Towards interpretable-by-design deep learning algorithms

Anonymous authors
Paper under double-blind review

Abstract

Most of the existing deep learning (DL) methods rely on parametric tuning and lack explainability. The few methods that claim to offer explainable DL solutions, such as ProtoPNet and xDNN, do require end-to-end training and finetuning. The proposed framework named IDEAL (Interpretable-by-design DEep learning ALgorithms) recasts the standard supervised classification problem into a function of similarity to a set of prototypes derived from the training data, while taking advantage of existing latent spaces of large neural networks forming so-called Foundation Models (FM). This decomposes the overall problem into two inherently connected stages: A) feature extraction (FE), which maps the raw features of the real world problem into a latent space, and B) identifying representative prototypes and decision making based on similarity and association between the query and the prototypes. This addresses the issue of explainability (stage B) while retaining the benefits from the tremendous achievements offered by DL models (e.g., visual transformers, ViT) pre-trained on huge data sets such as IG-3.6B + ImageNet-1K or LVD-142M (stage A). We show that one can turn such DL models into conceptually simpler, explainable-through-prototypes ones. The key findings can be summarized as follows: (1) the proposed models are interpretable through prototypes, mitigating the issue of confounded interpretations, (2) the proposed IDEAL framework circumvents the issue of catastrophic forgetting allowing efficient class-incremental learning, and (3) the proposed IDEAL approach demonstrates that ViT architectures narrow the gap between finetuned and non-finetuned models allowing for transfer learning in a fraction of time **without** finetuning of the feature space on a target dataset with iterative supervised methods. Furthermore, we show that the proposed approach **without** finetuning improves the performance on confounded data over finetuned counterparts avoiding overfitting. On a range of datasets (CIFAR-10, CIFAR-100, CalTech101, STL-10, Oxford-IIIT Pet, EuroSAT), we demonstrate, through an extensive set of experiments, how the choice of the latent space, prototype selection, and finetuning of the latent space affect the performance. Building upon this knowledge, we demonstrate that the proposed models have an edge over state-of-the-art baselines in class-incremental learning. Finally, we analyse the interpretations provided by the proposed IDEAL framework, as well as the impact of confounding on the interpretations.

1 Background

s

Deep-learning (DL) models can be formulated as deeply embedded functions of functions (Angelov & Gu (2019), Rosenblatt et al. (1962)):

$$\hat{y}(\mathbf{x}) = f_n(\ldots(f_1(\mathbf{x}|\boldsymbol{\theta}_1)\ldots)|\boldsymbol{\theta}_n), \tag{1}$$

where $f_n(\ldots(f_1(\mathbf{x}|\boldsymbol{\theta}_1)\ldots)|\boldsymbol{\theta}_n)$ is a layered function of the input $\mathbf{x}$, which has a generic enough, fixed parameterisation $\boldsymbol{\theta}$. to predict desirable outputs $\hat{y}$.

However, this problem statement has the following limitations:

(1) transfer learning typically requires finetuning (Kornblith et al. (2019)) using error back-propagation (EBP) on the target, "downstream" problem/data of interest

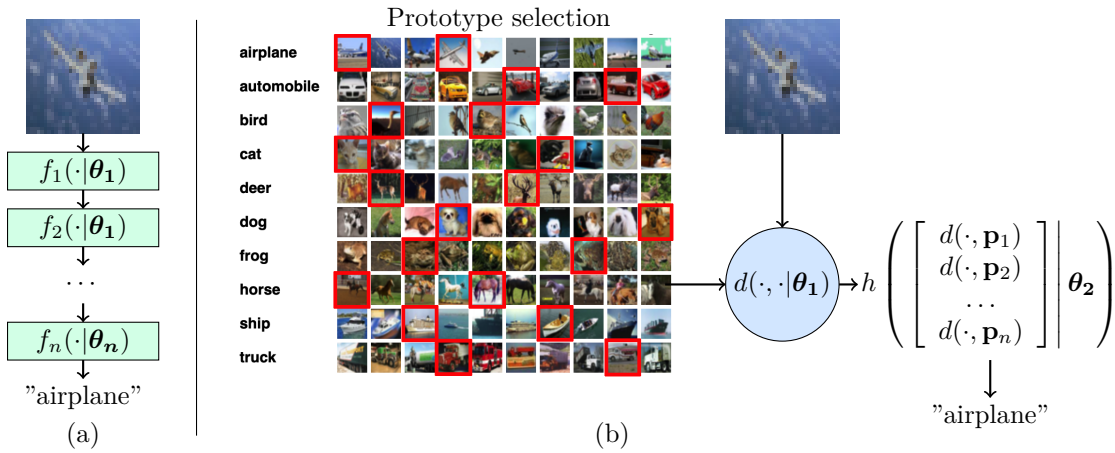


Figure 1: Difference between (a) a standard deep-learning model, and (b) the proposed prototype-based approach, IDEAL; the example is shown for CIFAR-10 dataset (Krizhevsky & Hinton (2009))

(2) such formulation does not depend upon training data, so the contribution of these samples towards the output $\hat{y}$ is unclear, which hinders interpretability; for the interpretable architectures, such as ProtoPNet (Chen et al. (2019)), finetuning leads to confounding interpretations (Bontempelli et al. (2022))

(3) finally, for lifelong learning problems, it creates obstacles such as catastrophic forgetting (Parisi et al. (2019))

We follow an alternative solution centered around prototypes inspired by xDNN (Angelov & Soares (2020)), which, at its core, is using a different formulation:

$$\hat{y} = g(\mathbf{x}|\boldsymbol{\theta}, \mathbb{P}), \tag{2}$$

where $\mathbb{P}$ is a set of prototypes. In fact, we consider a more restricted version of function $g(\cdot)$:

$$\hat{y} = g(\mathbf{x}|\boldsymbol{\theta}_{\{d,h\}}, \mathbb{X}) = h(d(\mathbf{x}, \mathbf{p}|\boldsymbol{\theta}_d)|_{\mathbf{p}\in\mathbb{P}}|\boldsymbol{\theta}_h), \tag{3}$$

where d is some form of (dis)similarity function (which can include DL FE), $\boldsymbol{\theta}_d$ and $\boldsymbol{\theta}_h$ are parameterisations of functions d and h.

The idea takes its roots from cognitive science and the way humans learn, namely using examples of previous observations and experiences (Zeithamova et al. (2008)). Prototype-based models have long been used in different learning systems: k nearest neighbours (Radovanovic et al. (2010)); decision trees (Nauta et al. (2021)); rule-based systems (Angelov & Zhou (2008)); case-based reasoning (Kim et al. (2014)); sparse kernel machines (Tipping (1999)). The advantages of prototype-based models has been advocated, for example, in Bien & Tibshirani (2011); the first prototypical architecture, learning both distances and prototypes, was proposed in Snell et al. (2017); more recently, they have been successfully used in Chen et al. (2019); Angelov & Soares (2020) and Wang et al. (2023).

In this paper, we demonstrate the efficiency of the proposed compact, easy to interpret by humans, fast to train and adapt in lifelong learning models that benefit from a latent data space learnt from a generic data set transferred to a different, more specific domain.

This can be summarised through following contributions:

- we propose a conceptually simple yet efficient framework, IDEAL, which transforms a given non-interpretable DL model into an interpretable one based on prototypes, derived from the training set.

- we demonstrate the benefits of the proposed framework on transfer and lifelong learning scenarios: in a fraction of training time, **without finetuning** of latent features, the proposed models achieve performance competitive with standard DL techniques.
- we demonstrate the model's interpretability, on classification and lifelong learning tasks, and show that **without** finetuning, the resulting models achieves **better** performance on confounded CUB data comparing to finetuned counterparts (Wah et al. (2011); Bontempelli et al. (2022))

We apply this generic new IDEAL framework to a set of standard DL architectures such as ViT (Dosovitskiy et al. (2020); Singh et al. (2022)), VGG (Simonyan & Zisserman (2014)), ResNet (He et al. (2016)) and xDNN (Angelov & Soares (2020)) on a range of data sets such as CIFAR-10, CIFAR-100, CalTech101, EuroSAT, Oxford-IIIT Pet, and STL-10.

2 Related work

Explainability The ever more complicated DL models (Krizhevsky et al. (2012); Dosovitskiy et al. (2020)) do not keep pace with the demands for human understandable explainability (Rudin (2019)). The spread of use of complex DL models prompted pursuit of ways to explain such models. Explainability of deep neural networks is especially important in a number of applications in automotive (Kim & Canny (2017)), medical (Ahmad et al. (2018)), Earth observation (Zhang et al. (2022)) problems alongside others. Demand in such models is necessitated by the pursuit of safety (Wei et al. (2022)), as well as ethical concerns (Peters (2022)). Some of the pioneering approaches to explaining deep neural networks involve *post hoc* methods; these include saliency models such as saliency map visualisation method (Simonyan et al. (2014)) as well as Grad-CAM (Selvaraju et al. (2017)). However, saliency-based explanations may be misleading and not represent the causal relationship between the inputs and outputs (Atrey et al. (2019)), representing instead the biases of the model (Adebayo et al. (2018)). An arguably better approach is to construct interpretable-by-design (*ante hoc*) models (Rudin (2019)). These models could use different principles: interpretable-by-design architectures (Böhle et al. (2022)), which are designed to provide interpretations at every step of the architecture, as well as prototype-based models, which perform decision making as a function of (dis)similarity to existing prototypes (Angelov & Soares (2020)). One of the limitations of the prototype based methods is that they are often still based on non-interpretable similarity metrics; this can be considered an orthogonal open problem which can be addressed by providing interpretable-by-design DL architectures (Böhle et al. (2022)).

Symbolic and sparse learning machines The idea of prototype-based machine learning is closely related to the symbolic methods (Newell et al. (1959)), and draws upon the sparse learning machines (Poggio & Girosi (1998)) and case based reasoning (Kim et al. (2014)). The idea of sparse learning machines (Poggio & Girosi (1998)) is to learn a linear (with respect to parameters) model, which is (in general, nonlinearly) dependent on a subset of training data samples (hence, the notion of sparsity). At the centre of many such methods is the kernel trick (Schölkopf et al. (2001)), which involves mapping of training and inference data into a space with different inner product within a reproducing Hilbert space (Aronszajn (1950)). Such models include support vector machines (SVMs) for classification (Boser et al. (1992)) and support vector regression (SVR) models (Smola & Schölkopf (2004)) for regression, as well as relevance vector machines (RVMs), which demonstrated improvements in sparsity (Tipping (2001)).

Prototype-based models (Snell et al. (2017)) proposed to use a single prototype per class in a few-shot learning supervised scenario. Li et al. (2018) proposed prototype-based learning for interpretable case-based reasoning. ProtoPNet (Chen et al. (2019)) extend this idea to classify an image through dissecting it into a number of patches, which are then compared to prototypes for decision making using end-to-end supervised training. xDNN (Angelov & Soares (2020)) considers whole images as prototypes resulting from the data density distribution resulting in possibly multiple prototypes per class in a non-iterative online procedure. It does consider, though finetuned on the "downstream"/target data set model for feature extraction for a better performance owing largely to the fact that weak backbone models such as VGG-16 were used. Versions of xDNN offering prototypes in a form of segments (Soares et al. (2021)) or even pixels (Zhang et al. (2022)) as prototypes were also reported. The concept of xDNN was used in the end-to-end prototype-based learning

method DNC (Wang et al. (2023)). In contrast to xDNN and DNC, we consider the **lifelong learning** scenario and investigate the properties of models, trained on generic and **not finetuned** datasets.

Large deep-learning classifiers In contrast to DNC (Wang et al. (2023)) and ProtoPNet (Chen et al. (2019)), the proposed framework goes beyond the end-to-end learning concept. Instead, it takes advantage of the feature space of large classifiers such as ResNet (He et al. (2016)), VGG (Simonyan & Zisserman (2014)), SWAG-ViT (Singh et al. (2022)), and shows that with carefully selected prototypes one can achieve, on a number of datasets, a performance comparable to end-to-end trained models, in offline and online (lifelong) learning scenarios with or even **without finetuning and end-to-end learning**, thus very fast and computationally efficient, yet interpretable.

Continual learning Continual learning models solve different related problems (van de Ven et al. (2022)). *Task-incremental learning* addresses the problem of incrementally learning known tasks, with the intended task explicitly input into the algorithm (Ruvolo & Eaton (2013); Li & Hoiem (2017); Kirkpatrick et al. (2017)). *Domain-incremental learning* (Wang et al. (2022a); Lamers et al. (2023)) addresses the problem of learning when the domain is changing and the algorithm is not informed about these changes. This includes such issues as *concept drift* when the input data distribution is non-stationary (Widmer & Kubat (1996)). Finally, *class-incremental learning* (Yan et al. (2021); Wang et al. (2022b)) is a problem of ever expanding number of classes of data. In this paper, we only focus on this last problem; however, one can see how the prototype-based approaches could help solve the other two problems by circumventing catastrophic forgetting (French (1999)) through incremental update of the prototypes (Baruah & Angelov (2012)).

Clustering Critically important for enabling continual learning is to break the iterative nature of the end-to-end learning and within the proposed concept which offers to employ clustering to determine prototypes. Therefore, we are using both online (ELM (Baruah & Angelov (2012)), which is an online version of mean-shift (Comaniciu & Meer (2002))) and offline (MacQueen et al. (1967)) methods. Although there are a number of online clustering methods, e.g. the stochastic Chinese restaurant process Bayesian non-parametric approach (Aldous et al. (1983)), they usually require significant amount of time to run and therefore we did not consider those.

3 Methodology

3.1 Problem statement

Two different definitions of the problem statement are considered: offline and online (lifelong) learning. In the experimental section, we discuss the implementations of the framework and the experimental results.

Offline learning Consider the following optimisation problem:

$$\arg\min_{\substack{\mathbb{P}=\mathbb{P}(\mathbb{X}),\\ \boldsymbol{\theta}_{\{d,h\}}}} \sum_{(\mathbf{x},y)\in(\mathbb{X},\mathbb{Y})} l(h(d(\mathbf{x},\mathbf{p}|\boldsymbol{\theta}_d)|_{\mathbf{p}\in\mathbb{P}}|\boldsymbol{\theta}_h),y), \tag{4}$$

where $(\mathbb{X},\mathbb{Y})$ are a tuple of inputs and labels, respectively, and $\mathbb{P}$ is a list of prototypes derived from data $\mathbb{X}$ (e.g., by selecting a set of representative examples or by clustering).

Brute force optimisation for the problem of selecting a set of representative examples is equivalent to finding a solution of the best subset selection problem, which is an NP-hard problem (Natarajan (1995)). While there are methods solving such subset selection problems in limited cases such as sparse linear regression (Bertsimas et al. (2016)), it still remains computationally inefficient in general case (polynomial complexity is claimed in Zhu et al. (2020)) and/or solving it only in a limited (i.e. linear) setting.

The common approach is to replace the original optimisation problem (equation (4)) with a surrogate one, where the prototypes $\mathbb{P}$ are provided by a data distribution (Angelov & Soares (2020)) or a geometric, e.g. clustering (Wang et al. (2023)) technique. Then, once the prototypes are selected, the optimisation problem

becomes:

$$\arg\min_{\boldsymbol{\theta}_{\{d,h\}}} \sum_{(\mathbf{x},y)\in(\mathbb{X},\mathbb{Y})} l(h(d(\mathbf{x},\mathbf{p}|\boldsymbol{\theta}_d)|_{p\in\mathbb{P}}|\boldsymbol{\theta}_h),y). \tag{5}$$

Online (lifelong) learning Instead of solving a single objective for a fixed dataset, the problem is transformed into a series of optimisation problems for progressively growing set $\mathbb{X}$:

$$\{\arg\min_{\boldsymbol{\theta}_{\{d,h\}}} \sum_{(\mathbf{x},y)\in(\mathbb{X}_n,\mathbb{Y}_n)} l(h(d(\mathbf{x},\mathbf{p}|\theta_d)|_{\mathbf{p}\in\mathbb{P}_n}|\boldsymbol{\theta}_h),y)\}_{n=1}^{N}, \mathbb{X}_n = \mathbb{X}_{n-1} + \{\mathbf{x}_n\}, \mathbb{X}_1 = \{\mathbf{x}_1\}. \tag{6}$$

Once the prototypes are found, the problem would only require only light-weight optimisation steps as described in Algorithms 1 and 2.

Data: Training data $\mathbb{X} = \{\mathbf{x}_1 \dots \mathbf{x}_N\}$;
Result: Prototype-based classifier $c(\mathbf{x}|\mathbb{P},\boldsymbol{\theta})$
$P \leftarrow$ FindPrototypes($\{\mathbf{x}_1 \dots \mathbf{x}_N\}$);
$\theta \leftarrow$ SelectParameters($\mathbb{X},\mathbb{Y},\boldsymbol{\theta}$);
$\hat{\mathbb{Y}_T} \leftarrow \{h(d(\mathbf{x},\mathbf{p}|\boldsymbol{\theta}_d)|_{\mathbf{p}\in\mathbb{P}}|\boldsymbol{\theta}_h)\}_{\mathbf{x}\in\mathbb{X}_T}$;

Algorithm 1: Training and testing (offline)

Data: Training data $\mathbb{X} = \{\mathbf{x}_1 \dots \mathbf{x}_N\}$;
Result: Prototype-based classifier $h(d(\mathbf{x},\mathbf{p}|\boldsymbol{\theta}_1)|_{p\in\mathbb{P}}|\boldsymbol{\theta}_2)$
$\mathbb{P} \leftarrow \{\}$;
for $\{\mathbf{x},y\} \in \mathbb{X}$ **do**
  $\hat{y} = h(d(\mathbf{x},\mathbf{p}|\boldsymbol{\theta}_d)|_{\mathbf{p}\in\mathbb{P}}|\boldsymbol{\theta}_h)$;
  $\theta \leftarrow$ UpdateParameters($\mathbb{X},\mathbb{Y},\boldsymbol{\theta}$);
  $\mathbb{P} \leftarrow$ UpdatePrototypes($\mathbb{P},\mathbf{x}$));
end

Algorithm 2: Training and testing (online)

3.2 Prototype selection through clustering

Selection of prototypes through many standard methods of clustering, such as k-means (Steinhaus et al. (1956)), is used by methods such as (Zhang et al. (2022)), DNC (Wang et al. (2023)), however, has one serious limitation: they utilise the averaging of cluster values, so the prototypes $\mathbb{P}$ do not, in general, belong to the original training dataset $\mathbb{X}$. It is still possible, however, to attribute the prediction to the set of the cluster members. This can create, as we show in the experimental section, a trade-off between interpretability and performance (see Section 4.2). The available options are summarised in Figure 2. Standard *black-box* classifiers do not offer interpretability through prototypes. Prototypes, selected through k-means, are non-interpretable by their own account as discussed above; however, it is possible to attribute such similarity to the members of the clusters. Finally, one can select real prototypes as cluster centroids; this way it is possible to attribute the decision to a number of real image prototypes ranked by their similarity to the query image.

4 Experiments

Throughout the experimental scenarios, we contrast three settings (see Figure 3):

- A) Standard DL pipeline involving training on generic data sets as well as finetuning on target/"downstream" task/data - both with iterative error backpropagation
- B) IDEAL **without finetuning**: the proposed prototype-based IDEAL method involving clustering in the latent feature space with subsequent decision making process such as using winner-takes-all analysis or k nearest neighbours as outlined in Algorithms 1 and 2

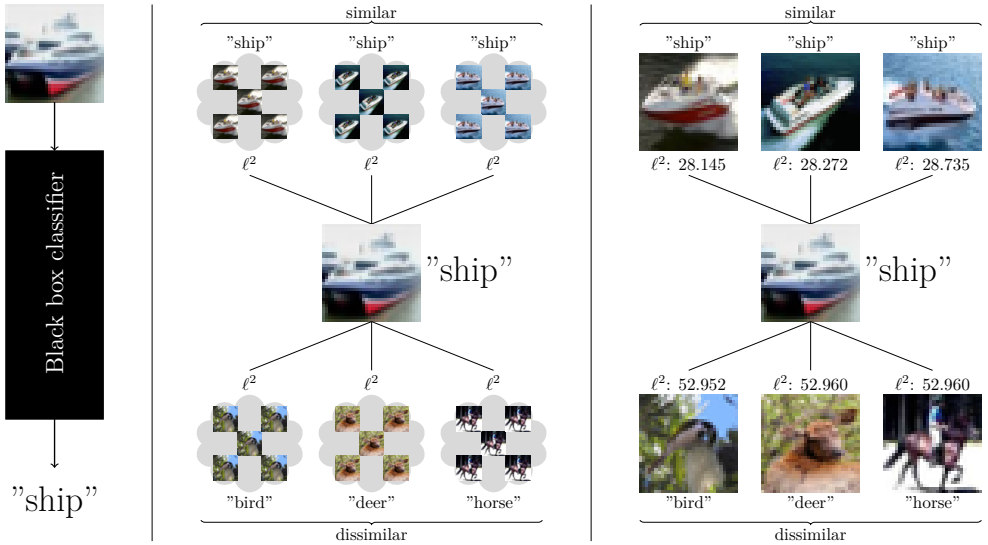


Figure 2: Black-box, k-means centroid prototypes, and interpretable prototypes (CIFAR-10)

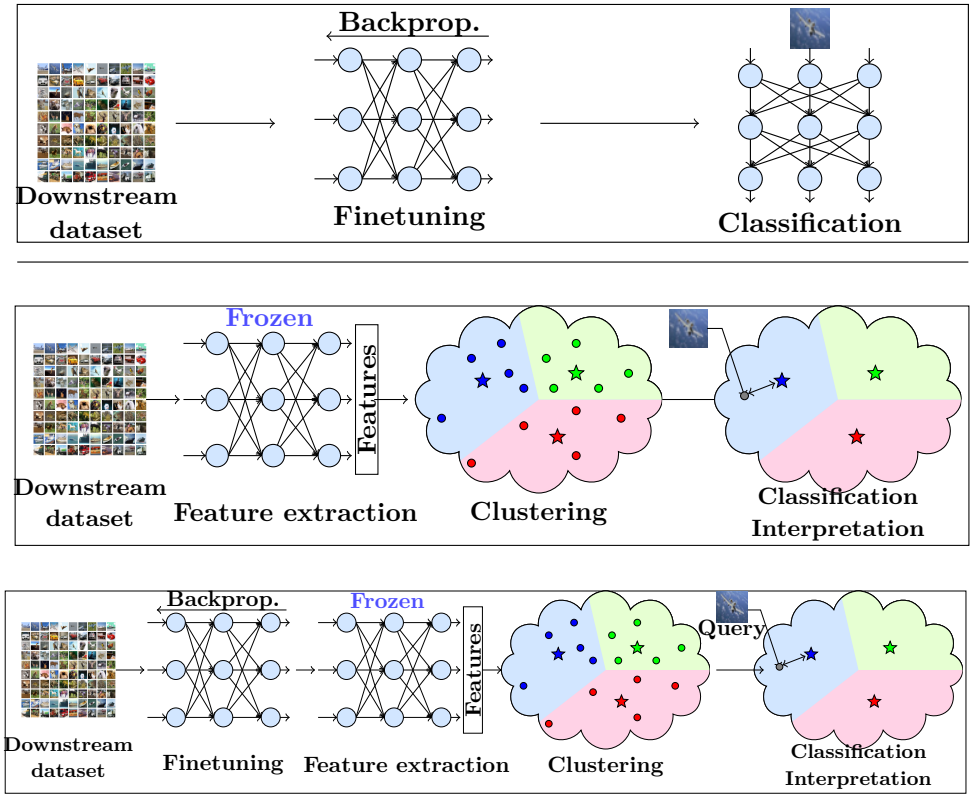


Figure 3: Experimental setup: top: standard DL model; middle: proposed framework with **no** finetuning; bottom: proposed framework **with** finetuning

- C) IDEAL **with finetuning**: Same as B) with the only difference that the clustering is performed in a latent feature space formed by finetuned on target data set (from the "downstream" task) using iterative error backpropagation. The main difference between the settings A) and C) is that setting C) does provide interpretable prototypes unlike setting A)

In an extensive set of experiments, we demonstrate that with state-of-the-art models, such as ViT, the proposed IDEAL framework can provide interpretable results even **without finetuning**, which are competitive and extend to the lifelong learning, and mitigate confounding bias. For reproducibility, the full parameterisation is described in Section A of the Appendix.

The outline of the empirical questions is presented below. Questions 1 and 2 confirm that the method delivers competitive results **even without finetuning**; building upon this initial intuition we develop the key questions 3, 4 and 5, analysing the performance for lifelong learning scenarios and interpretations proposed by IDEAL respectively.

Question 1. *How does the performance of the IDEAL framework **without** finetuning compare with the well-known deep learning frameworks?*

Section 4.2 and Appendix B show, with a concise summary in Figure 4 and Figure 11, that the gaps between finetuned and non-finetuned IDEAL framework are consistently much smaller (tens of percent vs a few percentage points) for vision transformer backbones comparing to ResNets and VGG. Furthermore, Figure 5 shows that the training time expenditure is more than an order of magnitude smaller comparing to the original finetuning.

Question 2. *To what extent does finetuning of the feature space for the target problem lead to overfitting?*

In Section 4.3, figures 7, 8, 9, we demonstrate the issue of overfitting on the target spaces by finetuning on CIFAR-10 and testing on CIFAR-100 in both performance and through visualising the feature space. Interestingly, we also show in Table 3 of the Appendix that, while the choice of prototypes greatly influences the performance of the IDEAL framework **without** finetuning of the backbone, it does not make any significant impact for the finetuned models (i.e., does not improve upon random selection).

Question 3 *How does the IDEAL framework **without** finetuning compare in the class-incremental learning setting?*

In Section 4.4 we build upon questions 1 and 2 and demonstrate: the small gap between pretrained and finetuned ViT models ultimately enables us to solve class-incremental learning scenarios, improving upon well-known baseline methods. IDEAL framework **without** finetuning shows performance results on a number of class-incremental learning problems, comparable to task-level finetuning. Notably, in CIFAR-100 benchmark, the proposed method provides 83.2% and 69.93% on ViT-L and ResNet-101 respectively, while the state-of-the-art method from (Wang et al. (2022b)) only reports 65.86%.

Question 4 *How does the IDEAL framework provide insight and interpretation?*

In Section 4.5, we present the analysis of interpretations provided by the method. In Figure 19 we demonstrate the qualitative experiments showing the human-readable interpretations provided by the model for both lifelong learning and offline scenarios.

Question 5. *Can models **without** finetuning bring advantage over the finetuned ones in terms of accuracy and help identify misclassifications due to confounding (spurious correlations in the input)?*

While, admittedly, the model only approaches but does not reach the same level of accuracy for the same backbone without finetuning in the standard benchmarks such as CIFAR-10, it delivers better performance in cases with confounded data (with spurious correlations in the input). In Section 4.6, Table 1 we demonstrate, building upon the intuition from Question 2, that finetuning leads to overfitting on confounded data, and leads to confounded predictions and interpretations. We also demonstrate that in this setting, IDEAL **without finetuning** improves upon F1 score against the finetuned baseline as well as provides interpretations for wrong predictions due to the confounding.

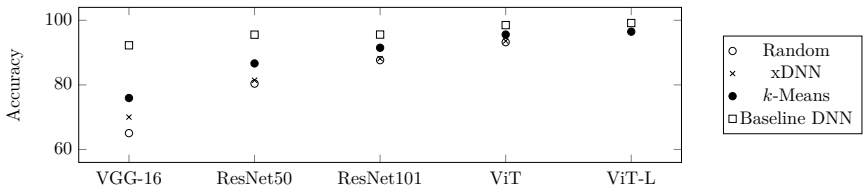


Figure 4: Comparison of the proposed IDEAL framework (**without** finetuning) on the CIFAR-10 data set with different clustering methods (random, the clustering used in xDNN (Soares et al. (2021)) and k-means method) vs the baseline DNN

4.1 Experimental setting

We use the negative Euclidean distance between the feature vectors for our experiments:

$$d(\mathbf{x}, \mathbf{p}|\theta_d) = -\ell^2(\phi(\mathbf{x}|\theta_d), \phi(\mathbf{p}|\theta_d)), \tag{7}$$

where ϕ is the normalised feature extractor output. The similarities bounded between $(0, 1]$ could be obtained by taking the exponential of the similarity function and normalising it.

Except from the experiment in Figure 10, the function h is a winner-takes-all operator:

$$h(\cdot) = \text{CLASS}(\arg\min_{p \in \mathbb{P}} d(\cdot, \mathbf{p}|\theta_d)) \tag{8}$$

Datasets CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton (2009)), STL-10 (Coates et al. (2011)), Oxford-IIIT Pet (Parkhi et al. (2012)), EuroSAT (Helber et al. (2018; 2019)), CalTech101 (Li et al. (2006)).

Feature extractors We consider a number of feature extractor networks such as VGG-16 (Simonyan & Zisserman (2014)), ResNet50 (He et al. (2016)), ResNet101 (He et al. (2016)), ViT-B/16 (Dosovitskiy et al. (2020), referenced further as ViT), ViT-L/16 (Dosovitskiy et al. (2020)) (referenced further as ViT-L) with or without finetuning; the pre-trained latent spaces for ViT models were obtained using SWAG methodology (Singh et al. (2022)); the computations for feature extractors has been conducted using a single V100 GPU.

Clustering techniques We include the results for such clustering techniques as k-means, k-means with a nearest data point (referred to as k-means (nearest)), and two online clustering methods: xDNN (Angelov & Soares (2020)) and ELM (Baruah & Angelov (2012)).

Baselines We explore trade-offs between standard deep neural networks, different architectural choices (averaged prototypes vs real-world examples), and, in Appendix B.

4.2 Offline classification

We found that the gap between the models on a range of tasks decreases for the modern, high performance, architectures, such as ViT (Dosovitskiy et al. (2020)). For CIFAR-10, these findings are highlighted in Figure 4: while finetuned VGG-16's accuracy is close to the one of ViT and other recent models, different prototype selection techniques (the one used in xDNN, k-means clustering, and random selection) all have accuracy between 60 and 80%. The picture is totally different for ViT, where k-means prototype selection **without finetuning** provides accuracy of 95.59% against finetuned ViT's own performance of 98.51%.

While the results above report on performance of the k-means clustering used as a prototype selection technique, the experimental results in Figure 10 explore choosing the nearest prototype to k-means cluster centroid for interpretability reasons. While it is clear (with further evidence presented in Appendix B) that the performance when selecting the nearest to the k-means centroids prototypes is lagging slightly behind the direct use of the centorids (denoted simply as k-means), it is possible to bring this performance closer

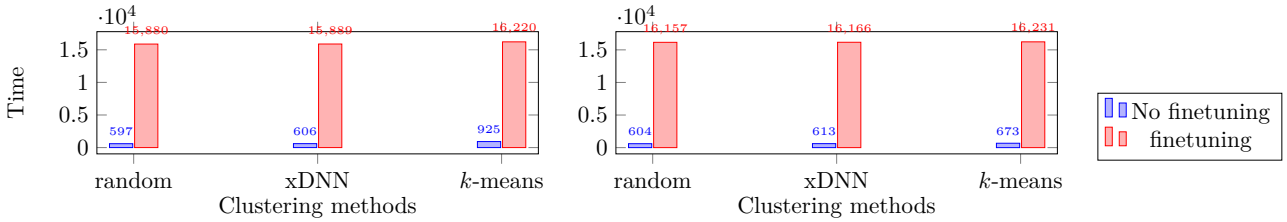


Figure 5: Comparison of training time expenditure on CIFAR-10 (left) and CIFAR-100 (right) with and without funetuning (ViT)

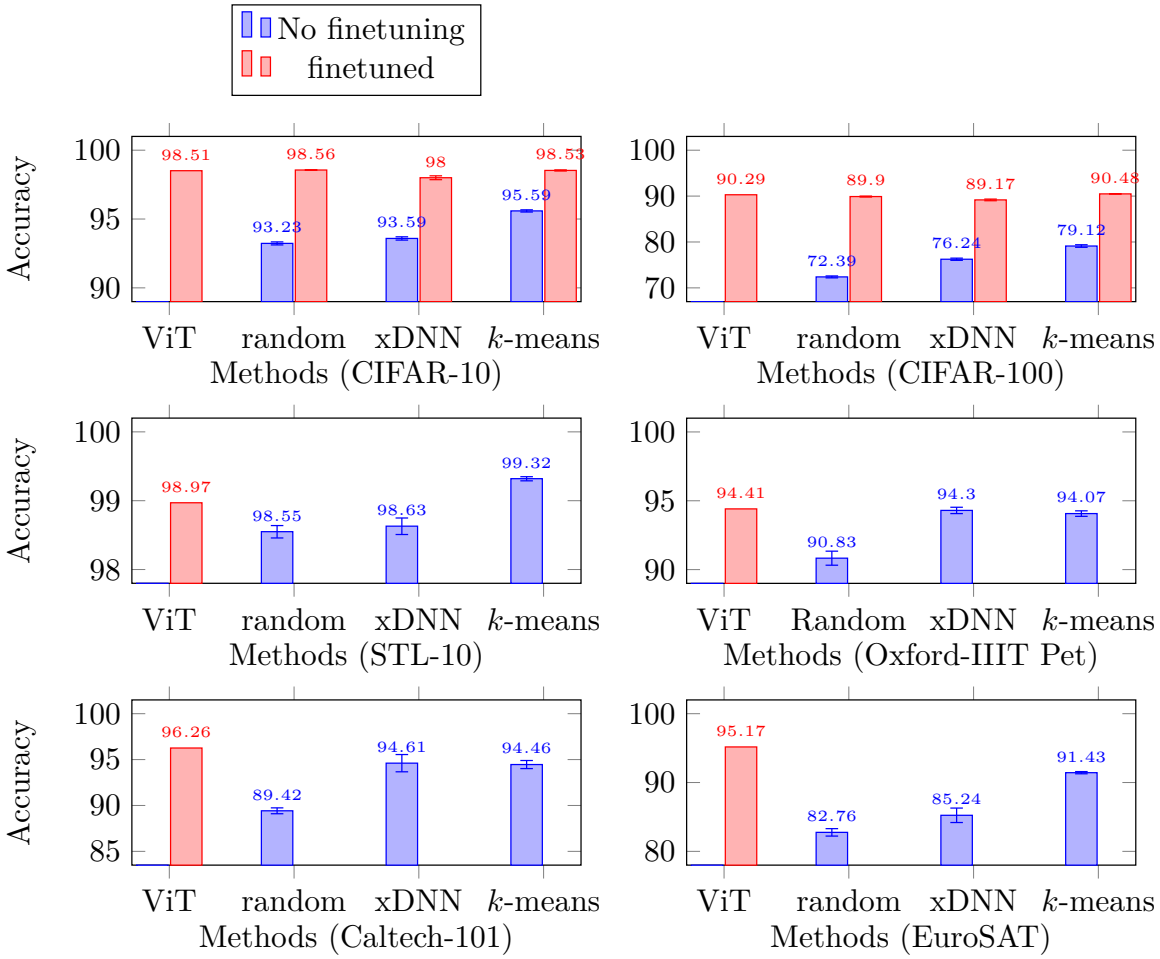


Figure 6: Comparison of results with ViT (Dosovitskiy et al. (2020)) as a feature extractor; {Random,xDNN, k-means}=Proposed ({Random, xDNN, k-means} prototype selection)

by replacing the winner-takes-all decision making approach (Equation (8)) with the k nearest neighbours method. For this purpose, we utilise the sklearn's `KNeighborsClassifier` function.

The abridged results for classification **without** finetuning for different tasks are presented in Figure 11 (one can find a full version for different methods in Section B).

Below, we analyse closer just the results with using ViT as a feature extractor forming the latent data space. One can see in Figure 6 that: (1) **without finetuning**, on a number of tasks the model shows competitive performance, and (2) with finetuning of the backbone, the difference between the standard backbone and

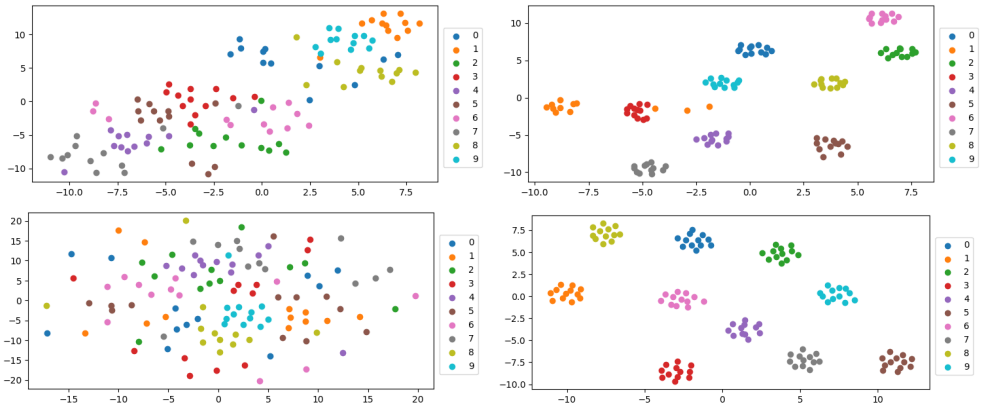


Figure 7: tSNE plots for original (top-left) vs finetuned (top-right) features of ResNet101, k-means prototypes; original (bottom left) vs finetuned (bottom right), ResNet101, random prototype selection, CIFAR-10

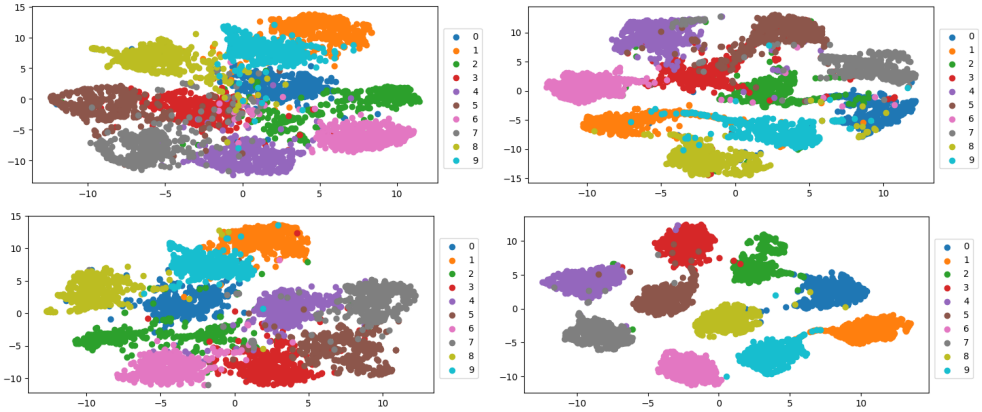


Figure 8: tSNE plots for original (top-left) vs finetuned (top-right) features of ViT, k-means prototypes; original (bottom left) vs finetuned (bottom right), ViT, random prototype selection, CIFAR-10

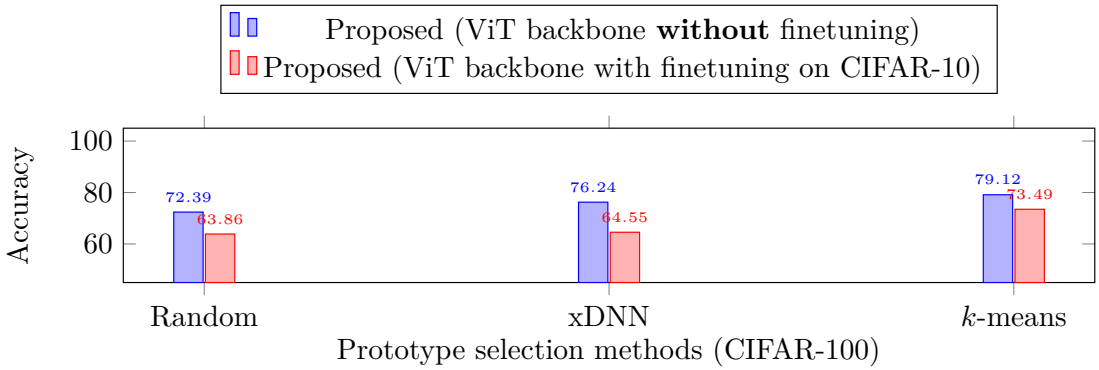


Figure 9: Comparison between the model performance on CIFAR-100 **without** finetuning and finetuning on CIFAR-10 for different prototype selection methods

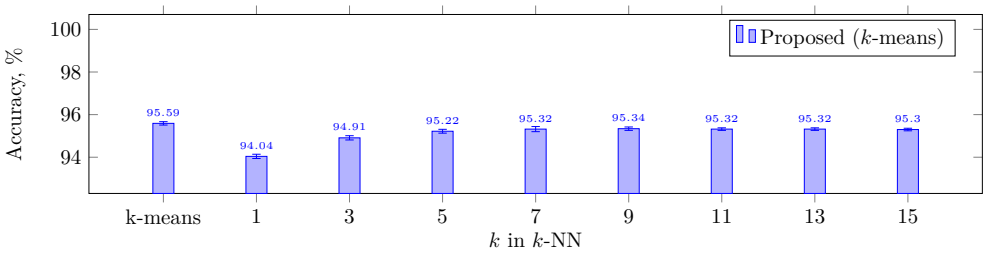


Figure 10: Comparison of results on CIFAR-10 (k nearest neighbours)

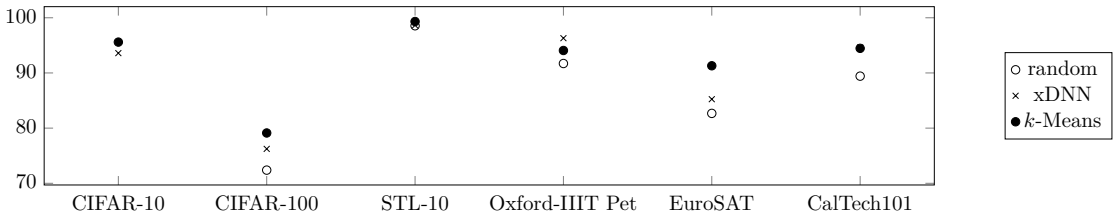


Figure 11: Results **without** finetuning for various problems (ViT)

the proposed model is insignificant within the confidence interval. In Figure 5, one can see the comparison of the time expenditure between the finetuned and **non-finetuned** model.

We also conducted (see Appendix C) an experiment to vary the selected number of prototypes for CIFAR-10 on ResNet101 backbone and the value k for the k-means method. It is a well-known specific of the k-means approach that it does require the number of clusters, k to be pre-defined. The online clustering method ELM, for example, does not require the number of clusters to be pre-defined, though it requires a single meta-parameter, called radius of the cluster to be pre-defined which can be related to the expected granulation level considering all data being normalized (Baruah & Angelov (2012)). Therefore, in Appendix B, we include results for ELM.

4.3 Demonstration of overfitting in the feature spaces

One clear advantage of transfer learning without finetuning is the dramatically lower computational costs reflected in the time expenditure. However, there is also another advantage: the evidence shows that the finetuned feature space shows less generalisation. In Figures 7 and 8, one can see the comparison of the tSNE plots between the finetuned and **non-finetuned** version of the method. While the finetuned method achieves clear separation on this task, using the same features to transfer to another task (from CIFAR-10 to CIFAR-100) leads to sharp decrease in performance (see Figure 9). Despite the time consumption and limited generalisation, the finetuned version of the proposed framework, see setting C), section 4 and also Tables 3 and 5. has one advantage: it demonstrates that with a small computational cost additional to finetuning, a standard DL classifier can be transformed into interpretable through prototypes ones with difference in performance within the confidence interval. While for the finetuned backbone, predictably, the results are not far off the standard DL models, they also show no significant difference between different types of prototype selection, including random (see Figure 6); however, for the non-finetuned results, the difference in top-1 accuracy between random and non-random prototype selection is drastic, reaching around 24% for VGG16.

The choice of prototypes greatly influences performance of a model when it is not finetuned as witnessed in a number of tasks for a number of backbone models. In Figure 4 and Appendix B, one can see that simple k-means prototype selection in the latent space can improve the performance by tens of percentage points; with the increase of the number of prototypes this difference decreases, but is still present. Furthermore, one can see that the proposed framework without finetuning and online prototype selection algorithm can be competitive with the state-of-the-art, especially when working in latent feature space defined by powerful

DNNs such as ViT on large data sets. When using finetuning, it is seen that the choice of prototypes, including random, does not make significant difference. This can be explained by the previous discussion of Figures 7 and 8: finetuning gives clear separation of features, so the features of the same class stay close; that makes the prototype choice practically unimportant for decision making.

4.4 Continual learning

The evidence from the previous sections motivates us to extend the analysis to continual learning problems: given a much smaller gap between the finetuned and non-finetuned models, can the IDEAL framework **without** finetuning compete with the state-of-the-art class-incremental learning baselines? It turns out the answer is affirmative. We repeat the setting from Rebuffi et al. (2017) (Section 4, iCIFAR-100 benchmark) using IDEAL without finetuning the latent space of the ViT-L model. This benchmark gradually adds the new classes with a class increment of 10, until it reaches 100 classes. The results, shown in Figure 12a, highlight excellent performance of the proposed method (the number of prototypes is set to 10% of data, however, as one can see in Appendix C, much lower number of prototypes, below 1000 or even just 10 per class on average can still lead to competitive results). While we report $64.18 \pm 0, 0.16$, $69.93 \pm 0.23\%$, 82.20 ± 0.23 for ResNet-50, ResNet-101, and ViT-L respectively, Wang et al. (2022b) reports in its Table 1 for the best performing method for class-incremental learning, based on ViT architecture and contrastive learning, accuracy of just $65.86 \pm 1.24\%$ (with the size of the buffer - 1000), while the original proposed benchmark Rebuffi et al. (2017) (iCarl) reports, according to Wang et al. (2022b), only $50.49 \pm 0.18\%$.

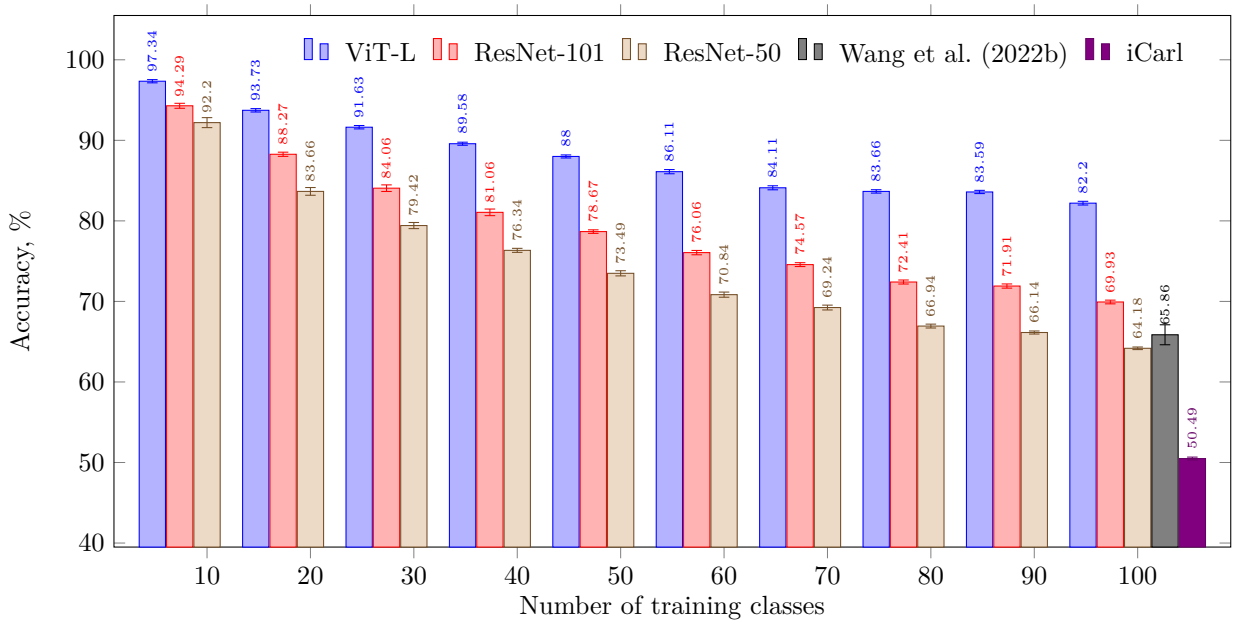
To demonstrate the consistent performance, we expanded iCIFAR-100 protocol to other datasets, referred to as iCaltech101 and iCIFAR-10. Figure 12 shows robust performance on iCaltech101 and iCIFAR-10. We use the class increment value of ten (eleven for the last step) and two for iCaltech101 and iCIFAR-10, respectively. The hyperparameters of the proposed methods are given in Appendix A. We see that for iCaltech101, the model performance changes insignificantly with adding the training classes, and all three datasets demonstrate performance similar to the offline classification performance (see Section 4.2).

4.5 Study of Interpretability

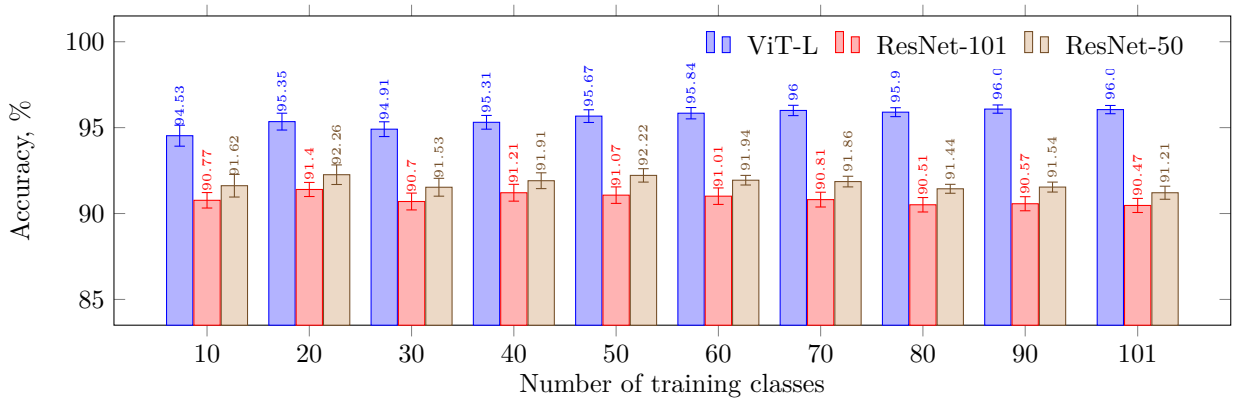
In Figure 15, we demonstrate the visual interpretability of the proposed model, through both most similar and most dissimilar prototypes. In addition, the results could be interpreted linguistically (see Appendix D). Figure 15 shows a number of quantitative examples for a number of datasets: Caltech101, STL-10, Oxford-IIIT Pets, all corresponding to the non-finetuned feature space scenario according to the experimental setup from Appendix A. We see that on a range of datasets, without any finetuning, the proposed IDEAL approach provides semantically meaningful interpretations. Furthermore, as there has been no finetuning, the ℓ^2 distances are defined in exact the same feature space and, hence, can be compared like-for-like between datasets (see subfigures 15a-15f). This strengthens the evidence of the benefits of our approach **without finetuning**. This experiment demonstrates that the incorrectly classified data tend to have larger distance to the closest prototypes than the correctly classified ones. Finally, Figure 16 outlines the evolution of predictions for the online scenario. For the sake of demonstration, we used the same setting as the one for the class-incremental lifelong learning detailed in Appendix A and Section 4.4, except from taking CIFAR-10 for class-incremental learning using ViT model with the increment batch of two classes. We trace the best and the worst matching and selected middle prototypes (according to the ℓ^2 metric) through the stages of class-incremental learning. For the successful predictions, while the best matching prototypes tend to be constant, the worst matching ones change over time when the class changes.

4.6 Impact of confounding on interpretations

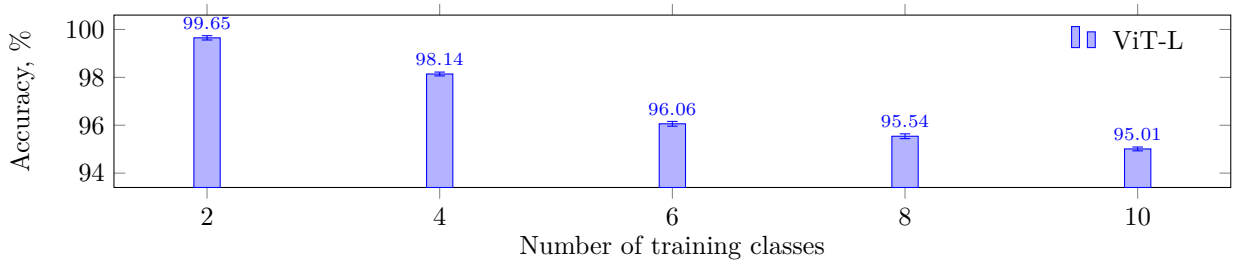
The phenomenon of confounding takes its origin in causal modelling and is informally described, as per Greenland et al. (1999), as *'a mixing of effects of extraneous factors (called confounders) with the effects of interest'*. In many real-world scenarios, images contain confounding features, such as watermarks or naturally occurring spurious correlations ('seagulls always appear with the sea on the background'). The challenge for the interpretable models is therefore multi-fold: (1) these models need to be resistant to such confounders



(a) iCIFAR-100



(b) iCaltech101



(c) iCIFAR10

Figure 12: Accuracy of IDEAL in class-incremental learning experiments for different backbones (ViT-L, ResNet-101 and 50); comparison with Wang et al. (2022b) and Rebuffi et al. (2017)

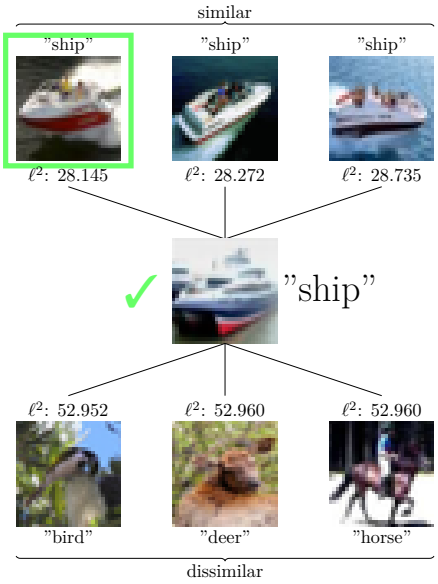


Figure 13: Interpreting the predictions of the proposed model (k-means (nearest), CIFAR-10, ViT)

(2) should these confounders interfere with the performance of the model, the model should highlight them in the interpretations.

To model confounding, we use the experimental setup from Bontempelli et al. (2022), which involves inpainting training images of three out of five selected classes of the CUB dataset with geometric figures (squares) which correlate with, but not caused by, the original data (e.g., every image of the `Crested Auklet` class is marked in the training data with a blue square). In Table 1, we compare the experimental results between the original (Wah et al. (2011)) and confounded (Bontempelli et al. (2022)) CUB dataset. We use the same original pre-trained feature spaces as stated in Appendix A. The finetuned spaces are obtained through finetuning on confounded CUB data from Bontempelli et al. (2022) for 15 epochs.

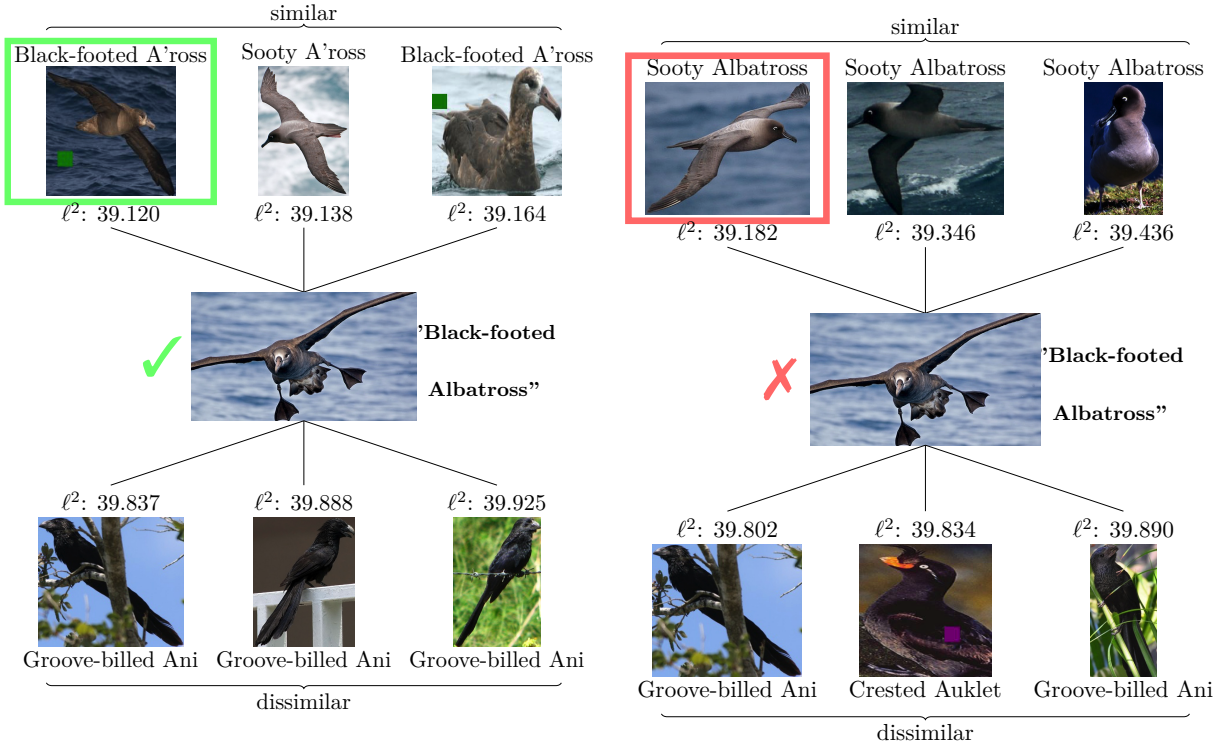
The results in Table 1 demonstrate clear advantage of models **without finetuning** on the confounded dataset for both k-means and k-means (nearest), in the case of ViT. Such gap, however, is much narrower for k-means prototype selection, VGG-16 and ResNet-50. It is consistent with the results of Question 1 on the larger performance gap of these models compared with the ViT. Furthermore, k-means (nearest) does not show improvements over finetuning in a k-means (nearest) scenario, in a stark contrast with the ViT results.

We demonstrate the interpretations for the confounding experiment in Figure 14. While the non-finetuned model successfully predicts the correct confounded class, black-footed albatross, the finetuned model fails at this scenario and predicts a similar class Sooty Albatross, which does not containt the confounder mark.

On the other hand, the finetuned model performs similarly or better on the original (not confounded) data. These results further build upon the hypothesis from Question 2 and demonstrate that the use of the proposed framework can help address the phenomenon of confounding.

5 Conclusion

The proposed IDEAL framework considers separately the representations from the latent spaces, learnt on generic large data sets, and learning of an interpretable, prototype-based model within this data space. We confirm an initial intuition that, in offline learning setting, contemporary ViT models drastically narrow the gap between the finetuned and non-finetuned models (Question 1). We justify the architectural choices for the framework such as selection of prototypes (Question 1) and demonstrate the margin of overfitting for finetuned ViTs (Question 2). This insight enables us to demonstrate that the proposed framework can



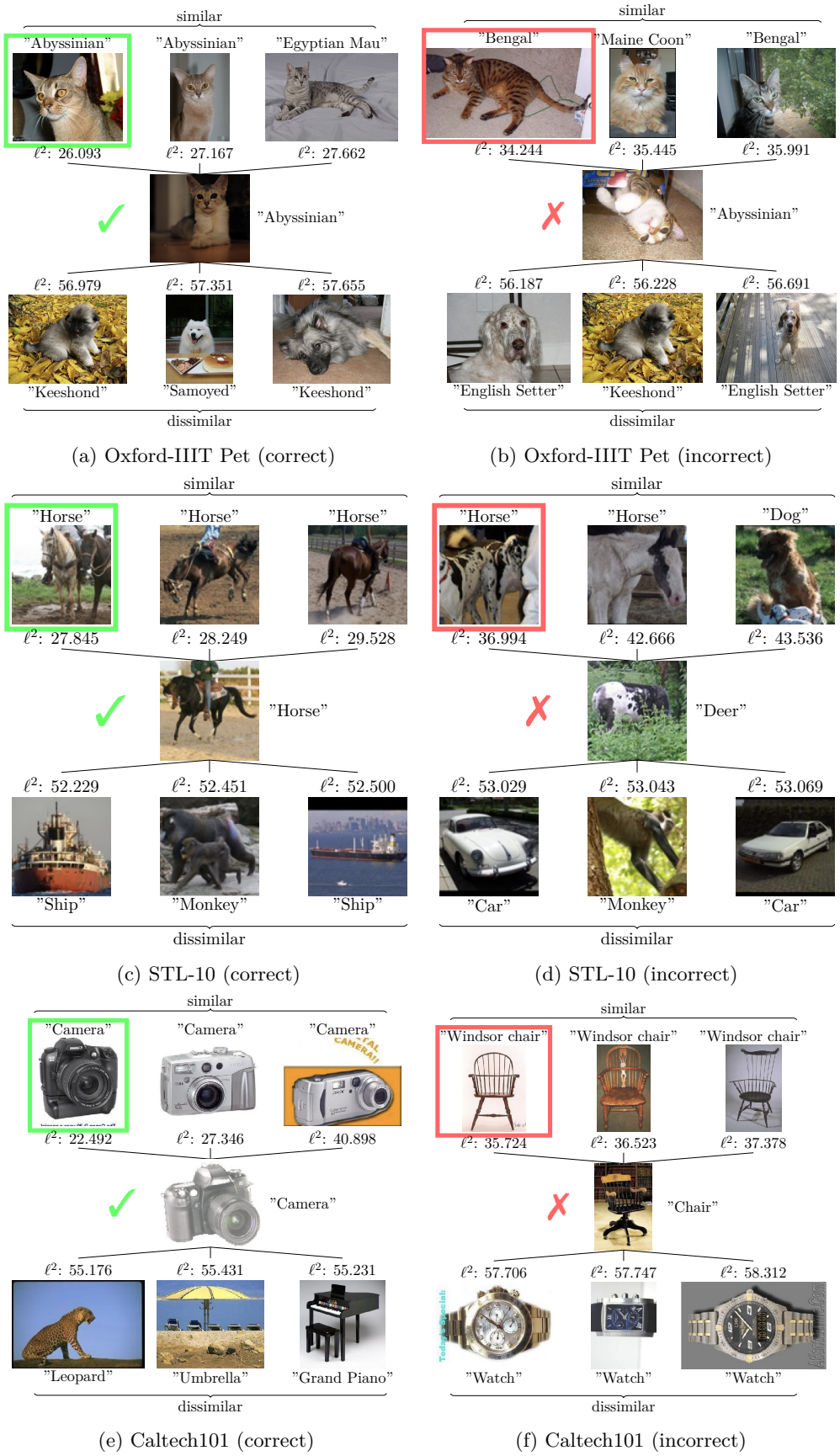
(a) Non-finetuned model interpretation (A'ross denotes 'Albatross')

(b) Finetuned model interpretation

Figure 14: Comparing the interpretations of the non-finetuned and finetuned model with confounding on confounded CUB (Bontempelli et al. (2022)) dataset

Feature space	Prototype selection	VGG16	ResNet-50	ViT
Confounded data (Bontempelli et al. (2022))				
Finetuned	N/A, backbone network	73.99 ± 2.91	70.42 ± 2.68	69.06 ± 4.40
Non-finetuned	k-means	$\mathbf{78.52 \pm 1.31}$	$\mathbf{76.68 \pm 1.63}$	$\mathbf{80.70 \pm 2.26}$
Finetuned	k-means	73.19 ± 1.43	67.16 ± 2.25	66.58 ± 5.81
Non-finetuned	k-means (nearest)	64.13 ± 1.37	67.68 ± 0.90	$\mathbf{82.88 \pm 2.17}$
Finetuned	k-means (nearest)	$\mathbf{71.00 \pm 2.92}$	69.03 ± 1.19	73.99 ± 5.19
Original data				
Finetuned	N/A, backbone network	83.66 ± 1.16	83.49 ± 1.22	93.92 ± 1.31
Non-finetuned	k-means	80.01 ± 1.27	80.10 ± 1.66	90.67 ± 1.13
Finetuned	k-means	81.98 ± 1.53	79.38 ± 2.87	92.85 ± 1.70
Non-finetuned	k-means (nearest)	72.11 ± 1.62	72.64 ± 1.87	88.57 ± 0.96
Finetuned	k-means (nearest)	$\mathbf{78.90 \pm 2.77}$	$\mathbf{80.05 \pm 2.64}$	$\mathbf{92.80 \pm 1.77}$

Table 1: F1 score comparison for CUB dataset (Wah et al. (2011)), confidence interval calculated over five runs; all k-means runs are for 10% (15) clusters/prototypes; the better results within its category are highlighted in bold, taking into account the confidence interval. While for the original data finetuning has strong performance benefits, non-finetuned model has an edge over the finetuned one for all architectures; for k-means (nearest) the non-finetuned model still performs clearly better with ViT architecture than the finetuned counterpart.



(a) Oxford-IIIT Pet (correct)

(b) Oxford-IIIT Pet (incorrect)

(c) STL-10 (correct)

(d) STL-10 (incorrect)

(e) Caltech101 (correct)

(f) Caltech101 (incorrect)

Figure 15: Interpreting the predictions (k-means (nearest), OxfordIIITPets/STL-10/Caltech101, ViT)

Figure 16: iCIFAR-10 continual learning: evolution of prototype ranking

surpass the state-of-the-art class-incremental learning methods (Question 3). We demonstrate interpretations through prototypes provided by the framework in offline and class-incremental learning scenarios (Question 4). Finally (Question 5), we demonstrate that in non-causal confounding scenarios, for modern architectures, such as ViT, finetuning results in both inferior performance and interpretations.

Broader Impact Statement

The proposed approach goes beyond the paradigm of first training and then finetuning complex models to the new tasks, which is standard for the field, where both these stages of the approach use expensive GPU compute to improve the model performance. We show that contemporary architectures, trained with extensive data sets, can deliver competitive performance in a lifelong learning setting even without such expensive finetuning. This can deliver profound impact on democratisation of high-performance machine learning models and implementation on Edge devices, on board of autonomous vehicles, as well as address important problems of environmental sustainability by avoiding using much energy to train new latent representations and finetune, providing instead a way to re-use existing models. Furthermore, the proposed framework can help define a benchmark on how deep-learning latent representations generalise to new tasks.

This approach also naturally extends to class- and potentially, domain-incremental learning, enabling learning new concepts. It demonstrates that with large and complex enough latent spaces, relatively simple strategies of prototype selection, such as clustering, can deliver results comparable with the state-of-the-art in a fraction of time and compute efforts. Importantly, unlike most of the state-of-the-art approaches, as described in the Related work section of the main paper, the proposed framework directly provides interpretability in linguistic and visual form and provides improved resistance to spurious correlations in input features.

Limitations

This work does not aim for explaining the latent spaces of the deep-learning architecture; instead, it explores explainable-through-prototypes decision making process in terms of similarity to the prototypes in the latent space.

References

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.

Muhammad Aurangzeb Ahmad, Carly Eckert, and Ankur Teredesai. Interpretable machine learning in healthcare. In *Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics*, pp. 559–560, 2018.

David Aldous, Illdar Ibragimov, and Jean Jacod. *Ecole d'Ete de Probabilites de Saint-Flour XIII, 1983*, volume 1117. Springer, 1983.

Plamen Angelov and Xiaowei Gu. *Empirical approach to machine learning*. Springer, 2019.

Plamen Angelov and Eduardo Soares. Towards explainable deep neural networks (xdnn). *Neural Networks*, 130:185–194, 2020.

Plamen Angelov and Xiaowei Zhou. Evolving fuzzy-rule-based classifiers from data streams. *Ieee transactions on fuzzy systems*, 16(6):1462–1475, 2008.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68 (3):337–404, 1950.

Akanksha Atrey, Kaleigh Clary, and David Jensen. Exploratory not explanatory: Counterfactual analysis of saliency maps for deep reinforcement learning. *arXiv preprint arXiv:1912.05743*, 2019.

Rashmi Dutta Baruah and Plamen Angelov. Evolving local means method for clustering of streaming data. In *2012 IEEE international conference on fuzzy systems*, pp. 1–8. IEEE, 2012.

Dimitris Bertsimas, Angela King, and Rahul Mazumder. Best subset selection via a modern optimisation lens. *The Annals of Statistics*, 44(2):813–852, 2016.

Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. *The Annals of Applied Statistics*, pp. 2403–2424, 2011.

Moritz Böhle, Mario Fritz, and Bernt Schiele. B-cos networks: alignment is all we need for interpretability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10329–10338, 2022.

Andrea Bontempelli, Stefano Teso, Katya Tentori, Fausto Giunchiglia, and Andrea Passerini. Concept-level debugging of part-prototype networks. *arXiv preprint arXiv:2205.15769*, 2022.

Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, 1992.

Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.

Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Robert French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Sander Greenland, Judea Pearl, and James M Robins. Confounding and collapsibility in causal inference. *Statistical science*, 14(1):29–46, 1999.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. In *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*, pp. 204–207. IEEE, 2018.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

Been Kim, Cynthia Rudin, and Julie A Shah. The bayesian case model: A generative approach for case-based reasoning and prototype classification. *Advances in neural information processing systems*, 27, 2014.

Jinkyu Kim and John Canny. Interpretable learning for self-driving cars by visualizing causal attention. In *Proceedings of the IEEE international conference on computer vision*, pp. 2942–2950, 2017.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2661–2671, 2019.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25, 2012.

Christiaan Lamers, René Vidal, Nabil Belbachir, Niki van Stein, Thomas Bäeck, and Paris Giampouras. Clustering-based domain-incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3384–3392, 2023.

Fei-Fei Li, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pp. 281–297. Oakland, CA, USA, 1967.

Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24 (2):227–234, 1995.

Meike Nauta, Ron Van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14933–14943, 2021.

Allen Newell, John C Shaw, and Herbert A Simon. Report on a general problem solving program. In *IFIP congress*, volume 256, pp. 64. Pittsburgh, PA, 1959.

German Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

Omkar Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Uwe Peters. Explainable ai lacks regulative reasons: why ai and human decision-making are not equally opaque. *AI and Ethics*, pp. 1–12, 2022.

Tomaso Poggio and Federico Girosi. A sparse representation for function approximation. *Neural computation*, 10(6):1445–1454, 1998.

Milos Radovanovic, Alexandros Nanopoulos, and Mirjana Ivanovic. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11(sept):2487–2531, 2010.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph Lampert. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5533–5542. IEEE, 2017.

Frank Rosenblatt et al. *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms*, volume 55. Spartan books Washington, DC, 1962.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. doi: 10.1038/s42256-019-0048-x. URL https://doi.org/10.1038/s42256-019-0048-x.

Paul Ruvolo and Eric Eaton. Ella: An efficient lifelong learning algorithm. In *International conference on machine learning*, pp. 507–515. PMLR, 2013.

Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *Computational Learning Theory: 14th Annual Conference on Computational Learning Theory, COLT 2001 and 5th European Conference on Computational Learning Theory, EuroCOLT 2001 Amsterdam, The Netherlands, July 16–19, 2001 Proceedings 14*, pp. 416–426. Springer, 2001.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. 2014.

Mannat Singh, Laura Gustafson, Aaron Adcock, Vinicius de Freitas Reis, Bugra Gedik, Raj Prateek Kosaraju, Dhruv Mahajan, Ross Girshick, Piotr Dollár, and Laurens van der Maaten. Revisiting Weakly Supervised Pre-Training of Visual Perception Models. In *CVPR*, 2022.

Alex Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14: 199–222, 2004.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

Eduardo Soares, Plamen Angelov, and Ziyang Zhang. An explainable approach to deep learning from ct-scans for covid identification. 2021.

Hugo Steinhaus et al. Sur la division des corps matériels en parties. *Bull. Acad. Polon. Sci*, 1(804):801, 1956.

Michael Tipping. The relevance vector machine. *Advances in neural information processing systems*, 12, 1999.

Michael Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244, 2001.

Gido van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, pp. 1–13, 2022.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

Wenguan Wang, Cheng Han, Tianfei Zhou, and Dongfang Liu. Visual recognition with deep nearest centroids. In *International Conference on Learning Representations (ICLR)*, 2023.

Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. *Advances in Neural Information Processing Systems*, 35: 5682–5695, 2022a.

Zhen Wang, Liu Liu, Yajing Kong, Jiaxian Guo, and Dacheng Tao. Online continual learning with contrastive vision transformer. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XX*, pp. 631–650. Springer, 2022b.

Dennis Wei, Rahul Nair, Amit Dhurandhar, Kush R Varshney, Elizabeth Daly, and Moninder Singh. On the safety of interpretable machine learning: A maximum deviation approach. *Advances in Neural Information Processing Systems*, 35:9866–9880, 2022.

Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine learning*, 23:69–101, 1996.

Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3014–3023, 2021.

Dagmar Zeithamova, W Todd Maddox, and David M Schnyer. Dissociable prototype learning systems: evidence from brain imaging and behavior. *Journal of Neuroscience*, 28(49):13194–13201, 2008.

Ziyang Zhang, Plamen Angelov, Eduardo Soares, Nicolas Longepe, and Pierre Philippe Mathieu. An interpretable deep semantic segmentation method for earth observation. *arXiv preprint arXiv:2210.12820*, 2022.

Junxian Zhu, Canhong Wen, Jin Zhu, Heping Zhang, and Xueqin Wang. A polynomial algorithm for best-subset selection problem. *Proceedings of the National Academy of Sciences*, 117(52):33117–33123, 2020.

A Experimental setup

In this work, all the experiments were conducted in PyTorch 2.0.0. The pre-trained models used in these experiments were obtained from TorchVision [1] while the finetuned models have been obtained from three different sources:

1. *Models that come from MMPreTrain* [2]. Specifically, ResNet50 and ResNet101 finetuned on the CIFAR-10, and ResNet 50 finetuned on CIFAR-100.

2. *finetuned TorchVision models.* finetuning was conducted by continuing the EBP across all network layers. Such models include VGG-16 and Vision Transformer (ViT) finetuned on CIFAR-10, as well as ResNet101, VGG-16, and ViT finetuned on CIFAR-100. For ResNet101 and VGG-16 models, we ran the training for 200 epochs, while the Vision Transformer models were trained for 10 epochs. The Stochastic Gradient Descent (SGD) optimizer was employed for all models, with a learning rate of 0.0005 and a momentum value of 0.9.

3. *Linearly finetuned TorchVision models.* In such case, only the linear classifier was trained and all the remaining layers of the network were fixed. For these models, we conducted training for 200 epochs for ResNet50, ResNet101, and VGG16, and 25 epochs for the ViT models. We adopted the Stochastic Gradient Descent (SGD) optimizer, with a learning rate of 0.001 and a momentum parameter set at 0.9.

We utilized k-means clustering and random selection methods, setting the number of prototypes for each class at 10% of the training data for the corresponding classes. Besides, we also set it to 12 per class and conducted experiments for ResNet50, ResNet101, and VGG-16 on CIFAR-10 and CIFAR-100 datasets, enabling us to evaluate the impact of varying the number of prototypes.

For ELM online clustering method, we experimented with varying radius values for each specific dataset and backbone network. We selected a radius value that would maintain the number of prototypes within the range of 0-20% of the training data. In the experiments without finetuning on the CIFAR-10 dataset, we set the radius to 8, 10, 19, and 12 for ResNet50, ResNet101, VGG-16, and Vision Transformer (ViT) models respectively. The radius was adjusted to 8, 11, 19, and 12 for these models when conducting the

[1] https://pytorch.org/vision/main/models.html
[2] https://github.com/open-mmlab/mmpretrain

same tasks without finetuning on CIFAR-100. For STL10, Oxford-IIIT Pets, EuroSAT, and CalTech101 datasets, the radius was set to 13 across all ELM experiments. In contrast, the xDNN model did not require hyper-parameter settings as it is inherently a parameter-free model.

We performed all experiments for Sections 4.2 and 4.4 of the main paper 5 times and report mean values and standard deviations for our results, with the exception of the finetuned backbone models where we just performed finetuning once (or sourced finetuned models as detailed above).

For the class-incremental learning experiments in Section **??**, we use the k-means clustering method for prototype selection and set the number of prototypes to 10% of the training data. Each time we add the incremental classes, the existing prototypes are be changed, the algorithm simply adds the prototypes for the new classes to the existing prototypes. All the experiments were executed 10 times to allow a robust comparison with the benchmark results.

To ensure a consistent and stable training environment, for every experiment we used a single NVIDIA V100 GPU from a cluster.

B Complete experimental results

Tables 2-9 contain extended experimental results for multiple benchmarks and feature extractors. These results further demonstrate the findings of the main paper.

Table 2 demonstrates the data behind Figures 3, 4, 5 of the main paper. It also highlights the performance of the k-means model on ViT-L latent space, when the nearest real training data point to the k-means cluster centre is selected (labelled as k-means (nearest)). One can also see that even with the small number of selected prototypes, the algorithm delivers competitive performance without finetuning.

Table 3 compares different latent spaces and gives the number of free (optimised) parameters for the scenario of finetuning of the models. With a small additional number of parameters, which is the number of possible prototypes, one can transform the opaque architectures into ones interpretable through proximity and similarity to prototypes within the latent space (this is highlighted in the interpretability column).

Tables 4-9 repeat the same analysis, expanded from Figure 5 of the main paper for different data sets. The results show remarkable consistency with the previous conclusions and further back up the claims of generalisation to different classification tasks.

C Sensitivity analysis for the number of prototypes

Figure 17 further backs up the previous evidence that even with a small number of prototypes, the accuracy is still high. It shows, however, that there is a trade-off between the number of prototypes and accuracy. It also shows, that after a few hundred prototypes per class on CIFAR-10 and CIFAR-100 tasks, the performance does not increase and may even slightly decrease, indicating saturation.

D Linguistic interpretability of the proposed framework outputs

To back up interpretability claim, we present two additional interpretability scenarios complementing the one in Figure 12 of the main text.

First, we show the symbolic decision rules in Figure 18. These symbolic rules are created using ViT-L backbone, with the prototypes selected using the nearest real image to k-means cluster centroids, in a no-finetuning scenario for OxfordIIITPets dataset.

Second, in Figure 19 we show how the overall pipeline of the proposed method can be summarised in interpretable-through-prototypes fashion. We show the normalised distance obtained through dividing by the sum of distances to all prototypes. This is to improve the perception and give relative, bound between 0 and 1, numbers for the prototype images.

FE	method	accuracy (%)	#prototypes	time, s
ResNet50	random	65.55 ± 1.93	120(0.24%)	85
	random	80.40 ± 0.37	5,000(10%)	85
	ELM	81.17 ± 0.04	5,500(11%)	365
	xDNN	81.44 ± 0.33	115(0.23%)	103
	k-means	84.12 ± 0.19	120(0.24%)	201
	k-means	$\mathbf{86.65 \pm 0.15}$	5,000(10%)	1,138
Resnet101	random	78.08 ± 1.38	120(0.24%)	129
	random	87.66 ± 0.25	5,000(10%)	129
	ELM	88.22 ± 0.09	7,154(14.31%)	524
	xDNN	88.13 ± 0.42	118(0.24%)	145
	k-means	90.19 ± 0.15	120(0.24%)	245
	k-means	$\mathbf{91.50 \pm 0.07}$	5,000(10%)	1,194
VGG-16	random	50.13 ± 2.37	120(0.24%)	95
	random	65.06 ± 0.32	5,000(10%)	95
	ELM	72.31 ± 0.08	1,762(3.52%)	215
	xDNN	70.03 ± 0.96	103(0.21%)	132
	k-means	74.48 ± 0.16	120(0.24%	346
	k-means	$\mathbf{75.94 \pm 0.15}$	5,000(10%)	2,362
ViT	random	93.23 ± 0.11	5,000(10%)	597
	ELM	90.61 ± 0.14	6,685(13.37%)	889
	xDNN	93.59 ± 0.12	112(0.2%)	606
	k-means	$\mathbf{95.59 \pm 0.08}$	5,000(10%)	925
ViT-L	k-means	$\mathbf{96.48 \pm 0.05}$	5,000(10%)	4,375
	k-means (nearest)	95.62 ± 0.07	5,000(10%)	4,352

Table 2: CIFAR-10 classification task comparison for the case of no finetuning of the feature extractor

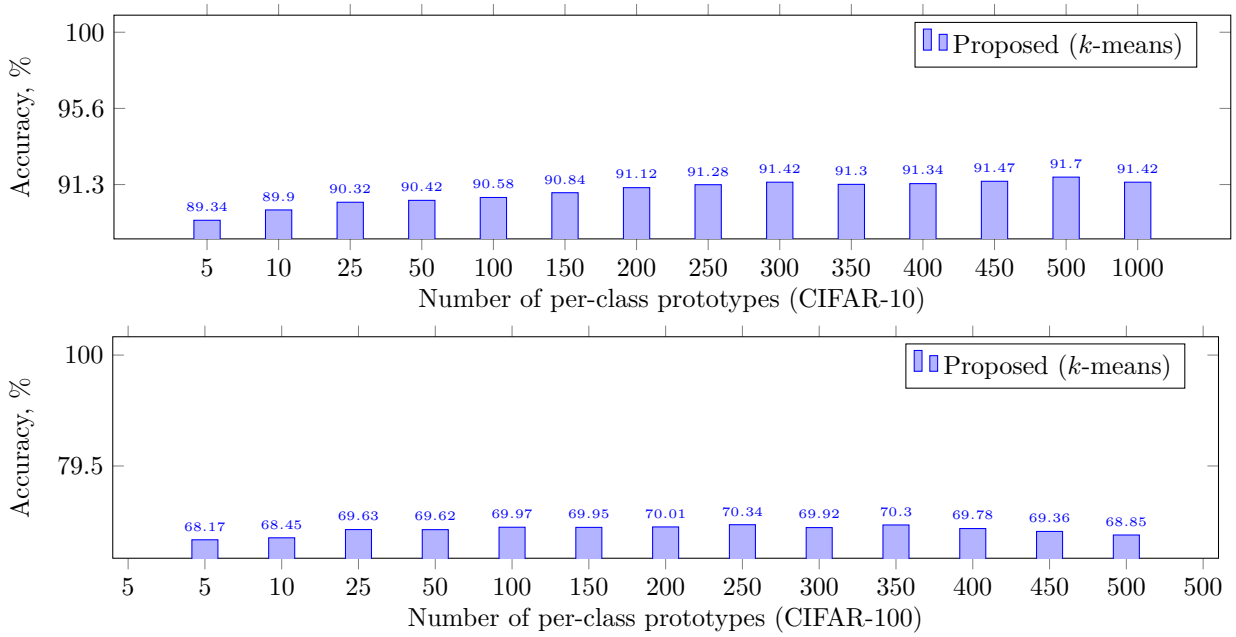


Figure 17: Accuracy sensitivity to the number of per-class prototypes (k-means, ResNet101, no finetuning)

FE	method	accuracy (%)	#parameters	#prototypes	time, s	interpretability
ResNet50	ResNet50	95.55 (80.71*)	$\sim 25M$ $(20K)$		$36,360$ $(13,122^*)$	✗
	random	94.92 ± 0.02	$\sim 25M + 50K$	$120(0.24\%)$	$36,360 + 24$	✓
	random	95.32 ± 0.09	$\sim 25M + 50K$	$5,000(10\%)$	$36,360 + 24$	✓
	xDNN	95.32 ± 0.12	$\sim 25M + 50K$	$111(0.22\%)$	$36,360 + 43$	✓
	k-means	94.91 ± 0.14	$\sim 25M + 50K$	$120(0.24\%)$	$36,360 + 208$	✓
	k-means	95.50 ± 0.06	$\sim 25M + 50K$	$5,000(10\%)$	$36,360 + 1,288$	✓
ResNet101	Resnet101	95.58 (84.44*)	$\sim 44M$ $(20K)$		$36,360$	✗
	random	95.47 ± 0.06	$\sim 44M + 50K$	$120(0.24\%)$	$36,360 + 37$	✓
	random	95.51 ± 0.01	$\sim 44M + 50K$	$5,000(10\%)$	$36,360 + 37$	✓
	xDNN	95.50 ± 0.10	$\sim 44M + 50K$	$107(0.21\%)$	$36,360 + 54$	✓
	k-means	95.55 ± 0.03	$\sim 44M + 50K$	$120(0.24\%)$	$36,360 + 231$	✓
	k-means	95.51 ± 0.04	$\sim 44M + 50K$	$5,000(10\%)$	$36,360 + 1,357$	✓
VGG-16	VGG-16	92.26 (83.71*)	$\sim 138M$ $(41K)$		$40,810$	✗
	random	87.48 ± 0.72	$\sim 138M + 50K$	$120(0.24\%)$	$40,810 + 94$	✓
	random	90.86 ± 0.19	$\sim 138M + 50K$	$5,000(10\%)$	$40,810 + 94$	✓
	xDNN	91.42 ± 0.25	$\sim 138M + 50K$	$102(0.20\%)$	$40,810 + 123$	✓
	k-means	92.24 ± 0.10	$\sim 138M + 50K$	$120(0.24\%)$	$40,810 + 369$	✓
	k-means	92.55 ± 0.16	$\sim 138M + 50K$	$5,000(10\%)$	$40,810 + 2,408$	✓
ViT	ViT	98.51 (96.08*)	$\sim 86M$ $(8K)$		$15,282$ $(15,565^*)$	✗
	random	98.56 ± 0.02	$\sim 86M + 50K$	$5,000(10\%)$	$15,282 + 598$	✓
	xDNN	98.00 ± 0.14	$\sim 86M + 50K$	$117(0.23\%)$	$15,282 + 607$	✓
	k-means	98.53 ± 0.04	$\sim 86M + 50K$	$5,000(10\%)$	$15,282 + 938$	✓

Table 3: CIFAR-10 classification task comparison for the case of finetuned models ($*$ denotes linear finetuning of the DL model)

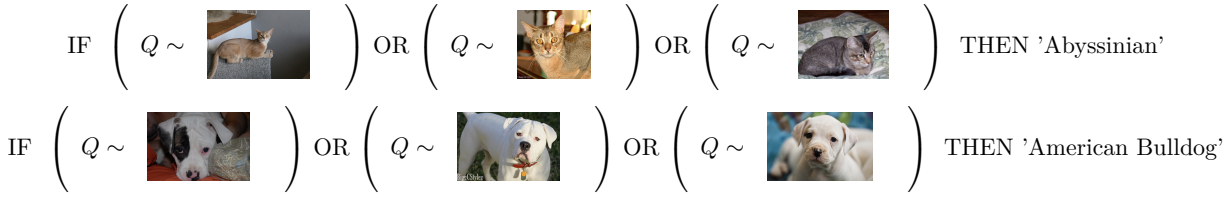


Figure 18: An example of symbolic decision rules (OxfordIIITPets), Q denotes the query image

FE	method	accuracy (%)	#prototypes	time, s
ResNet50	random	41.66 ± 0.74	$1,200(2.4\%)$	82
	random	54.37 ± 0.43	$10,000(20\%)$	82
	ELM	57.94 ± 0.11	$7,524(15.05\%)$	129
	xDNN	58.25 ± 0.64	$884(1.77\%)$	98
	k-means	62.67 ± 0.26	$1,200(2.4\%)$	124
	k-means	64.07 ± 0.37	$10,000(20\%)$	258
ResNet101	random	50.25 ± 0.71	$1,200(2.4\%)$	128
	random	61.90 ± 0.41	$10,000(20\%)$	128
	ELM	64.42 ± 0.12	$4,685(9.37\%)$	161
	xDNN	64.60 ± 0.39	$878(1.76\%)$	143
	k-means	68.59 ± 0.40	$1,200(2.4\%)$	170
	k-means	70.04 ± 0.12	$10,000(20\%)$	310
VGG16	random	26.16 ± 0.24	$1,200(2.4\%)$	94
	random	37.74 ± 0.48	$10,000(20\%)$	94
	ELM	48.53 ± 0.05	$2,878(5.76\%)$	122
	xDNN	47.78 ± 0.41	$871\ (1.74\%)$	119
	k-means	51.99 ± 0.24	$1,200(2.4\%)$	175
	k-means	52.55 ± 0.27	$1,200(2.4\%)$	437
ViT	random	72.39 ± 0.21	$10,000(20\%)$	604
	ELM	69.94 ± 0.06	$8,828(17.66\%)$	642
	xDNN	76.24 ± 0.24	$830(1.66\%)$	613
	k-means	79.12 ± 0.28	$10,000(20\%)$	673
ViT-L	k-means	$\mathbf{82.18 \pm 0.14}$	$10,000(20\%)$	$3,905$
	k-means (nearest)	78.75 ± 0.29	$10,000(20\%)$	$3,909$

Table 4: CIFAR-100 classification task comparison for the case of no finetuning of the feature extractor

FE	method	accuracy (%)	#parameters	#prototypes	time, s	interpretability
ResNet50	ResNet50	79.70 (56.39*)	$\sim 25M$ $(205K)$		$36,360(13,003^*)$	✗
	random	78.94 ± 0.17	$\sim 25M + 50K$	$1,200(2.4\%)$	$36,360 + 28$	✓
	random	79.52 ± 0.17	$\sim 25M + 50K$	$10,000(20\%)$	$36,360 + 28$	✓
	xDNN	79.75 ± 0.12	$\sim 25M + 50K$	$859(1.72\%)$	$36,360 + 45$	✓
	k-means	79.84 ± 0.07	$\sim 25M + 50K$	$1,200(2.4\%)$	36,360+82	✓
	k-means	79.77 ± 0.07	$\sim 25M + 50K$	$10,000(20\%)$	36,360+219	✓
ResNet101	ResNet50	84.38 (63.18*)	$\sim 44M$ $(205K)$		$45,619(18,955^*)$	✗
	random	82.26 ± 0.15	$\sim 44M + 50K$	$1,200(2.4\%)$	$45,619 + 175$	✓
	random	80.75 ± 0.19	$\sim 44M + 50K$	$10,000(20\%)$	$45,619 + 175$	✓
	xDNN	81.13 ± 0.16	$\sim 44M + 50K$	$831(1.66\%)$	$45,619 + 191$	✓
	k-means	83.03 ± 0.06	$\sim 44M + 50K$	$1,200(2.4\%)$	$45,619 + 220$	✓
	k-means	83.14 ± 0.19	$\sim 44M + 50K$	$10,000(20\%)$	$45,619 + 439$	✓
VGG-16	VGG-16	75.08 (62.74*)	$\sim 138M$ $(410K)$		$41,038(17,098^*)$	✗
	random	53.83 ± 0.91	$\sim 138M + 50K$	$1,200(2.4\%)$	$41,038 + 92$	✓
	random	64.17 ± 0.36	$\sim 138M + 50K$	$10,000(20\%)$	$41,038 + 92$	✓
	xDNN	72.63 ± 0.11	$\sim 138M + 50K$	$907(1.81\%)$	$41,038 + 120$	✓
	k-means	73.83 ± 0.16	$\sim 138M + 50K$	$1,200(2.4\%)$	$41,038 + 199$	✓
	k-means	73.73 ± 0.23	$\sim 138M + 50K$	$10,000(20\%)$	$41,038 + 460$	✓
ViT	ViT	90.29(82.79*)	$\sim 86M$ $(77K)$		$15,536(15,423^*)$	✗
	random	89.90 ± 0.10	$\sim 86M + 50K$	$10,000(20\%)$	$15,536 + 621$	✓
	xDNN	89.17 ± 0.18	$\sim 86M + 50K$	$809(1.61\%)$	$15,536 + 630$	✓
	k-means	90.48 ± 0.05	$\sim 86M + 50K$	$10,000(20\%)$	$15,536 + 695$	✓

Table 5: CIFAR-100 classification task comparison for the case of finetuned models ($*$ denotes linear finetuning of the DL model)

FE	method	accuracy (%)	#prototypes	time, s
ViT	random	98.55 ± 0.09	$500(10\%)$	61
	ELM	95.27 ± 0.03	$271(5.42\%)$	63
	xDNN	98.63 ± 0.12	$84(1.68\%)$	62
	k-means	99.32 ± 0.03	$500(10\%)$	65
ViT-L	k-means	99.71 ± 0.02	$500(10\%)$	377
	k-means(nearest)	99.56 ± 0.05	$500(10\%)$	377

Table 6: STL10 classification task comparison for the case of no finetuning (linear finetuning of the ViT gives 98.97%)

FE	method	accuracy (%)	#prototypes	time, s
ViT	random	90.82 ± 0.53	$365(9.92\%)$	48
	ELM	90.85 ± 0.03	$122(3.32\%)$	49
	xDNN	96.30 ± 0.23	$239(6.49\%)$	49
	k-means	94.07 ± 0.20	$365(9.92\%)$	50
ViT-L	k-means	95.78 ± 0.19	$365(9.92\%)$	279
	k-means (nearest)	94.76 ± 0.30	$740(9.92\%)$	279

Table 7: OxfordIIITPets classification task comparison for the case of no finetuning (linear finetuning of ViT gives 94.41%)

FE	method	accuracy (%)	#prototypes	time, s
ViT	random	82.67 ± 0.54	$2,154(9.97\%)$	266
	ELM	83.69 ± 0.01	$528(2.44\%)$	277
	xDNN	85.24 ± 1.05	$102(0.47\%)$	269
	k-means	91.30 ± 0.16	$2,154(9.97\%)$	330
ViT-L	k-means	88.93 ± 0.22	$2,154(9.97\%)$	1685
	k-means(nearest)	83.97 ± 0.16	$2,154(9.97\%)$	1685

Table 8: EuroSAT classification task comparison for the case of no finetuning (linear finetuning gives 95.17%)

FE	method	accuracy (%)	#prototypes	time, s
ViT	random	89.42 ± 0.32	$649(9.35\%)$	96
	ELM	91.12 ± 0.07	$516(7.43\%)$	97
	xDNN	94.61 ± 0.94	$579(8.34\%)$	97
	k-means	94.46 ± 0.44	$649(9.35\%)$	99
ViT-L	k-means	96.08 ± 0.34	$649(9.35\%)$	515
	k-means (nearest)	93.74 ± 0.42	$649(9.35\%)$	517

Table 9: CalTech101 classification task comparison (linear finetuning gives 96.26%)

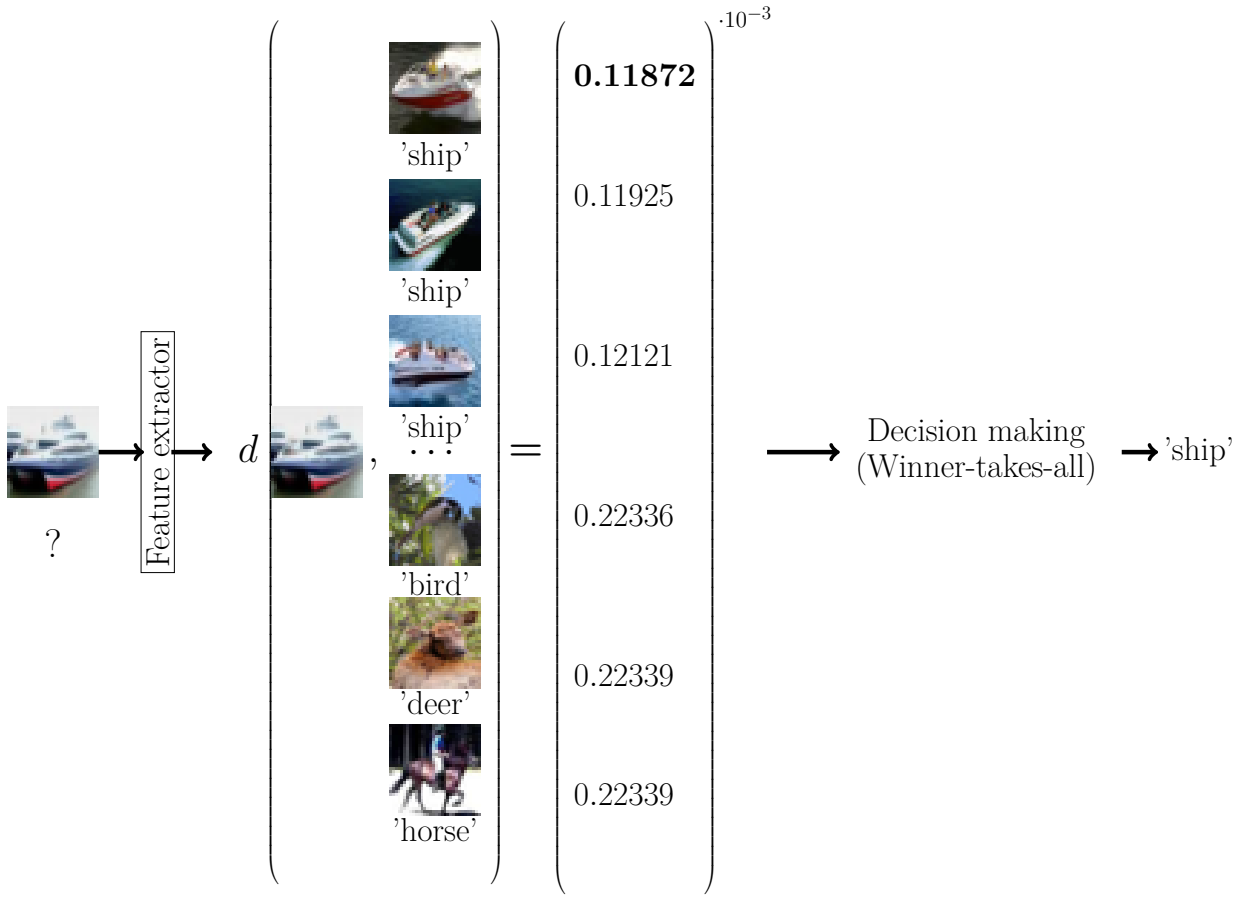


Figure 19: Interpreting the model predictions (k-means (nearest), 500 clusters per class, CIFAR-10, ViT)