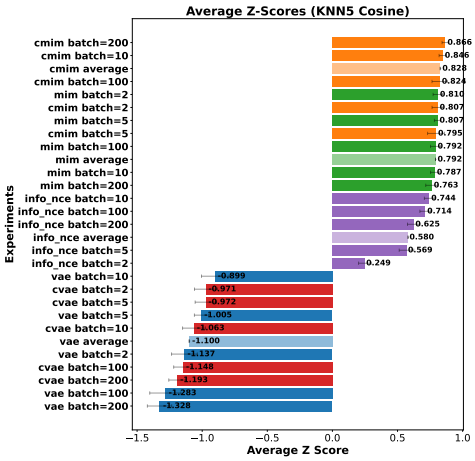
Optimization: We use ADAM optimizer [Kingma and Ba, 2015] with a learning rate of 1.0, betas of 0.9 and 0.999, weight decay of 0.0, and an epsilon value of 1.0e-8. We used Noam learning rate scheduler [Vaswani et al., 2017] with a warm-up ratio of 0.008, and a minimum learning rate of 1e-5. During training, we used a maximum sequence length of 512, dropout of 0.1, local batch size of 256, and global batch size of 16384. All models were trained for 1,000,000 steps with fp16 precision for 40 hours on 4 nodes with 16 GPU/node (Tesla V100 32GB). MolVAE was trained using β-VAE [Higgins et al., 2017] with $\beta = \frac{1}{H}$ where H is the number of hidden dimensions. We have found this choice to provide a reasonable balance between the rate and distortion (see Alemi et al. [2018] for details). It is important to note that MolMIM does not require the same β hyperparameter tuning as done for VAE.

C Additional Results

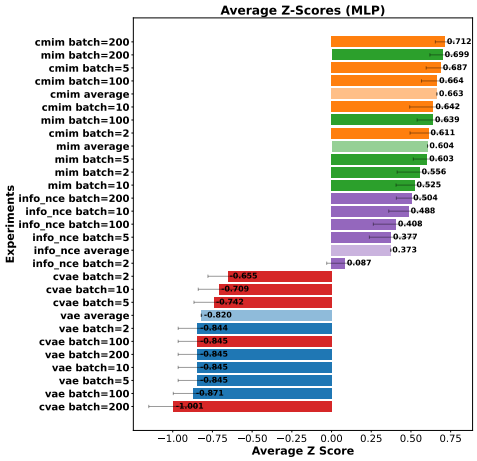
C.1 MNIST-like Image Classification



(a) KNN5 Euclidean

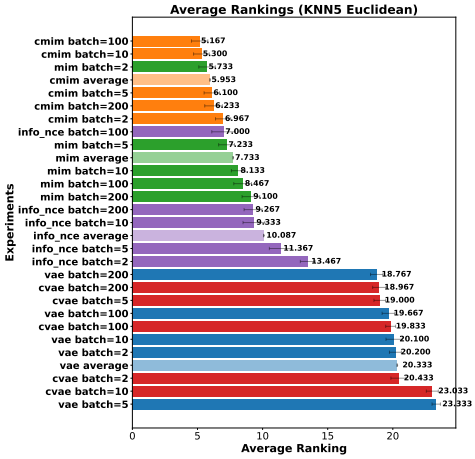


(b) KNN5 Cosine

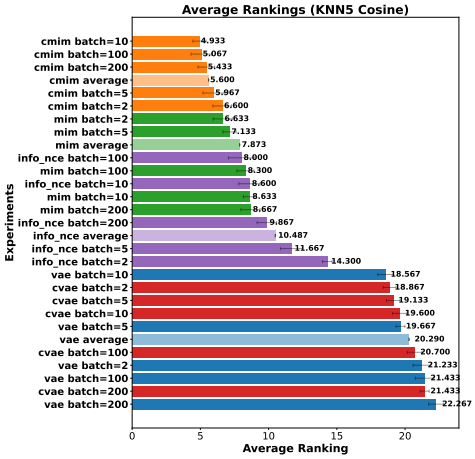


(c) MLP

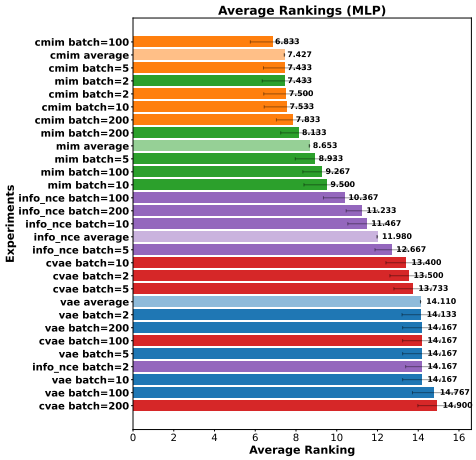
Figure 8: Z-scores with error bars for MNIST-like image classification tasks using different evaluation methods.



(a) KNN5 Euclidean



(b) KNN5 Cosine



(c) MLP

Figure 9: Rankings with error bars for MNIST-like image classification tasks using different evaluation methods.