Self-Supervised Training of Interpretable Neural Networks for Medical Applications

Michelle Espranita Liman

Technical University of Munich
michelle.liman@tum.de

Abstract. In the medical field, the adoption of Deep Learning has been slow due to the black-box nature of neural networks. Hence, ensuring the interpretability of these networks is essential for their incorporation into routine clinical workflows. Additionally, the scarcity of labelled medical datasets presents a significant challenge, making it difficult to rely on supervised training. To tackle these issues, we propose two methods — PCL-ProtoPNet and PCL-NW — which prioritize the *interpretability* of neural networks and leverage *self-supervised learning* on unlabelled datasets. We evaluate our methods on Alzheimer's Disease classification using the ADNI dataset. From our experiments, we conclude that our methods have not yet surpassed their purely supervised counterparts.

Keywords: Interpretability · Self-supervised learning · Alzheimer's Disease.

1 Introduction

While neural networks have made remarkable strides in Deep Learning, the adoption of these methods in the medical field has been slow. This is because most neural networks are black-box models that arrive at their predictions in an intransparent manner: It is not understandable to humans how these predictions are made. In contexts where human lives are at stake, such as healthcare, clinicians understandably hesitate to rely on tools whose decision-making process is not transparent [3]. Moreover, legal frameworks like the EU's General Data Protection Regulation (GDPR), require tools to provide their reasoning process before they can handle personal data [16]. Hence, the interpretability of neural networks is crucial for their integration into routine clinical workflows.

There is no formal consensus on how interpretability is defined and it is often used interchangeably with explainability. For our work, we adhere to the definition provided by [14], which is "an attempt to explain the decision-making process of the model in a way that is understandable for the end-users". Neural networks typically provide interpretability through two primary ways: 1) post-hoc explanations generated after model training; and 2) integration of interpretability into the model design. The latter approach has the advantage in that the explanations it provides genuinely reflect how the model arrives at its

decisions. In contrast, post-hoc explanations are merely approximations that may not accurately depict the model's behavior [13, 14].

For medical applications, a large body of work focuses on incorporating interpretability into the model design through case-based reasoning. Models employing case-based reasoning compare the features extracted from a query with features of learned prototypes, which are representative embeddings for a group of semantically similar instances [10]. The final prediction is the weighted sum of the similarity scores between the query and the prototypes.

Prototypes can be visualized by projecting them to the nearest train image or patch. Hence, they can be used to explain how the prediction on the query was made: We can visualize the most similar prototypes to the query. A notable implementation of case-based reasoning is the Prototypical Part Network (ProtoPNet) [1], which uses class-specific prototypes to explain the query. Contrary to popular belief that there is always a trade-off between performance and interpretability [22], ProtoPNet performs on par with black-box models.

Mohammadjafari et al. [12] uses ProtoPNet to provide interpretability in Alzheimer's Disease (AD) classification (AD/healthy) on 2D brain MRI images. Combining ProtoPNet and transfer learning, they achieved accuracy as high as 91% on the ADNI dataset. Another work by Wang et al. [19] combines the high accuracy of a global image classifier and the interpretability of ProtoPNet for mammogram classification (cancer/non-cancer). Their ProtoPNet distills knowledge from the global classifier, and both models form an ensemble by averaging the predictions by the global classifier and the ProtoPNet. For a 3-way AD classification (AD/MCI/CN), Wolf et al. [21] not only provides interpretability on 3D brain MRI scans through XProtoNet [8] (another implementation of case-based reasoning), but it also provides interpretability on tabular data.

Besides prototypes, recent work [17, 18] introduces the Nadaraya-Watson (NW) head, which provides interpretability using a support set. The support set contains samples with known labels which influence the model's prediction on the query: The prediction is a weighted sum of the support labels, in which the weights are similarity scores between the query and the support samples.

All the works mentioned above require labels or supervision for training the models. However, labelling medical datasets is a laborious and expensive process, both in terms of time and money. An alternative strategy is self-supervised learning, which learns representations of data without labels, enabling us to leverage large, unlabelled datasets. Prototypical Contrastive Learning (PCL) [10] is a self-supervised learning technique that aims to learn representations that capture high-level semantics with the help of prototypes. In contrast to ProtoPNet, PCL's prototypes are not visualizable; thus, PCL is not interpretable.

In this paper, we aim to incorporate the *interpretability* of neural networks and *self-supervised learning* on large, unlabelled datasets. We propose two methods — PCL-ProtoPNet and PCL-NW. The "PCL" component of the method allows self-supervised learning, while the "ProtoPNet" or "NW" component provides interpretability. We evaluate our methods on the challenging 3-way AD classification task (AD/MCI/CN) using 3D brain MRI images. We use PCL

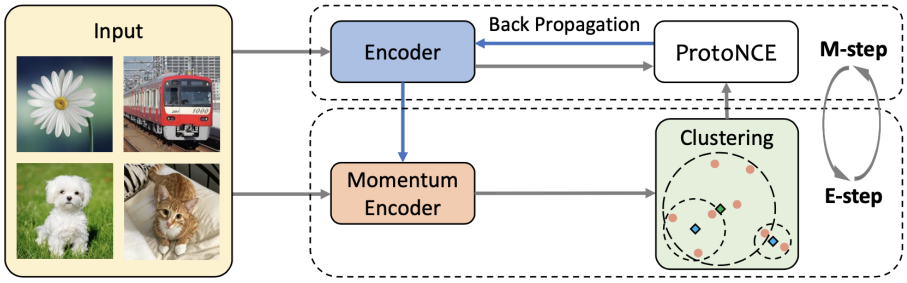

Fig. 1. Training process of Prototypical Contrastive Learning (PCL), taken from [10].

as a pre-training step on the unlabelled UK BioBank (UKBB) dataset. Subsequently, we either fix or finetune the resulting encoder and classification head on the labelled Alzheimer's Disease Neuroimaging Initiative (ADNI) dataset.

2 Methods

2.1 PCL-ProtoPNet

PCL-ProtoPNet is a combination of Prototypical Contrastive Learning (PCL) [10] and Prototypical Part Network (ProtoPNet) [1]. PCL-ProtoPNet provides interpretability by using **prototypes** $P = \{p_j\}_{j=1}^{k}$, which are representative embeddings for a group of train samples. In the context of images, prototypes can be visualized as patches or images. By visualizing which prototypes correspond most to a given query image, we can gain insights into how the model achieves its prediction.

ProtoPNet is a supervised learning method that learns k prototypes for each class. The learned prototypes are then projected ("pushed") onto the nearest latent training patch from the same class as the prototype. Meanwhile, PCL is a self-supervised learning method that learns k prototypes for the whole dataset, given that it does not know the class of each sample.

Originally, PCL was not designed for interpretability using prototypes but rather for learning self-supervised representations that capture high-level semantics with the help of prototypes. However, combining PCL and ProtoPNet allows us to leverage the advantages of both methods: 1) using self-supervised learning, which means no labelled dataset is required for learning prototypes; and 2) gaining interpretability of the model predictions.

We follow the same 3-stage training process as ProtoPNet with some modifications.

Training the encoder and learning the prototypes using PCL In this stage, we replace ProtoPNet's method of training the encoder f_θ and learning prototypes with PCL, enabling the learning of prototypes without supervision

to leverage our larger, unlabelled dataset. We can think of this stage as a pre-training stage. Training alternates between two steps (Fig. 1): First, we train the encoder to minimize the ProtoNCE loss. Second, we perform k-means clustering on the latent features of the train samples generated by the momentum encoder (as defined in MoCo [4]), resulting in k centroids $C = \{c_j\}_{j=1}^{k}$ that correspond to prototypes. These prototypes are then used to calculate the ProtoNCE loss in the next iteration.

The ProtoNCE loss is a generalization of the InfoNCE loss that encourages learned representations to be closer to their assigned prototypes. Like InfoNCE, we sample r negative samples or prototypes. To enable encoding of hierarchical semantics, we cluster the samples M times with different numbers of clusters $K = \{k_m\}_{m=1}^{M}$:

$$\mathcal{L}_{\text{ProtoNCE}} = \sum_{i=1}^{n} - \left(\log \frac{\exp\left(v_i \cdot v_i'/\tau\right)}{\sum_{j=0}^{r} \exp\left(v_i \cdot v_j'/\tau\right)} + \frac{1}{M} \sum_{m=1}^{M} \log \frac{\exp\left(v_i \cdot c_s^m/\phi_s^m\right)}{\sum_{j=0}^{r} \exp\left(v_i \cdot c_j^m/\phi_j^m\right)} \right) \tag{1}$$

where $v_i = f_\theta(x_i)$ is the latent representation of query x_i, c_s the centroid of the cluster assigned to x_i, and ϕ_s the concentration of said cluster.

Our goal is to learn representations that capture disease-specific features from the unlabelled dataset so the trained encoder can be used with a classification head for AD classification as the downstream task.

In contrast to ProtoPNet which defines prototypes at the granularity of image patches, we use whole images (3D scans). One reason is, to make predictions on medical images, models often need to look at the entire image [8]. Dividing the image into patches would cause the model to lose a global view of the image.

Another reason is, it has been shown by Wolf et al. [20] that there is no spatial dependency between the latent feature map and the image space: If the patch in the leftmost corner of a latent feature map lights up, it does not necessarily mean that it is highlighting the patch in the leftmost corner of the image. This is because the locality of the latent feature is lost after a few convolutional layers, as the receptive field of the latent feature grows larger. Since the original ProtoPNet assumes this, it does not produce faithful explanations by using patches as prototypes.

Projection of prototypes In the prototype layer g_p, the prototypes learned in the previous stage are projected to the nearest train images of the smaller, labelled dataset. This projection is done by calculating the L_2 distance between each prototype and the latent feature of each image encoded by the momentum encoder. As a result, these images serve as the new prototypes $P = \{p_j\}_{j=1}^{k}$. This stage is important to enable prototype visualization. Furthermore, the projection onto samples of the labelled dataset allows us to determine which label/class each prototype represents.

$$p_j \leftarrow \underset{v \in \mathcal{V}}{\arg\min} \|v - c_j\|_2, \text{ where } \mathcal{V} = \{v_i\}_{i=1}^{n} \tag{2}$$

Training only the fully-connected (FC) head To enable predictions for the downstream task, we append a one-layer FC head h, with weight matrix w_h, to the PCL-pre-trained encoder. The FC head takes in the cosine similarity scores between the query and each prototype (in contrast to the original ProtoPNet, which uses a log activation). It then learns how to weigh these scores accordingly for the prediction of each class.

We keep the encoder and prototypes fixed, and initialize the FC head like ProtoPNet: Let $w_h^{(z,j)}$ be the weight connection between the similarity score of the query with prototype p_j and the logit of class z. We set $w_h^{(z,j)} = 1$ for all prototypes p_j belonging to class z ($p_j \in P_z$), and $w_h^{(z,j)} = -0.5$ for all prototypes p_j not belonging to class z ($p_j \notin P_z$). This initialization ensures that the FC head assigns higher weights to prototypes associated with the correct class and lower weights to those associated with other classes.

Then, the FC head is trained on the smaller, labelled dataset to minimize the following loss:

$$\min_{w_h} \mathbb{E}_{(x,y)\in\mathcal{D}}(h(g_p(f(x))), y) + \lambda \sum_{z=0}^{Z-1} \sum_{j:p_j \notin P_z} |w_h^{(z,j)}| \tag{3}$$

where Z is the number of classes to be predicted and P_z the prototypes of class z. The L_1 regularization aims to make $w_h^{(z,j)} \approx 0$ when $p_j \notin P_z$, so the model does not rely on a negative reasoning process.

2.2 PCL-NW

Another method that we propose is the PCL-NW, which combines PCL [10] and the Nadaraya-Watson (NW) head [17, 18]. Similar to PCL-ProtoPNet, this technique involves pre-training the encoder on the larger, unlabelled dataset using PCL. However, unlike PCL-ProtoPNet, the learned prototypes are discarded afterward as they are not used by the subsequent NW head.

The Nadaraya-Watson head is a non-parametric, non-learnable head that can be attached to an encoder. The encoder and the NW head are trained end-to-end to minimize the Cross-Entropy loss. It provides interpretability by using a **support set** $\mathcal{S} = \{z_i : (x_i, y_i)\}_{i=1}^{N_s}$, comprising N_s samples or images with known labels. These samples are selected to influence the prediction for a query.

Unlike prototypes, which are fixed, the support set is customizable after training. This is beneficial because we can restrict the set only to the desired samples. For example, when we have prior knowledge about the classes the query may belong to, the support set can be restricted solely to samples of these classes.

The prediction for a query x is calculated as the weighted sum of the support labels $\overrightarrow{y_i}$:

$$f(x, \mathcal{S}) = \sum_{i=1}^{N_s} w(x, x_i)\overrightarrow{y_i} \tag{4}$$

The weights are defined as the normalized similarity scores between the latent feature of the query and that of each support sample. They can be interpreted as the influence that each support sample has on the prediction:

$$w_\theta(x, x_i) = \frac{\exp\{-\|g_\theta(x) - g_\theta(x_i)\|_2/\tau\}}{\sum_{j=1}^{N_s} \exp\{-\|g_\theta(x) - g_\theta(x_j)\|_2/\tau\}} \tag{5}$$

where g_θ is the encoder with parameters θ and τ is a temperature hyperparameter. Notice that: In PCL-ProtoPNet, the similarity scores are *weighed*. In PCL-NW, the similarity scores are the *weights*.

The NW head offers both interpretability and calibrated predictions, addressing challenges encountered with the FC head. Additionally, it has the advantage of having no learnable weights. Wang et al. [18] have demonstrated that the NW head can achieve comparable, or even superior, performance to the FC head on various computer vision datasets. In a subsequent work [17], they extended their experimentation to include the application of the NW head on medical images.

3 Experimental Setup

3.1 Dataset

Table 1. Demographic statistics of the UKBB and ADNI datasets. CN: Cognitively Normal. MCI: Mildly Cognitively Impaired. AD: Alzheimer's Disease.

Dataset	Diagnosis	# Samples	% Female	Age
UKBB	-	39,561	52.8%	63.6 ± 7.5
ADNI	CN	379	49.1%	73.5 ± 5.9
	MCI	610	58.5%	72.3 ± 7.3
	AD	256	59.4%	74.5 ± 7.9

To train and evaluate our methods, we choose the Alzheimer's Disease classification problem and use 3D brain MRI images from the UK BioBank (UKBB) and ADNI datasets. The UKBB dataset represents the larger, unlabelled dataset used for self-supervised pre-training using PCL; and the ADNI dataset represents the smaller, labelled dataset for fine-tuning and evaluating the classification performance. Tab. 1 summarizes demographic statistics on both datasets.

UKBB The UK BioBank (UKBB) collects imaging data, including 3D brain MRI images, from predominantly healthy individuals with the aim of tracking their health outcomes over decades to study early disease prediction [11]. Although no labels for Alzheimer's Disease are available, these brain MRI images can be used for self-supervised pre-training of the encoder.

ADNI We obtained our data from the Alzheimer's Disease Neuroimaging Initiative (ADNI) [7] database.[1] The ADNI database was launched in 2004 with the purpose of understanding the development and course of Alzheimer's Disease by collecting various data types, including MRI and PET images, genetic information, cognitive results, and CSF and blood biomarkers. Our subset of the ADNI dataset contains 1,245 samples, with 379 labelled as CN (Cognitively Normal), 610 labelled as MCI (Mildly Cognitively Impaired), and 256 labelled as AD (Alzheimer's Disease).

3.2 Evaluation Strategy

As baselines for evaluating our methods, we train a 3D ResNet18 [5] (implementation from PANIC [21]) on the ADNI dataset with randomly initialized weights. This model, by design, uses the FC head as its final prediction layer. We call this model *ResNet + FC*. Additionally, we train the same model but use the NW head instead of the FC head, and name this model *ResNet + NW*.

We also compare the performance of our methods with models that are trained only with PCL, i.e., models that only consist of an encoder, and whose prototypes are projected, but do not have a head (whether FC or NW). We name this method *PCL + Prototypes*. For inference, we calculate the similarity scores between the query and each prototype and choose the label of the most similar prototype as the predicted label.

Several variations can be made to PCL-ProtoPNet and PCL-NW, and we compare their performance with each other. For PCL-ProtoPNet, we either train or do not train the ProtoPNet-initialized FC head, while keeping the PCL-pre-trained encoder fixed. We call the first approach *PCL-ProtoPNet (train head)* and the second *PCL-ProtoPNet (fix head)*.

We do not consider finetuning the encoder in PCL-ProtoPNet as this would mean re-projecting the prototypes every time the encoder weights are changed. Meanwhile, the FC head requires the prototypes to be fixed because it takes in the similarity scores of the query with the prototypes as input. So, finetuning the encoder would be like chasing a moving target.

Since no prototypes are used in PCL-NW, we can either finetune or not finetune the encoder after the PCL pre-training. We call the first approach *PCL-NW (finetune encoder)* and the second *PCL-NW (fix encoder)*. There is no option for training the NW head as it is non-parametric.

For all models, we train using the PCL method not only on the UKBB dataset but also on the ADNI dataset (without the labels) to observe whether PCL pre-training on UKBB, which is much larger than ADNI, is advantageous.

To report our results, we perform a 5-fold cross-validation by stratifying the ADNI dataset based on sex, age, and diagnosis (label), as done by [21]. Then, each train set is split such that 64% is used as the actual train set and the remaining 16% as the val set. We report the balanced accuracy of the val and test sets (mean ± std by averaging 5 folds).

[1] https://adni.loni.usc.edu/

3.3 Implementation Details

Baselines For training *ResNet + FC*, we carry out a hyperparameter search using Bayesian optimization [15] for 15 runs with the search space: learning rate=(10^{-5}, 10^{-2}) and weight decay=(10^{-5}, 10^{-2}). We train for 100 epochs with a batch size of 64 using the Adam [9] optimizer. We also use the same configuration for training *ResNet + NW*.

PCL pre-training We use DenseNet-121 [6] as the encoder for PCL pre-training on UKBB and 3D ResNet18 on ADNI, as DenseNet-121 performs better on larger datasets and 3D ResNet18 better on smaller.

For PCL pre-training on UKBB, we use the Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.03 and weight decay of 10^{-4}. We train for 80 epochs in total (using the first 10 epochs for minimizing only the InfoNCE loss) with an effective batch size of 256. We use 16 samples or prototypes for negative pairs. We set k=24 and k=96 for k-means clustering but only use the resulting 24 cluster centroids for prototype projection. We choose a latent dimension of 128 to encode the image features.

For PCL pre-training on ADNI, we use the SGD optimizer with a learning rate of 10^{-3} and weight decay of 10^{-4}. We also train for 80 epochs in total (using the first 10 epochs for minimizing only the InfoNCE loss) but with an effective batch size of 64. We use 8 samples or prototypes for negative pairs. We set k=16 and k=24 for k-means clustering but only use the resulting 16 cluster centroids for prototype projection. We choose a latent dimension of 1024 to encode the image features.

In all cases, we use random flipping (p=0.9) and random affine transformation (p=0.9) with rotation range [-90, 90], scaling range [-0.05, 0.05], and translation range [-10, 10] as data augmentation.

PCL-ProtoPNet For training the FC head, we carry out a hyperparameter search using Bayesian optimization [15] for 25 runs with the search space: learning rate=(10^{-5}, 10^{-2}), weight decay=(10^{-5}, 10^{-2}), λ_{L_1}=(10^{-5}, 10^{-2}). We train for 100 epochs with a batch size of 64 using the Adam [9] optimizer.

PCL-NW For finetuning the encoder, we train for 200 epochs using a batch size of 1. We use the SGD optimizer with a learning rate of 10^{-3} and weight decay of 10^{-4} and multi-step scheduler, which decays the learning rate by 0.1 at 100 and 150 epochs. For inference, we include all train samples in the support set. This corresponds to the "Full" inference mode in [18].

4 Results and Discussion

Baselines Comparing *ResNet + FC* and *ResNet + NW*, we observe that the interpretability provided by the NW head comes at the cost of performance since the bAcc decreases by 8% on both val and test sets when using the NW head.

Table 2. Balanced accuracy (bAcc) of Alzheimer's Disease classification on the ADNI validation and test sets. Whether using PCL pre-training or not, all methods are trained on the ADNI dataset for the downstream task. The minus sign ("-") means "not applicable". The numbers highlighted in bold represent the best performance.

Category	Method	PCL dataset	Use head	Train head	Finetune encoder	Val	Test
Baselines	ResNet + FC	-	✓	✓	-	**0.64 ± 0.03**	**0.60 ± 0.03**
	ResNet + NW	-	✓	-	-	0.56 ± 0.04	0.52 ± 0.04
PCL + Prototypes	PCL on UKBB	UKBB	✗	✗	-	0.34 ± 0.03	0.38 ± 0.06
	PCL on ADNI	ADNI	✗	✗	-	0.39 ± 0.04	0.39 ± 0.06
PCL-ProtoPNet	Fix head	UKBB	✓	✗	✗	0.33 ± 0.04	0.38 ± 0.07
	Fix head	ADNI	✓	✗	✗	0.40 ± 0.02	0.40 ± 0.05
	Train head	UKBB	✓	✓	✗	0.50 ± 0.02	0.46 ± 0.05
	Train head	ADNI	✓	✓	✗	0.43 ± 0.02	0.41 ± 0.03
PCL-NW	Fix encoder	UKBB	✓	-	✗	0.44 ± 0.04	0.39 ± 0.03
	Fix encoder	ADNI	✓	-	✗	0.38 ± 0.02	0.39 ± 0.02
	Finetune encoder	UKBB	✓	-	✓	0.57 ± 0.04	0.55 ± 0.03
	Finetune encoder	ADNI	✓	-	✓	0.56 ± 0.05	0.51 ± 0.03

PCL + Prototypes on UKBB vs. ADNI The model pre-trained on UKBB performs 5% worse on the val set than the model pre-trained on ADNI, though not much differently on the test set.

Compared to the baselines, these models perform much worse. This indicates that the prototype closest to a query image often does not actually represent the class the query belongs to. Furthermore, in PCL, we cannot control the number of prototypes that exist for each class. In the extreme case, a class can be presented by no prototypes at all. Consequently, samples belonging to that class will never be classified as that class.

As seen in the t-SNE plots of the latent features of the ADNI samples produced by the encoders of both models (Fig. 2), the prototypes (depicted by the bigger dots) are often surrounded by samples that do not belong to their own class. Meanwhile, prototypes of the same class are scattered across the plot. We also see that overall, the classes are not well-separated. This means PCL failed to learn disease-specific features (CN/MCI/AD) from the UKBB or the ADNI dataset.

PCL + Prototypes vs. PCL-ProtoPNet (fix head) Adding a ProtoPNet-initialized FC head on top of the *PCL + Prototypes* models, without training the head, does not significantly affect performance. This suggests that even though similarity scores to other prototypes are also weighed (taken into account) by the head, the prototype closest to the query has the most influence on the prediction.

PCL + Prototypes vs. PCL-ProtoPNet (train head) Training the FC head while fixing the PCL-pre-trained encoder increases performance dramatically (16% on val, 8% on test) when using the encoder pre-trained on UKBB

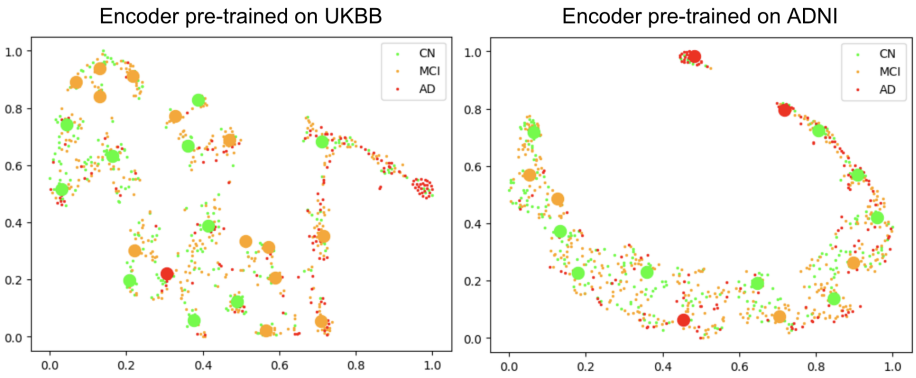

Fig. 2. t-SNE plots of the latent features of ADNI train samples (fold 0) generated by the encoder pre-trained on UKBB and the encoder pre-trained on ADNI using PCL. The prototypes are depicted by the bigger dots.

but only delivers marginal improvements (4% on val, 2% on test) when using the encoder pre-trained on ADNI.

The FC head learns to classify samples using the similarity scores of samples to prototypes. For a particular class, it has to learn to weigh similarity scores of the prototypes appropriately: Prototypes that contribute much to the class will be weighed higher (regardless of the actual label of the prototype), and those that contribute less will be weighed lower.

We hypothesize that the reason the FC head barely improves performance when using the encoder pre-trained on ADNI is that the samples around each of the prototypes produced by this encoder are very diverse (come from more than one class). Hence, during learning, more than one class might "claim" the same prototype by giving it more weight to contribute to the prediction of those classes. Hence, a query with very high similarity score to that prototype will contribute to high probability scores of more than one class. Thus, it becomes harder to distinguish such classes from each other.

Despite improvements from training the head, these models still could not outperform the baselines. This is because as we've seen from the t-SNE plots, the underlying PCL-pre-trained encoder does not really produce disease-specific features. Since the encoder is fixed, the FC head can only do so much to differentiate the samples.

PCL + Prototypes vs. PCL-NW (fix encoder) Discarding the prototypes and adding the NW head to the *PCL + Prototypes* models only improves the performance (10% on val, 1% on test) when using the encoder pre-trained on UKBB. This suggests that compared to the encoder pre-trained on ADNI, this encoder produces features that are more similar to each other for samples belonging to the same class. Hence, these samples can contribute to each other's

predictions better. However, since the improvement only applies to the val set, this is a hypothesis that requires further examination.

PCL + Prototypes vs. PCL-NW (finetune encoder) Finetuning the encoder weights after PCL improves bAcc by 17% (val) and 12% (test) for the encoder pre-trained on ADNI, and by 23% (val) and 17% (test) for the encoder pre-trained on UKBB.

PCL-ProtoPNet (fix head) vs. PCL-NW (fix encoder) These models require no further training after PCL. They differ only in their heads (FC vs. NW) and the usage of prototypes (PCL-NW doesn't use any). In both cases, the encoder is fixed. Compared to using an FC head with the encoder pre-trained on UKBB, using the NW head increases bAcc by 11% on val set, but only 1% on test set. With the encoder pre-trained on ADNI, bAcc degrades by 2% on val set and 1% on test set.

PCL-ProtoPNet (train head) vs. PCL-NW (finetune encoder) These models require further training after PCL. Finetuning the encoder weights has a more positive influence on performance than just training the head since the head's ability to improve depends on the features produced by the encoder. In both cases, PCL pre-training on UKBB yields higher bAcc than on ADNI.

Summary *PCL-NW (finetune encoder)* delivers the best performance among all our proposed methods. In cases where further training is done after PCL, pre-training the encoder on UKBB proves to be more beneficial than on ADNI. Our best method does not improve much from the baseline *ResNet + NW*, meaning that initializing the encoder with PCL pre-trained weights might not be very helpful. Since both methods provide interpretability, directly training *ResNet + NW* on the ADNI dataset seems more advantageous.

5 Conclusion

In this paper, we proposed two methods incorporating the *interpretability* of neural networks and *self-supervised learning* on large, unlabelled datasets — PCL-ProtoPNet and PCL-NW. We evaluated our methods on the Alzheimer's Disease (AD) classification task (CN/MCI/AD). Despite pre-training using PCL on a larger, unlabelled dataset, our methods could not surpass the performance of the baselines which were trained only using supervised learning on the smaller, labelled dataset.

We believe the key to improving performance, while still providing interpretability using prototypes/support set, lies in successfully learning disease-specific features during PCL pre-training. We think the method by Dufumier et al. [2] is promising for our future work, in which we guide the PCL pre-training

using metadata related to the 3D brain MRI scans. In the specific case of AD classification, age as metadata can be especially useful since AD is correlated with older ages.

References

1. Chen, C., Li, O., Tao, C., Barnett, A.J., Su, J., Rudin, C.: This looks like that: Deep learning for interpretable image recognition (2019)
2. Dufumier, B., Gori, P., Victor, J., Grigis, A., Wessa, M., Brambilla, P., Favre, P., Polosan, M., McDonald, C., Piguet, C.M., Duchesnay, E.: Contrastive learning with continuous proxy meta-data for 3d mri classification (2021)
3. Gomolin, A., Netchiporouk, E., Gniadecki, R., Litvinov, I.V.: Artificial intelligence applications in dermatology: Where do we stand? Frontiers in Medicine **7** (2020). https://doi.org/10.3389/fmed.2020.00100, `https://www.frontiersin.org/articles/10.3389/fmed.2020.00100`
4. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning (2020)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
6. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks (2018)
7. Jack, Jr, C.R., Bernstein, M.A., Fox, N.C., Thompson, P., Alexander, G., Harvey, D., Borowski, B., Britson, P.J., L Whitwell, J., Ward, C., Dale, A.M., Felmlee, J.P., Gunter, J.L., Hill, D.L.G., Killiany, R., Schuff, N., Fox-Bosetti, S., Lin, C., Studholme, C., DeCarli, C.S., Krueger, G., Ward, H.A., Metzger, G.J., Scott, K.T., Mallozzi, R., Blezek, D., Levy, J., Debbins, J.P., Fleisher, A.S., Albert, M., Green, R., Bartzokis, G., Glover, G., Mugler, J., Weiner, M.W.: The alzheimer's disease neuroimaging initiative (ADNI): MRI methods. J. Magn. Reson. Imaging **27**(4), 685–691 (Apr 2008)
8. Kim, E., Kim, S., Seo, M., Yoon, S.: Xprotonet: Diagnosis in chest radiography with global and local explanations (2021)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017)
10. Li, J., Zhou, P., Xiong, C., Hoi, S.C.H.: Prototypical contrastive learning of unsupervised representations (2021)
11. Miller, K.L., Alfaro-Almagro, F., Bangerter, N.K., Thomas, D.L., Yacoub, E., Xu, J., Bartsch, A.J., Jbabdi, S., Sotiropoulos, S.N., Andersson, J.L.R., Griffanti, L., Douaud, G., Okell, T.W., Weale, P., Dragonu, I., Garratt, S., Hudson, S., Collins, R., Jenkinson, M., Matthews, P.M., Smith, S.M.: Multimodal population brain imaging in the uk biobank prospective epidemiological study. Nature Neuroscience **19**(11), 1523–1536 (Nov 2016). https://doi.org/10.1038/nn.4393, `https://doi.org/10.1038/nn.4393`
12. Mohammadjafari, S., Cevik, M., Thanabalasingam, M., Basar, A., Initiative, A.D.N.: Using ProtoPNet for Interpretable Alzheimer's Disease Classification. In: Canadian AI 2021. Canadian Artificial Intelligence Association (CAIAC) (jun 8 2021), https://caiac.pubpub.org/pub/klwhoig4
13. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead (2019)
14. Salahuddin, Z., Woodruff, H.C., Chatterjee, A., Lambin, P.: Transparency of deep neural networks for medical image analysis: A review of interpretability methods (2021)

15. Snoek, J., Larochelle, H., Adams, R.P.: Practical bayesian optimization of machine learning algorithms (2012)
16. Sovrano, F., Vitali, F., Palmirani, M.: Making Things Explainable vs Explaining: Requirements and Challenges Under the GDPR, p. 169–182. Springer International Publishing (2021). https://doi.org/10.1007/978-3-030-89811-3$_1$2, http://dx.doi.org/10.1007/978-3-030-89811-3_12
17. Wang, A.Q., Nguyen, M., Sabuncu, M.R.: Learning invariant representations with a nonparametric nadaraya-watson head. NeurIPS (2023)
18. Wang, A.Q., Sabuncu, M.R.: A flexible nadaraya-watson head can offer explainable and calibrated classification (2023)
19. Wang, C., Chen, Y., Liu, Y., Tian, Y., Liu, F., McCarthy, D.J., Elliott, M., Frazer, H., Carneiro, G.: Knowledge Distillation to Ensemble Global and Interpretable Prototype-Based Mammogram Classification Models, p. 14–24. Springer Nature Switzerland (2022). https://doi.org/10.1007/978-3-031-16437-8$_2$, http://dx.doi.org/10.1007/978-3-031-16437-8_2
20. Wolf, T.N., Bongratz, F., Rickmann, A.M., Pölsterl, S., Wachinger, C.: Keep the faith: Faithful explanations in convolutional neural networks for case-based reasoning. Proceedings of the AAAI Conference on Artificial Intelligence **38**(6), 5921–5929 (Mar 2024). https://doi.org/10.1609/aaai.v38i6.28406, https://ojs.aaai.org/index.php/AAAI/article/view/28406
21. Wolf, T.N., Pölsterl, S., Wachinger, C.: Don't panic: Prototypical additive neural network for interpretable classification of alzheimer's disease (2023)
22. Yang, G., Ye, Q., Xia, J.: Unbox the black-box for the medical explainable ai via multi-modal and multi-centre data fusion: A mini-review, two showcases and beyond. Information Fusion **77**, 29–52 (2022). https://doi.org/https://doi.org/10.1016/j.inffus.2021.07.016, https://www.sciencedirect.com/science/article/pii/S1566253521001597