TwinTURBO: Semi-Supervised Fine-Tuning of Foundation Models via Mutual Information Decompositions for Downstream Task and Latent Spaces

Guillaume Quétant[*]
Dept. of Computer Science
University of Geneva
Carouge, Switzerland

Pavlo Molchanov
NVIDIA

Slava Voloshynovskiy[*]
Dept. of Computer Science
University of Geneva
Carouge, Switzerland

Abstract

We present a semi-supervised fine-tuning framework for foundation models that utilises mutual information decomposition to address the challenges of training for a limited amount of labelled data. Our approach derives two distinct lower bounds: i) for the downstream task space, such as classification, optimised using conditional and marginal cross-entropy alongside Kullback-Leibler divergence, and ii) for the latent space representation, regularised and aligned using a contrastive-like decomposition. This fine-tuning strategy retains the pre-trained structure of the foundation model, modifying only a specialised projector module comprising a small transformer and a token aggregation technique. Experiments on several datasets demonstrate significant improvements in classification tasks under extremely low-labelled conditions by effectively leveraging unlabelled data.

1 Introduction

Foundation models are large-scale neural networks pre-trained on diverse data to learn general-purpose representations that can be fine-tuned for specific downstream tasks. This poses significant challenges, especially in the case of low-labelled data, a semi-supervised learning setting where only a small fraction of the data samples are labelled, while the majority remain unlabelled. While foundation models are pre-trained on large datasets in a self-supervised manner, their deployment often requires fine-tuning on new datasets with limited labelled samples and potential distribution shifts. Furthermore, the downstream tasks frequently differ from the pre-training objectives, complicating the adaptation process. Existing semi-supervised approaches, such as pseudo-labelling, rely heavily on assumptions about data distributions or task-specific tuning, limiting their generalisability. Addressing these challenges is essential to fully exploit the potential of foundation models and ensure their adaptability and scalability in diverse applications. The main contributions of this study are:

- **A new framework for foundation models fine-tuning:** We introduces a fine-tuning strategy based on mutual information decomposition. This framework guides the adaptation of foundation models to downstream tasks and latent space representations, providing a theoretically sound and efficient alternative to traditional supervised learning.
- **Novel theoretical insights:** The framework derives two distinct mutual information bounds: one for optimising downstream task performance and another for aligning latent space

[*]Corresponding authors: guillaume.quetant@unige.ch, svolos@unige.ch

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Structured Probabilistic Inference & Generative Modeling (SPIGM).

representations. These insights deepen our understanding of how information can be effectively utilised in semi-supervised learning.

- **Practical validation:** Extensive experiments on classification tasks under extremely low-labelled data conditions demonstrate significant performance improvements. These results highlight the effectiveness of focusing on theoretical decompositions over heuristic-based approaches, and reveal previously underexplored opportunities in semi-supervised fine-tuning.

TwinTURBO builds on and extends existing work in semi-supervised learning and mutual information-based optimisation [1; 2; 3]. Traditional approaches often emphasise pseudo-label generation or global fine-tuning strategies, which can be computationally expensive and less adaptable to diverse tasks. Our formalism provides a lightweight and modular framework that combines pre-trained vision foundation models with a specialised adapter module. This approach integrates with state-of-the-art methods such as InfoNCE-based contrastive learning [4; 5; 6; 7], while addressing their limitations by providing a unified perspective on labelled and unlabelled semi-supervised fine-tuning.

2 Related work

Semi-Supervised Machine Learning. Semi-supervised learning methods aim to exploit large amounts of unlabelled data to improve performance when labelled data is scarce. Traditional approaches include pseudo-labelling [8], consistency regularisation [9; 10], and entropy minimisation [11]. In the context of semi-supervised learning for foundation models, recent works explore the fine-tuning of pre-trained models with minimal labelled data while maximising the use of unlabelled samples. Methods such as FixMatch [12] and FlexMatch [13] refine pseudo-labelling by enforcing consistency under data augmentations. These approaches aim to minimise the reliance on large labelled datasets while maintaining strong generalisation capabilities.

Fine-Tuning of Foundation Models. Full-model fine-tuning of large-scale models has been shown to be effective, but computationally expensive. To mitigate the computational cost of full-model fine-tuning, several works introduce adapter-based methods where additional small modules are trained while the pre-trained backbone is frozen. LoRA [14], VeRA [15], DoRA [16] and others introduce low-rank adaptation layers that allow efficient task-specific tuning without updating the entire model. Other studies, such as Compacter [17] and Prompt Tuning [18] have demonstrated that tuning a fraction of the parameters can achieve competitive results compared to full-model fine-tuning, especially in low-data regimes. We also refer the reader to [3] for a recent study on fine-tuning of vision foundation models.

Mutual Information for Machine Learning. Mutual information bounds play a crucial role in modern machine learning by formalising the relationship between observed data, latent representations and downstream task labels. Classical approaches, such as the information bottleneck [19], propose to minimise mutual information between inputs and latent variables while maximising mutual information between latent representations and outputs. More recent works decompose mutual information for machine learning applications. The study presented in [20] introduces a variational lower bound on mutual information. Other information-theoretic studies, such as the variational information bottleneck [21], the exploration of several bounds [22; 23] and Deep InfoMax [24] apply mutual information maximisation to learn robust representations. More recently, TURBO [2] also uses mutual information maximisation to give a meaningful interpretation of a wide family of deep learning models. InfoNCE [4] and its extensions provide tractable mutual information estimates via contrastive learning that is a common basis for modern unimodal self-supervised methods such as SimCLR [5] and DINO [6] and multimodal ones such as CLIP [7].

3 TwinTURBO

TwinTURBO is a framework for guiding semi-supervised fine-tuning of foundation models using mutual information decompositions. It adapts models to downstream tasks or latent spaces by leveraging data and statistical structures, bridging theoretical insights with practical applications, while connecting to and advancing state-of-the-art approaches.

3.1 Decomposition principle

We consider the problem of predicting, from a random variable $\mathbf{X}$, the random variable $\mathbf{Y} = f(\mathbf{X})$, for a given function f.[2] This function can usually be stochastic and is thought as any process that represent a physical or virtual relation between the two modalities $\mathbf{X}$ and $\mathbf{Y}$. Once the function parameterised by a neural network f_θ, where θ denotes the parameters of the network, we look for a meaningful description of the training procedures utilised in modern machine learning. A natural consideration is to express the mutual information between $\mathbf{X}$ and $\mathbf{Y}$

$$I(\mathbf{X};\mathbf{Y}) = \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{y}|\mathbf{x})}{p(\mathbf{y})}\right], \tag{1}$$

and take it as a maximum objective towards which we want to push our neural network. To achieve this, we inject a parametric conditional density $p_\theta(\mathbf{y}|\mathbf{x})$, characterised by the network outputs, and use the bound proposed in [20]

$$\begin{aligned} I(\mathbf{X};\mathbf{Y}) &= \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p_\theta(\mathbf{y}|\mathbf{x})}{p(\mathbf{y})}\right] + \mathbb{E}_{p(\mathbf{x})}\left[D_{\mathrm{KL}}\left(p(\mathbf{y}|\mathbf{x}) \,||\, p_\theta(\mathbf{y}|\mathbf{x})\right)\right] \\ &\geq \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p_\theta(\mathbf{y}|\mathbf{x})}{p(\mathbf{y})}\right], \end{aligned} \tag{2}$$

in order to get a parametric lower bound to maximise. However, practical implementations of such a term typically require pairs of data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$ in order to evaluate the only θ-dependent part $\mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log\, p_\theta(\mathbf{y}|\mathbf{x})\right] = -H(p(\mathbf{y}|\mathbf{x}); p_\theta(\mathbf{y}|\mathbf{x}))$, namely the negative conditional cross-entropy. Although suitable for some simple datasets, the vast majority of modern problems involve unlabelled data, or could greatly benefit from its use. To also exploit the unpaired data, similarly to the development described in [2], we inject the parametric marginal density $p_\theta(\mathbf{y})$ and re-express the bound as

$$I(\mathbf{X};\mathbf{Y}) \geq \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p_\theta(\mathbf{y}|\mathbf{x})}{p_\theta(\mathbf{y})}\right] - D_{\mathrm{KL}}\left(p(\mathbf{y}) \,||\, p_\theta(\mathbf{y})\right). \tag{3}$$

This formulation has the benefit of involving an additional *divergence* term between the real and parametric marginal densities, which can be used in certain situations to ensure better coverage of the data distribution. In practice, this term is usually implemented as a discriminator classifier. The other appearance of the parametric marginal density through the cross-entropy $-\mathbb{E}_{p(\mathbf{y})}\left[\log\, p_\theta(\mathbf{y})\right] = H(p(\mathbf{y}); p_\theta(\mathbf{y}))$ leads to so-called *contrastive*-like training. Indeed, it creates an opposition between the likelihood of samples from paired and unpaired data, often referred to as positive and negative pairs, respectively. We can also neglect the cross-entropy $H(p(\mathbf{y}); p_\theta(\mathbf{y}))$ since it is non-negative, in principle for discrete random variable only, as presented in [2]. This study argues that if it is appropriate to the problem at hand, it should be retained.

3.2 Density parameterisations

As a generic formula, any parametric conditional density $p_\theta(\mathbf{y}|\mathbf{x})$ can be expressed as a carefully normalised exponential function such as

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{y})}{Z_\theta(\mathbf{x})}\, e^{s_\theta(\mathbf{x},\mathbf{y})}, \qquad Z_\theta(\mathbf{x}) = \sum_{\mathbf{y}\in\mathcal{Y}} p(\mathbf{y})\, e^{s_\theta(\mathbf{x},\mathbf{y})}, \tag{4}$$

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\theta(\mathbf{x})}\, e^{s_\theta(\mathbf{x},\mathbf{y})}, \qquad Z_\theta(\mathbf{x}) = \sum_{\mathbf{y}\in\mathcal{Y}} e^{s_\theta(\mathbf{x},\mathbf{y})}, \tag{5}$$

$$p_\theta(\mathbf{y}|\mathbf{x}) = e^{s_\theta(\mathbf{x},\mathbf{y})}, \qquad 1 = \sum_{\mathbf{y}\in\mathcal{Y}} e^{s_\theta(\mathbf{x},\mathbf{y})}, \tag{6}$$

where $Z_\theta(\mathbf{x})$ is the partition function and $s_\theta(\mathbf{x},\mathbf{y})$ is a score associated to the $(\mathbf{x},\mathbf{y})$ pair. We call Eqs. (4) to (6) the *scaled-normalised*, *globally-normalised* and *self-normalised* density cases, respectively. Note that the partition function of the scaled-normalised density can be expressed as $Z_\theta(\mathbf{x}) = \sum_{\mathbf{y}\in\mathcal{Y}} p(\mathbf{y})\, e^{s_\theta(\mathbf{x},\mathbf{y})} = \mathbb{E}_{p(\mathbf{y})}\left[e^{s_\theta(\mathbf{x},\mathbf{y})}\right]$. With these expressions at hands, it is straightforward to define the corresponding marginal density as

$$p_\theta(\mathbf{y}) = \mathbb{E}_{p(\mathbf{x})}\left[p_\theta(\mathbf{y}|\mathbf{x})\right]. \tag{7}$$

[2] Capital letters are reserved for random variables and small letters for their realisations.

3.3 Bound formulations

The lower bound and density parameterisations presented so far are theoretical formulas valid in general. In order to formulate proper loss functions, approximations are often necessary. We present here the generic derivations of the objectives used in our experiments. The translation into the actual setup and notations will be discussed later.

Categorical Cross-Entropy. The formalism leads to the popular categorical cross-entropy loss when a carefully chosen score function is used. Let us define $s_\theta(\mathbf{x}, \mathbf{y}) = \mathbf{y} \cdot f_\theta(\mathbf{x}) = \sum_{c=1}^{C} y_c \, f_\theta(\mathbf{x})_c$, where the vector-valued function f_θ is a neural network often called a classifier and where y_c is the c-th component of the one-hot vector $\mathbf{y} \in \{0, 1\}^C$ indicating to which of the C classes the data $\mathbf{x}$ belongs. The normalisation of the globally-normalised density of Eq. (5) becomes the sum over the C classes $Z_\theta(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{Y}} e^{\,\mathbf{y} \cdot f_\theta(\mathbf{x})} = \sum_{c=1}^{C} e^{\,f_\theta(\mathbf{x})_c}$ and the conditional density finally becomes the *softmax* function

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{e^{\,\mathbf{y} \cdot f_\theta(\mathbf{x})}}{\sum_{\mathbf{y}' \in \mathcal{Y}} e^{\,\mathbf{y}' \cdot f_\theta(\mathbf{x})}}. \tag{8}$$

The role of the softmax function is to rescale the predictions of the classifier to values in the interval $(0, 1)$ and that sum to 1. Softening the latter condition, we can use the self-normalised density parameterisation of Eq. (6) instead and replace the rescaling with the *sigmoid* function

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{e^{\,\mathbf{y} \cdot f_\theta(\mathbf{x})}}{1 + e^{\,\mathbf{y} \cdot f_\theta(\mathbf{x})}}. \tag{9}$$

We concisely denote both formulations as $p_\theta(\mathbf{y}|\mathbf{x}) = \sigma_{\mathbf{y}}(f_\theta(\mathbf{x}))$.

In most applications, categorical cross-entropy training problems only take into account the conditional term $\mathbb{E}_{p(\mathbf{x},\mathbf{y})} \left[\log \, p_\theta(\mathbf{y}|\mathbf{x})\right] = -H(p(\mathbf{y}|\mathbf{x}); p_\theta(\mathbf{y}|\mathbf{x}))$. In such a case, sigmoid rescaling leads to a collapse of all classifier outputs to 1 as a trivial maximum. We argue in this study that a better choice is to use the tighter lower bound developed in Eq. (3), leading to

$$I(\mathbf{X}; \mathbf{Y}) \geq \mathbb{E}_{p(\mathbf{x},\mathbf{y})} \left[\log \frac{\sigma_{\mathbf{y}}(f_\theta(\mathbf{x}))}{\mathbb{E}_{p(\mathbf{x}')} \left[\sigma_{\mathbf{y}}(f_\theta(\mathbf{x}'))\right]}\right] - D_{\mathrm{KL}} \left(p(\mathbf{y}) \,||\, p_\theta(\mathbf{y})\right), \tag{10}$$

where the exact nature of the function $\sigma_{\mathbf{y}}$ is left open. Both the denominator of the contrastive term and the divergence prevent the classifier from collapsing.

Binary Cross-Entropy. A potential drawback of this approach is that the log-sum-exp expansion (see Appendix A) cannot be used to compute the denominator term $-\log \mathbb{E}_{p(\mathbf{x}')} \left[\sigma_{\mathbf{y}}(f_\theta(\mathbf{x}'))\right]$ efficiently, which can therefore be numerically unstable. However, we observe that its role in the loss function is to force the classifier to output values closer to 0. This contrasts with the numerator term $\log \sigma_{\mathbf{y}}(f_\theta(\mathbf{x}))$, which forces the output for the correct class towards 1. A stable alternative that mimic this contrastive goal can be found in binary cross-entropy. Indeed, the categorical cross-entropy formulation assumes that the predicted probability of the class $\mathbf{y}$ is fully given by the corresponding output of the classifier. The binary cross-entropy formulation includes all the outputs and tries to push not only the correct one towards 1, but also the wrong ones towards 0. In essence, this has a similar role as contrastive learning. This is achieved by reformulating the conditional density as the product of the probability that each classifier output gives the correct answer

$$p_\theta(\mathbf{y}|\mathbf{x}) = \prod_{\substack{c=1 \\ y_c=1}}^{C} \left[\sigma(f_\theta(\mathbf{x}))_c\right] \cdot \prod_{\substack{c=1 \\ y_c=0}}^{C} \left[1 - \sigma(f_\theta(\mathbf{x}))_c\right], \tag{11}$$

where y_c is the c-th component of the one-hot vector $\mathbf{y}$ and where σ (without subscript) is the vector-valued softmax or sigmoid function applied to the vector $f_\theta(\mathbf{x})$.[3] Plugging this expression in Eq. (3) leads to the usual binary cross-entropy

$$\begin{aligned} I(\mathbf{X}; \mathbf{Y}) \geq \; & \mathbb{E}_{p(\mathbf{x},\mathbf{y})} \left[\sum_{c=1}^{C} y_c \log \sigma(f_\theta(\mathbf{x}))_c\right] + \mathbb{E}_{p(\mathbf{x},\mathbf{y})} \left[\sum_{c=1}^{C} (1 - y_c) \log(1 - \sigma(f_\theta(\mathbf{x}))_c)\right] \\ & - D_{\mathrm{KL}} \left(p(\mathbf{y}) \,||\, p_\theta(\mathbf{y})\right), \end{aligned} \tag{12}$$

[3]The relation between $\sigma_{\mathbf{y}}$ and σ can be understood as $\sigma_{\mathbf{y}}(f_\theta(\mathbf{x})) = \mathbf{y} \cdot \sigma(f_\theta(\mathbf{x}))$ for a one-hot vector $\mathbf{y}$.

where the neglected unstable denominator is compensated for by a conditional density designed to achieve the same objective.

InfoNCE. We develop here two slightly different ways to implement the InfoNCE loss. By using our TwinTURBO formalism of Eq. (3), we can plug the self-normalised density of Eq. (6) to get

$$I(\mathbf{X};\mathbf{Y}) \geq \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{e^{s_\theta(\mathbf{x},\mathbf{y})}}{\mathbb{E}_{p(\mathbf{x}')}\left[e^{s_\theta(\mathbf{x}',\mathbf{y})}\right]}\right] - D_{\mathrm{KL}}\left(p(\mathbf{y})\,||\,p_\theta(\mathbf{y})\right). \tag{13}$$

The choice of the exact form of the score $s_\theta(\mathbf{x},\mathbf{y})$ is free. This formulation needs the divergence term in order to keep being a lower bound on the mutual information. For adapted problems, this additional constraint on the global shape of the data manifold can be very helpful. However, the score function then needs to be flexible enough to properly self-normalise.

If the dataset does not allow a proper calculation of the divergence, e.g. because there is no meaningful prior on the marginal density, another lower bound is required. A simple way the recover the usual InfoNCE loss is to depart from the bound of Eq. (2) originally proposed in [20] and use the scaled-normalised density of Eq. (4)

$$I(\mathbf{X};\mathbf{Y}) \geq \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{\frac{p(\mathbf{y})}{Z_\theta(\mathbf{x})}\,e^{s_\theta(\mathbf{x},\mathbf{y})}}{p(\mathbf{y})}\right] = \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{e^{s_\theta(\mathbf{x},\mathbf{y})}}{\mathbb{E}_{p(\mathbf{y}')}\left[e^{s_\theta(\mathbf{x},\mathbf{y}')}\right]}\right]. \tag{14}$$

Note the subtle but crucial difference between the expectations in the denominator of Eq. (13), derived from the marginal density $p_\theta(\mathbf{y})$, and the denominator of Eq. (14), derived from the partition function $Z_\theta(\mathbf{x})$.

Discriminator. A Kullback-Leibler divergence term is present in Eqs. (10), (12) and (13). Unfortunately, this quantity is intractable in practice and usually replaced by the symmetric Jensen-Shannon divergence. In order to compute it, the density ratio trick is often used through a discriminator network. It follows from defining the ratio of densities as $r_\theta(\mathbf{y}) = \frac{p(\mathbf{y})}{p_\theta(\mathbf{y})} \equiv \frac{p_\alpha(\mathbf{y}|i=1)}{p_\alpha(\mathbf{y}|i=0)} = \frac{p_\alpha(i=1|\mathbf{y})}{p_\alpha(i=0|\mathbf{y})}$ where $p(i=0) = p(i=1)$ for an indicative variable i and a set of parameters α.[4] By defining a discriminator network $D_\alpha(\mathbf{y}) := p_\alpha(i=1|\mathbf{y})$, we can express the Jensen-Shannon divergence as

$$D_{\mathrm{JS}}\left(p(\mathbf{y})\,||\,p_\theta(\mathbf{y})\right) = \frac{1}{2}\mathbb{E}_{p(\mathbf{y})}\left[\log(D_\alpha(\mathbf{y}))\right] + \frac{1}{2}\mathbb{E}_{p_\theta(\mathbf{y})}\left[\log(1-D_\alpha(\mathbf{y}))\right] + \log 2, \tag{15}$$

which is the negative binary cross-entropy, up to the non-trainable $\log 2$ term and $1/2$ factor. We can now alternatively train the discriminator D_α to maximise and the predictor f_θ to minimise Eq. (15). A perfectly trained discriminator would also allow to estimate the Kullback-Leibler divergence via

$$D_{\mathrm{KL}}\left(p(\mathbf{y})\,||\,p_\theta(\mathbf{y})\right) = \mathbb{E}_{p(\mathbf{y})}\left[\log(D_\alpha(\mathbf{y}))\right] - \mathbb{E}_{p(\mathbf{y})}\left[\log(1-D_\alpha(\mathbf{y}))\right], \tag{16}$$

but this expression does not depend on θ and cannot be used to train f_θ. Therefore, as is common practice, we use Eq. (15) as a proxy for the Kullback-Leibler divergence in Eqs. (10), (12) and (13).

4 Experiments

We test our TwinTURBO formalism on several semi-supervised classification experiments where the number of labelled data is kept below 0.2% and 2%.

4.1 Datasets

The datasets examined are MNIST [25], CIFAR10 [26] and SVHN [27]. MNIST provides 60'000 greyscale handwritten digits from 0 to 9 and is considered to be an easy dataset to learn. CIFAR10 is an analogous dataset that focuses on 50'000 natural images. It contains 10 classes of low resolution photographs of animals and vehicles. SVHN is a much harder dataset of 73'257 house numbers taken from the street. Only the central digit of the image determines the class. Therefore, it has a lot of contamination from surrounding numbers and background, making it a perfect benchmark for testing our framework. All three datasets provide an additional test set of 10'000 samples.

[4] The θ dependency is effectively present through $\mathbf{y} = f_\theta(\mathbf{x})$.

The semi-supervised setup is constructed by creating a dual dataset that samples in parallel all images with labels removed and a small duplicated subset of 100 or 1000 labelled images. This reflects a situation with very little labelled data. The labelled samples are used to compute the supervised cross-entropy and latent regularisation losses, while the unlabelled samples are used by the discriminator and the self-supervised latent regularisation loss, as described below.

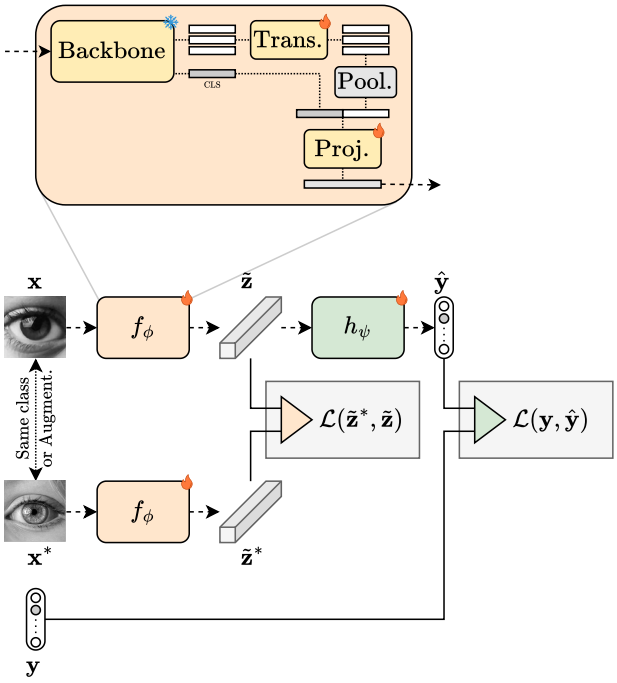

Figure 1: Model architecture and training setup.

4.2 Model architecture

To fully exploit the dataset and the TwinTURBO formalism, our fine-tuning model follows an encoder-predictor architecture as depicted in Fig. 1. The encoder f_ϕ consists of a frozen RADIOv2.5-B [28; 29; 30] foundation model backbone, plus a trainable transformer and projector.[5] The backbone takes an image $\mathbf{x}$ as input, creates patches from it and outputs several patch tokens and a global representation, also called a CLS token. Most classification experiments based on foundation model representations use only the CLS token, as it corresponds to the global features extracted from the input image. However, there is no guarantee that it contains all the necessary information, as certain local details may still be useful to identify the content of an image. Therefore, inspired by [31], we also consider the patch tokens. They are passed in a single layer transformer to allow cross-interactions before an average pooling operation is performed.[6] The resulting token is concatenated with the CLS token and a final projector produces the latent representation $\tilde{\mathbf{z}} = f_\phi(\mathbf{x})$. The role of the second copy of the encoder as well as the $\mathbf{x}^*$ and $\tilde{\mathbf{z}}^*$ variables in the latent space alignment will be explained later. The predictor h_ψ is a classifier head that predicts the class $\hat{\mathbf{y}} = h_\psi(\tilde{\mathbf{z}})$ of the input image from the latent representation $\tilde{\mathbf{z}}$. We emphasise again that the foundation model is frozen, even though for brevity we refer to the whole backbone-transformer-projector block as f_ϕ. This is also a hint to the seamless extension of the framework to a trainable backbone. Our network architecture follows [1; 3]. More details are available in Appendix B.

4.3 Losses

The TwinTURBO formalism is used in both the latent space $\mathbf{Z}$ and downstream task space $\mathbf{Y}$ of our setup. We translate in Appendix C the lower bounds on mutual information presented above into losses to be minimised in each of these spaces and show the final combined objective bellow.

Combined loss. The combined total loss used to train the entire model is

$$\mathcal{L} = \mathcal{L}^{\text{cat-cross/cat-twin/bin-cross}} + \lambda_C \, \mathcal{L}^{\text{critic}} + \lambda_L \, \mathcal{L}^{\text{latent}} + \lambda_A \, \mathcal{L}^{\text{augment.}}, \tag{17}$$

where λ_C, λ_L and λ_A are weight hyperparameters that trade-off the different loss components. In the following these weights and their corresponding loss terms will be referred to as *critic*, *latent* and

[5] Only the RADIOv2.5-B is tested, as it shows state-of-the-art results in [3] and a larger model would require too expensive computing capacity. Furthermore, we focus on a theoretically grounded study and present a proof of principle. It is straightforward to apply the same setup to any other foundation model.

[6] We do not pass the CLS token through the transformer layer because i) the backbone is assumed to have already extracted the required features from the patches and ii) the higher dimension of the CLS token relative to the patch tokens created by the RADIOv2.5-B model would require a transformer with many more trainable parameters.

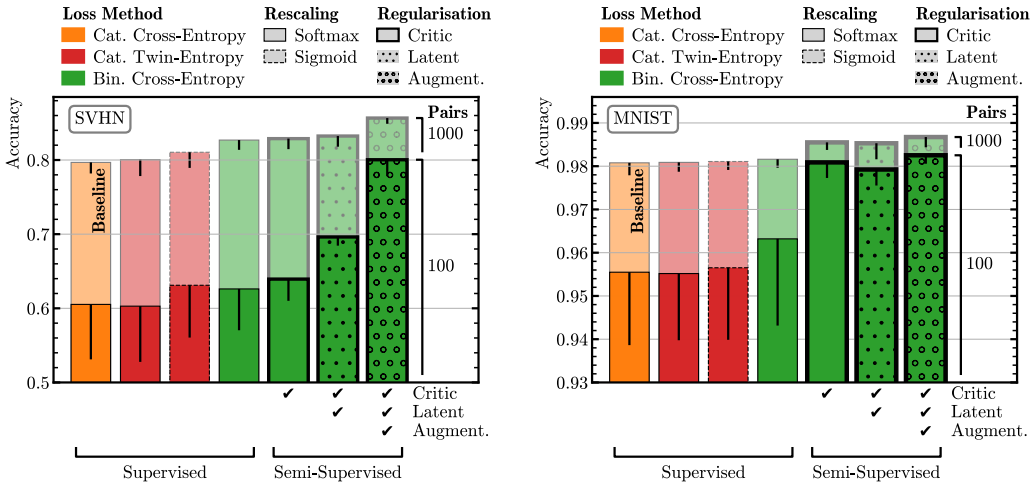

Figure 2: Ablation study of the classification accuracy for the SVHN (left) and MNIST (right) datasets. The baseline training uses categorical cross-entropy loss $\mathcal{L}^{\text{cat-cross}}$ only with the softmax rescaling. For SVHN, the number of unlabelled samples is 73'257 and the weights are set to $\lambda_C = 0.001$, $\lambda_L = 0.1$ and $\lambda_A = 0.1$. For MNIST, the number of unlabelled samples is 60'000 and the weights are set to $\lambda_C = 1.0$, $\lambda_L = 0.1$ and $\lambda_A = 0.1$.

augmentation weights or losses, respectively. In the legend of the figures, they are simply abbreviated as *critic*, *latent* and *augment*. It is important to note that we always train the entire model on the total loss. No individual scheduler of the different weights is used, which means that every term of Eq. (17) is activated all the time, except when its respective weight is equal to 0.

4.4 Results

We present an ablation study of our framework in Fig. 2 for the SVHN and MNIST datasets. The results for the CIFAR10 dataset are available in Appendix D. The figures show the evolution of the classification accuracy, estimated on the test set, as each term of Eq. (17) is activated one by one. Both cases of 100 and 1000 labelled images are shown, denoted by the number of pairs in the dataset. We also test the difference between the softmax and sigmoid rescaling functions of Eqs. (8) and (9), respectively. The *baseline* represents the standard method typically used to fine-tune foundation models in a supervised manner. The other bars show how performance evolves when our semi-supervised framework is fully leveraged. In each case, the λ_C, λ_L and λ_A weights are fixed to their overall best performing values. Moreover, each setup is trained three times with three different seeds.[7] The best seed is selected and the range of accuracy achieved is shown as a vertical line on each bar, highlighting the variability of the setup. Exhaustive results of the whole sweep over the weights are shown in Appendix D.

The results obtained for SVHN and shown in Fig. 2 (left) confirm that, as expected, the performance of the model is greatly improved when more pairs are available for the supervised losses. This is reflected in a 20% increase in accuracy in the baseline case. Intriguingly, switching to the categorical twin-entropy loss does not change the results much if we stick to the softmax rescaling. However, also switching to the sigmoid rescaling adds 1% to 3% accuracy, supporting our claim that the tighter bound of Eq. (3) has a positive impact on the convergence of the model.We believe that relaxing the constraints on the sum of the predictor outputs allows the network to be more expressive, so that the contrastive-like denominator of $\mathcal{L}^{\text{cat-twin}}$ is able to guide the training towards a better local minimum of the loss. We observe another 2% boost in performance in the case of 1000 pairs when we switch to the more stable binary cross-entropy alternative to our contrastive-like loss.[8] These results confirms both that supervised contrastive training leads to a gain in accuracy and that $\mathcal{L}^{\text{bin-cross}}$ is indeed more stable than $\mathcal{L}^{\text{cat-twin}}$ in its current implementation. The results discussed so far only employed supervised loss in the downstream task space. Adding the unsupervised critic loss does

[7] Seeds are 42, 1337 and 3435.

[8] At this point, sigmoid rescaling shows a decrease in performance, requiring deeper investigation.

slightly improve the accuracy on the SVHN dataset, but more interestingly decreases the variability over seeds. In other words, it stabilises the training. This behaviour, also studied in [1], shows that an unsupervised regularisation of the output space, exploiting the large amount of unlabelled data, is necessary in the situation of low-labelled regimes. A further 6% improvement, for the case of 100 pairs, is due to the activation of the supervised latent alignment loss. Having a more consistent latent space with respect to the different classes presumably favours a better prediction. Finally, including unlabelled data in the latent space alignment loss by creating pairs via augmentations results in a significant improvement in performance. The increase in accuracy is approximately 3% and 10% for the cases of 1000 and 100 pairs, respectively, which greatly reduces the gap between them. This last point is particularly worth emphasising: by fully exploiting the unlabelled data and all the consistent loss terms derived in our framework, the model performs nearly as well with 100 and 1000 pairs. Another important observation is that variability across seeds is reduced as soon as semi-supervised training is activated, with this effect becoming even more pronounced when the latent space is regularised.

Qualitatively, similar results are observed for the MNIST dataset in Fig. 2 (right). However, the overall accuracy is much higher in this case due to the much simpler nature of the dataset and probably because these images have already been seen by the backbone foundation model during its pre-training phase. Some notable differences are worth mentioning. We observe that activating the critic loss is highly effective, almost bridging the gap between the cases of 100 and 1000 pairs. We believe that, due to the aforementioned simplicity of the dataset and the high accuracy achieved, the impact of supervised classification loss may be reduced. Therefore, we expect the additional unsupervised constraints to have a more visible effect. It then seems that the performance reaches a plateau, as the accuracy is only marginally impacted by both supervised and self-supervised latent alignment losses.

Overall, we find that the performance of the classifier is highly dependent on the dataset, and most likely on what has already been seen during the pre-training of the foundation model. We can assume that samples from MNIST and CIFAR10, two very common datasets in the machine learning community, were used to train the foundation model. Therefore, our hypothesis is that the goal of the fine-tuning strategy is to retrieve these samples from the model's memory so that it starts working properly for classification. This study shows that mutual information maximisation, and in particular the lower bound derived in Eq. (3), works better towards this goal than the naive supervised categorical cross-entropy alone, especially for low-labelled data regimes.

5 Conclusion

We demonstrate that our mutual information maximisation framework is able to enhance the fine-tuning of foundation model-based classifiers thanks to three crucial aspects. Firstly, although it is possible to form a lower bound that includes only the conditional density as in Eq. (2), it is preferable to decompose it further and let the Kullback-Leibler divergence appear as in Eq. (3) in situations where a prior on the marginal density is known. Note that in such cases this divergence term should not be omitted in order for the expression to be a proper lower bound. Secondly, the appearance of the Kullback-Leibler divergence is closely related to the emergence of the contrastive-like denominator in Eq. (3). Preventing the predictor outputs from collapsing, this term allows the use of the more flexible sigmoid rescaling. It also leads to an improvement in performance by leveraging the relationships between each pair of samples in a batch of data. Thirdly, we observe that replacing our categorical twin-entropy loss with a binary cross-entropy stable alternative, which plays an analogous role, gives a new boost to the fine-tuning of the model. Finally, the supervised and self-supervised latent space alignment losses, also expressed as lower bounds on the mutual information, bring a further increase in accuracy. Indeed, the more the latent representations of images from the same class coincide, the better the predictor can rely on them to classify the images. The results on MNIST, CIFAR10 and SVHN undeniably support the valorisation of the available unlabelled data.

All of these improvements are even more pronounced when we have only a few labelled samples at our disposal. This can be particularly relevant in various situations, such as in scientific domains, where many observations are made, but only a hand-picked selection is accurately labelled due to time constraints, noisy data, unknown variables, etc. Our framework would fit well in such situations. In addition, while the focus of this study is on vision models, the proposed methods hold promise for extension to multimodal models, such as vision-language models, paving the way for broader

applications in semi-supervised learning. As a final thought, our results suggest that foundation models should be trained on as many datasets as possible in order to enable efficient fine-tuning. Moreover, only a proper fine-tuning method would benefit from this and fully exploit the memory retrieval capability of such a foundation model.

References

[1] Slava Voloshynovskiy, Olga Taran, Mouad Kondah, Taras Holotyak, and Danilo Rezende. Variational Information Bottleneck for Semi-Supervised Classification. *Entropy*, 22(9):943, 2020. doi: 10.3390/e22090943.

[2] Guillaume Quétant, Yury Belousov, Vitaliy Kinakh, and Slava Voloshynovskiy. TURBO: The Swiss Knife of Auto-Encoders. *Entropy*, 25(10):1471, 2023. doi: 10.3390/e25101471.

[3] Mariia Drozdova, Vitaliy Kinakh, Yury Belousov, Erica Lastufka, and Slava Voloshynovskiy. Semi-Supervised Fine-Tuning of Vision Foundation Models with Content-Style Decomposition, 2024. URL `https://arxiv.org/abs/2410.02069`.

[4] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding, 2018. URL `https://arxiv.org/abs/1807.03748`.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020. URL `https://arxiv.org/abs/2002.05709`.

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9650–9660, 10 2021.

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision, 2021. URL `https://arxiv.org/abs/2103.00020`.

[8] Dong-Hyun Lee. Pseudo-Label: The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. *Workshop on challenges in representation learning, ICML*, 2013.

[9] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization With Stochastic Transformations and Perturbations for Deep Semi-Supervised Learning. *Conference on Neural Information Processing Systems*, 2016.

[10] Samuli Laine and Timo Aila. Temporal Ensembling for Semi-Supervised Learning. *International Conference on Learning Representations*, 2017.

[11] Yves Grandvalet and Yoshua Bengio. Semi-supervised Learning by Entropy Minimization. *Conference on Neural Information Processing Systems*, 2005.

[12] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Ekin Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. *Conference on Neural Information Processing Systems*, 2020.

[13] Bowen Zhang, Yidong Wang, Wenxin Hou, Qizhao Wu, Hao Wu, Zihan Zhu, Xuelong Li, and Guo-Jun Qi. FlexMatch: Boosting Semi-Supervised Learning with Curriculum Pseudo Labeling. *Conference on Neural Information Processing Systems*, 2021.

[14] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models, 2021. URL `https://arxiv.org/abs/2106.09685`.

[15] Dawid J. Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. VeRA: Vector-based Random Matrix Adaptation, 2023. URL `https://arxiv.org/abs/2310.11454`.

[16] Yulong Mao, Kaiyu Huang, Changhao Guan, Ganglin Bao, Fengran Mo, and Jinan Xu. DoRA: Enhancing Parameter-Efficient Fine-Tuning with Dynamic Rank Distribution, 2024. URL https://arxiv.org/abs/2405.17357.

[17] Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and Iryna Gurevych. Compacter: Efficient Low-Rank Hypercomplex Adapter Layers. *Conference on Neural Information Processing Systems*, 2021.

[18] Brian Lester, Rami Al-Rfou, and Noah Constant. The Power of Scale for Parameter-Efficient Prompt Tuning. *Conference on Empirical Methods in Natural Language Processing*, 2021.

[19] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method, 2000. URL https://arxiv.org/abs/physics/0004057.

[20] David Barber and Felix Agakov. The IM Algorithm: A variational approach to Information Maximization. *Advances in neural information processing systems*, 16(320):201, 2004.

[21] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep Variational Information Bottleneck, 2016. URL https://arxiv.org/abs/1612.00410.

[22] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On Variational Bounds of Mutual Information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019.

[23] Michael Tschannen, Josip Djolonga, Paul K. Rubenstein, Sylvain Gelly, and Mario Lucic. On Mutual Information Maximization for Representation Learning, 2019. URL https://arxiv.org/abs/1907.13625.

[24] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization, 2018. URL https://arxiv.org/abs/1808.06670.

[25] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

[26] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, 2009. URL https://www.cs.toronto.edu/~kriz/cifar.html.

[27] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading Digits in Natural Images with Unsupervised Feature Learning. In *Workshop on deep learning and unsupervised feature learning, NIPS*, volume 2011, page 4. Granada, 2011.

[28] Mike Ranzinger, Greg Heinrich, Jan Kautz, and Pavlo Molchanov. AM-RADIO: Agglomerative Vision Foundation Model Reduce All Domains Into One. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12490–12500, 6 2024.

[29] Mike Ranzinger, Jon Barker, Greg Heinrich, Pavlo Molchanov, Bryan Catanzaro, and Andrew Tao. PHI-S: Distribution Balancing for Label-Free Multi-Teacher Distillation, 2024. URL https://arxiv.org/abs/2410.01680.

[30] Greg Heinrich, Mike Ranzinger, Hongxu, Yin, Yao Lu, Jan Kautz, Andrew Tao, Bryan Catanzaro, and Pavlo Molchanov. RADIO Amplified: Improved Baselines for Agglomerative Vision Foundation Models, 2024. URL https://arxiv.org/abs/2412.07679.

[31] Cijo Jose, Théo Moutakanni, Dahyun Kang, Federico Baldassarre, Timothée Darcet, Hu Xu, Daniel Li, Marc Szafraniec, Michaël Ramamonjisoa, Maxime Oquab, Oriane Siméoni, Huy V. Vo, Patrick Labatut, and Piotr Bojanowski. DINOv2 Meets Text: A Unified Framework for Image- and Pixel-Level Vision-Language Alignment, 2024. URL https://arxiv.org/abs/2412.16334.

A Log-sum-exp expansion

The numerical instability of the softmax and sigmoid functions is a well-known problem. Indeed, they can quickly underflow to zero or overflow to infinity, destroying the ability to train a network. Fortunately, in many cases only the logarithm of these functions is present in the loss, and simple tricks can be used to stabilise the training.

To simplify the notation, let us define the *softmax* function as

$$\sigma(\mathbf{x})_i = \frac{e^{\mathbf{x}_i}}{\sum_i e^{\mathbf{x}_i}}, \tag{18}$$

for a vector $\mathbf{x}$. The numerator $e^{\mathbf{x}_i}$ can easily overflow or underflow due to the rapid evolution of the exponential function and the limited precision of a computer. To solve the overflow issue, we can shift the exponent by computing equivalently

$$\sigma(\mathbf{x})_i = \frac{e^{\mathbf{x}_i - \max(\mathbf{x})}}{\sum_i e^{\mathbf{x}_i - \max(\mathbf{x})}}, \tag{19}$$

so that the exponential is always between 0 and 1. This also ensures that the denominator is never 0, since at least one term of the sum is $e^0 = 1$. To further solve the underflow issue when computing the logarithm of the softmax, we simply need to rewrite the formula as

$$\log \sigma(\mathbf{x})_i = \mathbf{x}_i - \max(\mathbf{x}) - \log \sum_i e^{\mathbf{x}_i - \max(\mathbf{x})}, \tag{20}$$

The stabilisation of the *sigmoid* function follows another simple trick. Let us define it again as

$$\sigma(\mathbf{x})_i = \frac{e^{\mathbf{x}_i}}{1 + e^{\mathbf{x}_i}}. \tag{21}$$

Taking the logarithm and rewriting the expression as

$$\begin{aligned} \log \sigma(\mathbf{x})_i &= \log \frac{e^{\mathbf{x}_i}}{1 + e^{\mathbf{x}_i}} \\ &= \log \frac{1}{1 + e^{-\mathbf{x}_i}} \\ &= -\log(1 + e^{-\mathbf{x}_i}), \end{aligned} \tag{22}$$

solves the overflow issue. To solve the underflow issue, we switch to the linear function when $\mathbf{x}_i \ll 0$, namely when $-\log(1 + e^{-\mathbf{x}_i}) \approx -\log(e^{-\mathbf{x}_i}) = \mathbf{x}_i$.

B Architecture details

For the common networks, our model follows the architecture described in [1; 3]. We highlight the relevant features of the RADIOv2.5-B model in Table 1 and summarise all the details of the transformer, projector, predictor and discriminator networks in Tables 2 to 5, respectively. When applied, augmentations involve resizing, cropping, horizontal flip (if not altering the semantic content of the image), colour jitter, grey scale, Gaussian blur and solarisation.[9] The models are trained with the AdamW optimiser for 5 epochs (around 500-700 steps) with a batch size of 512, a learning rate of $5 \cdot 10^{-5}$, and a 150-step linear warm-up scheduler with an initial factor of 0.001. All models are trained on internal clusters of GPUs including NVIDIA A100, L40, GeForce RTX 3090 and RTX A5000/A5500 for approximately 1 to 2 hours each.

B.1 Backbone

The RADIOv2.5-B foundation model takes an image as input. It outputs tokens corresponding to multiple patches of the image and a CLS token containing the global information about the image. A few of its relevant features are highlighted in Table 1.

[9] Parameters follow `https://github.com/facebookresearch/dinov2`.

Table 1: Architecture highlights of the RADIOv2.5-B model.

Input	Output	Layer
$28 \times 28 \mid 32 \times 32$	224×244	Bilinear interpolation
		⋮
/	2304	CLS token
/	768	Patch tokens

B.2 Transformer

The transformer takes the backbone patch tokens as input. It applies a single layer with both a single-headed self-attention block and a SwiGLU network to allow further cross-interaction of the tokens. Its architecture is detailed in Table 2.

Table 2: Architecture details of the transformer network.

Input	Output	Layer
768	768	Single-Headed Self-Attention
		SwiGLU network
768	768 + 768	Linear (chunks: 2)
768	768	SiLU (on chunk 1)
768 + 768	768	Mult(chunk 1, chunk 2)
768	768	Linear

B.3 Projector

The projector takes the concatenation of the backbone CLS token and the average pooling of the transformer patch tokens as input. It applies a dense network with dropout to project these features to a latent space. Its architecture is detailed in Table 3.

Table 3: Architecture details of the projector network.

Input	Output	Layer
2304 + 768	8000	Linear
8000	8000	LeakyReLU (slope: 0.01)
8000	8000	Dropout (probability: 0.3)

B.4 Predictor

The predictor takes the projector latent representation as input. It applies a dense network to classify the initial image by means of a one-hot output vector. Its architecture is detailed in Table 4.

Table 4: Architecture details of the predictor network.

Input	Output	Layer
8000	1024	Linear
1024	1024	LeakyReLU (slope: 0.01)
1024	10	Linear

B.5 Discriminator

The discriminator takes the predictor one-hot vector as input. It applies a dense network to predict whether this vector is consistent with a categorical distribution or not. Its architecture is detailed in Table 5.

Table 5: Architecture details of the discriminator network.

Input	Output	Layer
10	500	Linear
500	500	ReLU
500	500	Linear
500	500	ReLU
500	1	Linear

C Losses

The TwinTURBO formalism is used in both the latent space $\mathbf{Z}$ and downstream task space $\mathbf{Y}$ of our setup. We translate here the lower bounds on mutual information presented in Section 3 into losses to be minimised in each of these spaces.

Supervised classification. In the downstream task space, the losses are derived from the mutual information $I(\mathbf{X};\mathbf{Y})$ between the random variables $\mathbf{X}$ and $\mathbf{Y}$. We use three variants of the the cross-entropy derived in Eqs. (10) and (12). Considering a batch of N independent samples $\{(\mathbf{x}_i,\mathbf{y}_i)\}_{i=1}^N$, the baseline categorical cross-entropy loss corresponds to the numerator of the first term of Eq. (10)

$$\mathcal{L}^{\text{cat-cross}} = -\frac{1}{N}\sum_{i=1}^{N}\log\sigma_{\mathbf{y}_i}(h_\psi(f_\phi((\mathbf{x}_i)))), \tag{23}$$

the categorical *twin-entropy* loss corresponds to the entire first term

$$\mathcal{L}^{\text{cat-twin}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\sigma_{\mathbf{y}_i}(h_\psi(f_\phi((\mathbf{x}_i))))}{\frac{1}{N}\sum_{j=1}^{N}\sigma_{\mathbf{y}_i}(h_\psi(f_\phi(\mathbf{x}'_j)))}, \tag{24}$$

and the binary cross-entropy loss corresponds to the first two terms of Eq. (12)

$$\begin{aligned}\mathcal{L}^{\text{bin-cross}} = &-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}\mathbf{y}_{i,c}\log\sigma(h_\psi(f_\phi(\mathbf{x}_i)))_c \\ &-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}(1-\mathbf{y}_{i,c})\log(1-\sigma(h_\psi(f_\phi(\mathbf{x}_i)))_c),\end{aligned} \tag{25}$$

where the sums over i and j are the empirical approximations to the true expectations over the batch of N samples. These losses use only the labelled data and force the network to predict the correct class $\mathbf{y}$ for the input image $\mathbf{x}$.

Unsupervised distribution covering. In order for the three expressions above to form proper bounds, we must not omit the Kullback-Leibler divergence term derived earlier. As already stated, we use the density ratio trick in each case and compute

$$\mathcal{L}^{\text{critic}} = \frac{1}{N}\sum_{i=1}^{N}\log(D_\alpha(\mathbf{y}_i)) + \frac{1}{N}\sum_{i=1}^{N}\log(1-D_\alpha(h_\psi(f_\phi(\mathbf{x}_i))))]. \tag{26}$$

This loss ensures that the $p_\theta(\mathbf{y})$ distribution of the model's output is consistent with the categorical distribution given by $p(\mathbf{y})$. The f_ϕ and h_ψ networks are trained by minimising $\mathcal{L}^{\text{critic}}$, while the D_α discriminator is trained to maximise it. To calculate this loss, the entire dataset is used, without labels. Our discriminator architecture follows [1; 3]. More details are available in Appendix B.

Supervised and self-supervised latent alignment. In addition to the downstream task space, we can also improve the representation given by the backbone foundation model and processed by the transformer and projector. To do this, we apply the InfoNCE bound of Eq. (14) on the mutual information $I(\mathbf{X};\mathbf{Z}^*)$ between the $\mathbf{X}$ and $\mathbf{Z}^*$ spaces and obtain the loss

$$\mathcal{L}^{\text{latent/augment.}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s_\phi(\mathbf{x}_i,\tilde{\mathbf{z}}_i^*)}}{\frac{1}{N}\sum_{j=1}^{N}e^{s_\phi(\mathbf{x}_i,\tilde{\mathbf{z}}_j^*)}}, \tag{27}$$

where the score function $s_\phi(\mathbf{x}, \tilde{\mathbf{z}}^*) = \frac{\tilde{\mathbf{z}}^* \cdot f_\phi(\mathbf{x})}{\|\tilde{\mathbf{z}}^*\| \|f_\phi(\mathbf{x})\|}$ is chosen to be the cosine similarity. A crucial aspect here is that the usual *positive* and *negative* pairs can be created in two different ways. On the one hand, the common self-supervised creation of pairs is done via augmentations of the input images, passed through the network to create a new latent representation $\tilde{\mathbf{z}}^* = f_\phi(\mathbf{x}^*)$. The entire unlabelled training dataset can be used in this case and the loss is denoted as $\mathcal{L}^{\text{augment.}}$. On the other hand, the supervised pairs are not created by artificially augmenting the input image, but rather by sampling another image from the same or a different class, respectively. This is possible because of the labelled nature of the data used in this space through the $\mathcal{L}^{\text{latent}}$ loss. In both cases, the variable $\tilde{\mathbf{z}}^* = f_\phi(\mathbf{x}^*)$ is assumed to be the ground truth to which the encoder output $\tilde{\mathbf{z}} = f_\phi(\mathbf{x})$ must be aligned. Note that the same network is used to generated both latent variables, without any update of its parameters. More details are available in Appendix B.

D Additional plots

The results presented in Figs. 3 to 6 show the accuracy of the models trained for every combinations of weights, except for the augmentation weight, which is always set to $\lambda_A = 0$.

D.1 Ablation plot for CIFAR10

Fig. 3 shows the ablation study performed with the CIFAR10 dataset. The results qualitatively support the claims made in Section 4 and the observations presented in Fig. 2.

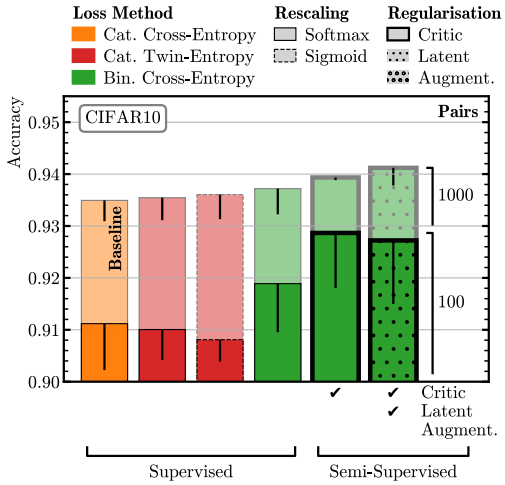

Figure 3: Caption

D.2 All results

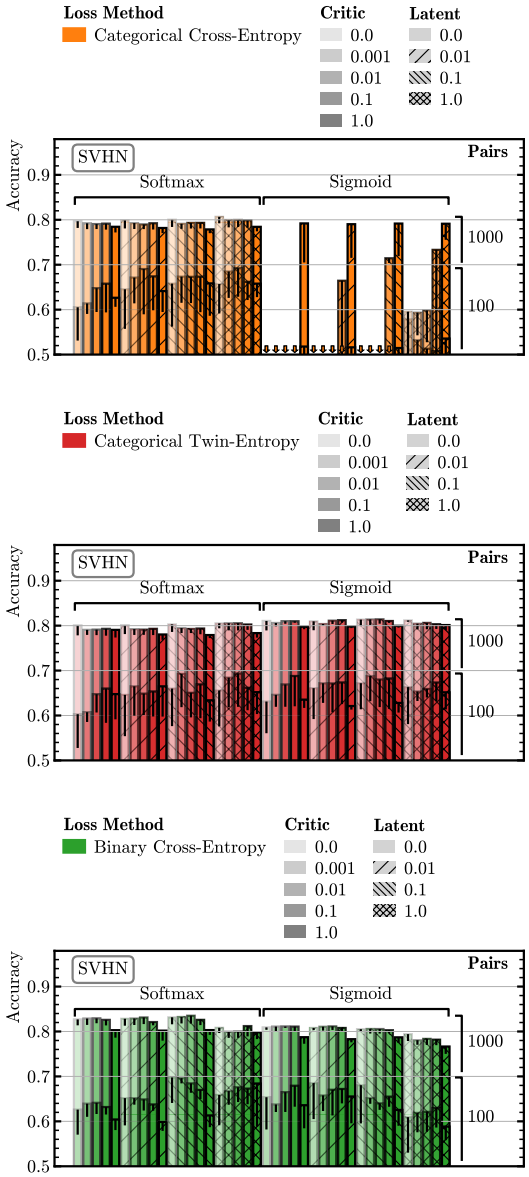

Figure 4: All results for the SVHN dataset.

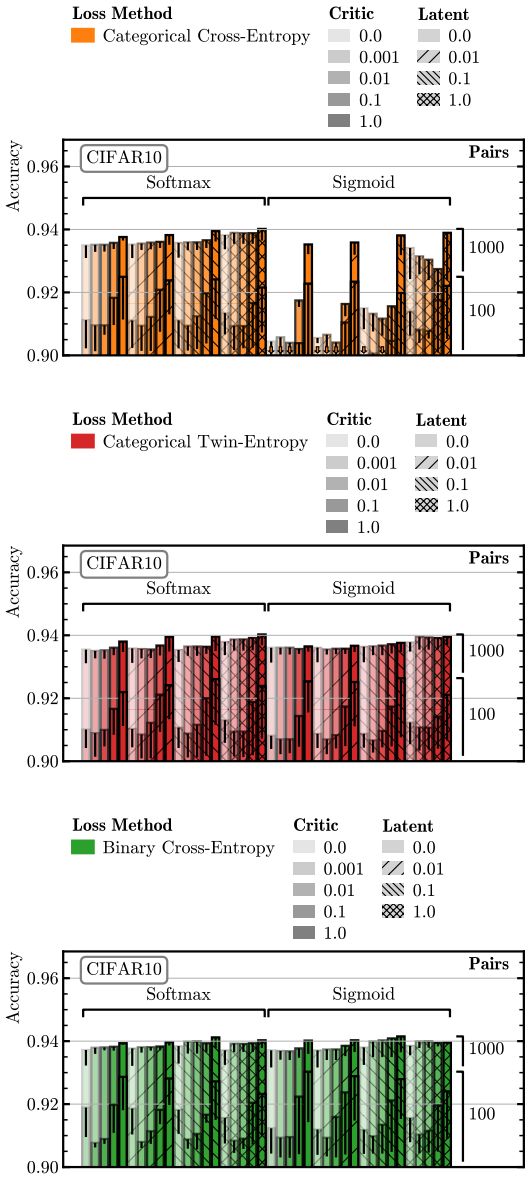

Figure 5: All results for the CIFAR10 dataset.

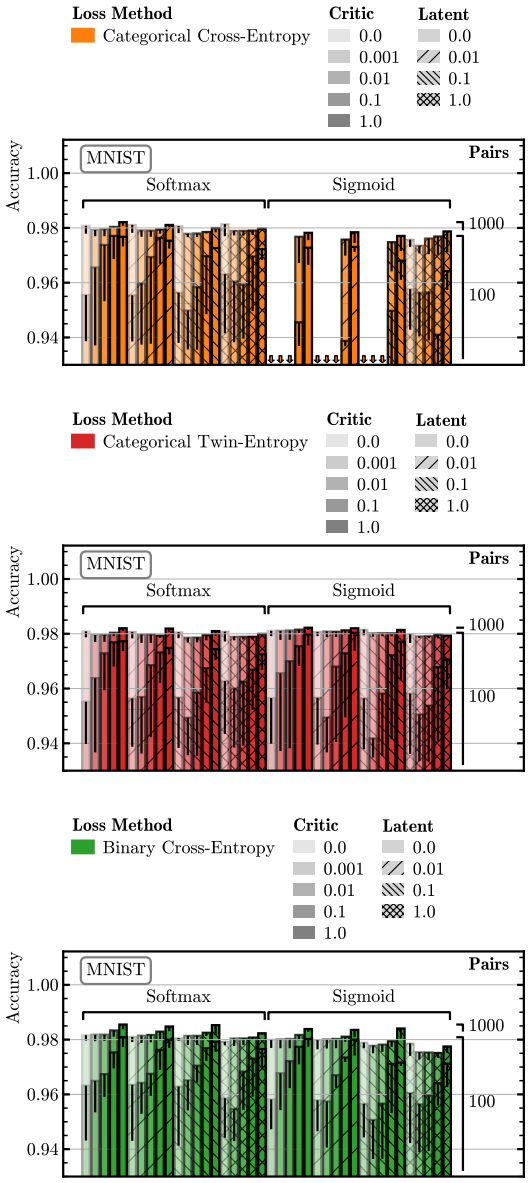

Figure 6: All results for the MNIST dataset.