Comparative Study of Interpretable Image Classification Models

Adél Bajcsi*, Anna Bajcsi*, Szabolcs Pável*†, Ábel Portik*, Csanád Sándor* †, Annamária Szenkovits*, Orsolya Vas*, Zalán Bodó*, and Lehel Csató*

***Abstract*—Explainable models in machine learning are increas- ingly popular due to the interpretability-favoring architectural features that help human understanding and interpretation of the decisions made by the model. Although using this type of model – similarly to "robustification" – might degrade prediction accuracy, a better understanding of decisions can greatly aid in the root cause analysis of failures of complex models, like deep neural networks.**

In this work, we experimentally compare three self-explainable image classification models on two datasets – MNIST and BDD100K –, briefly describing their operation and highlighting their characteristics. We evaluate the backbone models to be able to observe the level of deterioration of the prediction accuracy due to the interpretable module introduced, if any. To improve one of the models studied, we propose modifications to the loss function for learning and suggest a framework for automatic assessment of interpretability by examining the linear separability of the prototypes obtained.

***Index Terms*—deep learning, image classification, interpretability, self-explainable models.**

I. Introduction

Explainable artificial intelligence (xAI) is a large set of methods that allows humans to understand the results of a machine learning algorithm [1], [2]. Explainability is defined as the process of making humanly understandable the decision of a machine learning model. Namely, it is the study of relations between the model's decisions and intermediate data representations aimed at better understanding system decisions. Thus, XAI can help us to ensure that the system we built, relying on machine learning algorithms, deep learning, or neural networks, is working as expected. In the literature, there are several techniques for achieving and increasing the explainability of machine learning models, and they differ in their approach and the type of machine learning model used. Among these, the focus will be on self-explainable neural networks, which can be used to increase the transparency of the learning process.

In our work, we studied three self-explainable models: PrototypeDL, ProtoPNet, and BagNet; the primary goal was their evaluation on two datasets: MNIST, which is a dataset of handwritten digits, and BDD100K, which is the largest driving video dataset with 100 000 color images of high resolution. A result of the comparison is the suggestion of possible improvements by examining some components of the models from different aspects, like components of the loss function, and the study of the separability of the prototype vectors that these models use.

In Section I-A we introduce the notion of explainable models in deep learning and in Section I-B we discuss the notion of interpretability in image classification and describe the architecture and operation of our three selected models (PrototypeDL, ProtoPNet, and BagNet). Section I-C contains the experimental comparison of the models, while Section I-D discusses the methods we have used to measure the interpretability of the models and their possible improvements. The concluding Section II enumerates our conclusions from the experiments, as well as the formulation of possible future research directions.

A. Explainable models in deep learning

The increasing popularity of using machine learning models for critical applications like autonomous driving systems or medical diagnosis, suggests an imperative need for methodologies that can help to understand and evaluate the predictions of these models. The main drawback of current state-of-the-art deep neural network models is the lack of reliability and the lack of interpretability of their decisions.

According to a recent overview by [3], *XAI* is a field of AI that aims at providing automated explanations for each decision made by the system. These models can be divided into two types: *post-hoc* and *build-in* methods.

To explain a black-box system, we can start after the training process concluded – in a post-hoc manner: the *linear proxy models* – like e.g. LIME [4] – use local linear models based on the data from the original model (using perturbed inputs). The method can be used to identify the regions of the input that most influence the decision. Decision trees [5] and other rule extraction techniques – like *if-then* rule extraction [6] – increase the transparency of neural networks, however, they are hard to construct.

The concept of *saliency map* was introduced in [7], [8]. The authors created an intensity map illustrating the most/least important pixels or regions used in the computation of the output. Since the training phase is completely independent of the interpretation or explanation stage, the above methods are called post-hoc explainability models.

In this work we focus on self-explainable models for image classification, therefore the system provides visual clues next to the decision. The term self-explainable means that the

[*] Faculty of Mathematics and Computer Science, Babeş-Bolyai University of Cluj-Napoca, Romania

[†] Robert Bosch SRL, Cluj-Napoca, Romania

DOI: 10.36244/ICJ.2023.5.4

explainable part is built in during training. As this domain becomes more important, there were different methods used for explainability. Attention mechanism's [9], [10] main idea is to introduce attention weights over the input sequence to prioritize the set of positions where relevant information is present for generating the next output token. Dosovitskiy et al. [11] introduce the Vision Transformers networks (ViT), inspired by the attention mechanisms [12], where a transformer-based model is used in image classification. When pre-trained on large datasets and transferred to smaller image recognition benchmarks, the model outperforms state-of-the-art CNN networks. Disentangled representations have individual dimensions that describe meaningful and independent factors of variation like variational autoencoder [13], Beta-VAE [14], InfoGAN [15]. Disentangled units can be used to create interpretable CNNs with individual units that detect coherent and meaningful patches instead of difficult-to-interpret mixtures of patterns. Deep networks can also be designed to generate human-understandable explanations as part of the explicit training of the system. In this paper, we highlight three such models and examine their performance and explanations.

B. Interpretability in image classification

According to [3], an *interpretable model* in case of classification details the internals of the decision-making process that is understandable. A model is considered *explainable* when it answers the question "Why the output?". The answer is satisfying when – quote: "one could no longer keep asking why". In the field of image classification, it is achieved by highlighting the region that is likely to be responsible for the prediction.

In this section, three models are presented where the self-explainable part was a feature due to the construction of the model.

1) PrototypeDL: presented in [16], is a self-interpretable neural network architecture for image classification aimed at creating a deep learning architecture that "naturally" explains the reasoning behind each prediction. The architecture contains an autoencoder and a prototype classification network. The classifier network has three different layers: a so-called *prototype layer*, a fully connected layer, and a softmax layer. The encoder reduces the dimensionality of the input and allows making comparisons within the latent space. The decoder restores the encoded input allowing to visualize elements from the latent space – e.g. the learned prototypes. Let $\mathcal{D} = [\boldsymbol{X}, \boldsymbol{Y}] = \{(x_n, y_n)\}_{n=1}^N$ be the training dataset, where $x_n \in \mathbb{R}^p$ and $y_n \in \{1, \ldots, K\}$ denote the predictor and target variables, respectively. The training objective has four terms:

$$\begin{aligned} L_{PDL}(\mathcal{D}) &= \mathrm{CrossEnt}(\boldsymbol{Y}, \widehat{\boldsymbol{Y}}) \\ &+ \lambda_0 \ \mathrm{Rec}(\boldsymbol{X}) \\ &+ \lambda_1 \ \mathrm{R}_1(\mathcal{P}, \boldsymbol{X}) + \lambda_2 \ \mathrm{R}_2(\mathcal{P}, \boldsymbol{X}), \end{aligned} \tag{1}$$

where $\lambda_0, \lambda_1, \lambda_2$ are hyperparameters of the loss function, $\mathrm{CrossEnt}(\boldsymbol{Y}, \widehat{\boldsymbol{Y}})$ is the cross-entropy – the classification – error function, $\mathrm{Rec}(\boldsymbol{X})$ is the autoencoder's reconstruction error:

$$\mathrm{Rec}(\boldsymbol{X}) = \frac{1}{n} \sum_{i=1}^{n} \|(\boldsymbol{\Phi} \circ \boldsymbol{\Phi}')(x_i) - x_i\|_2^2,$$

with $\boldsymbol{\Phi}(\cdot)$ and $\boldsymbol{\Phi}(\cdot)'$ the encoder and decoder functions respectively, and the pair R_1 and R_2 are costs for the quality of prototypes:

$$\begin{aligned} \mathrm{R}_1(\mathcal{P}, \boldsymbol{X}) &= \frac{1}{M} \sum_{m=1}^{M} \min_{n=1,N} \|\boldsymbol{p}_m - \boldsymbol{\Phi}(x_n)\|_2^2, \\ \mathrm{R}_2(\mathcal{P}, \boldsymbol{X}) &= \frac{1}{N} \sum_{n=1}^{N} \min_{m=1,M} \|\boldsymbol{\Phi}(x_n) - \boldsymbol{p}_m\|_2^2. \end{aligned}$$

In the above formula $\mathcal{P} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_M\}$ is the set of prototype vectors, each vector corresponding to a *prototype unit* in the architecture; M is the number of prototypes and N is the size of the training dataset. These two terms encourage that (1) every prototype to be close to at least one training example ensuring the existence of a meaningful prototype, and that (2) every training example to be close to at least one prototype; yielding a clustering of the training examples around the prototypes.

In contrast to ProtoPNet, the next model to be presented in Section I-B2, PrototypeDL produces full-size prototypes, i.e. of the same size as the encoded images. As a drawback, it fails to produce realistic/interpretable images when trained on natural pictures, due to the autoencoder used as a feature extractor. The decoder will return blurry images like Figure 1, which can not be used as explanations.

2) ProtoPNet: The prototypical part network (ProtoPNet), introduced in [17], is an improvement over the PrototypeDL architecture from Section I-B1. Instead of using an autoencoder to obtain the features, the model includes as a backbone a convolutional network for classification, such as VGG-16, VGG-19, ResNet-152, DenseNet-121, or DenseNet-161 [18]. The "convolutional" part of the above-mentioned backbone networks will serve as prototypes: the system considers the last output layer as a set of features. After extracting the *set* of features is followed by a prototype layer and the fully connected layer that performs multi-class classification. If we assume that the output of the convolution is of shape (H, W, D), then we consider every (H_1, W_1, D) patch a prototype – in practice, we will use $(1, 1, D)$. If we consider the output as a representation of the input image, the prototypes will correspond to a patch (of non-uniform size) from the original image. The prototypes (denoted by $\mathcal{P}$) will have the same size as these convolutional patches, i.e. of (H_1, W_1, D), and each class is assigned a fixed number of prototypes. For a given input image, the j-th prototype unit in the prototype layer computes the Euclidean distance between the j-th prototype vector and all the patches of the convolutional output, resulting in a heatmap that shows which parts of the input image are most similar to the prototype. These maps are reduced to a single similarity score for each prototype vector using global max pooling; the reduced scores can be interpreted as to which extent the given prototype is present in the input image.

The loss function of ProtoPNet can be separated into three parts:

$$L_{PPN}(\mathcal{D}) = \text{CrossEnt}(\boldsymbol{Y}, \widehat{\boldsymbol{Y}}) + \lambda_1 \text{Clst}(\mathcal{P}, \boldsymbol{X}) + \lambda_2 \text{Sep}(\mathcal{P}, \boldsymbol{X}), \tag{2}$$

where $\text{CrossEnt}(\boldsymbol{Y}, \widehat{\boldsymbol{Y}})$ is again cross-entropy, $\text{Clst}(\mathcal{P}, \boldsymbol{X})$ is a clustering term ensuring that images of a given class have at least one latent patch close to a prototype of the same class, while $\text{Sep}(\mathcal{P}, \boldsymbol{X})$ pushes the latent patches apart from the prototypes of other classes:

$$\text{Clst}(\mathcal{P}, \boldsymbol{X}) = \frac{1}{N}\sum_{n=1}^{N} \min_{\boldsymbol{p}_j \in \boldsymbol{P}_n} \min_{\boldsymbol{z} \in \text{patches}(\boldsymbol{\Phi}(\boldsymbol{x}_n))} \|\boldsymbol{z} - \boldsymbol{p}_j\|_2^2,$$
$$\text{Sep}(\mathcal{P}, \boldsymbol{X}) = -\frac{1}{N}\sum_{n=1}^{N} \min_{\boldsymbol{p}_j \notin \boldsymbol{P}_n} \min_{\boldsymbol{z} \in \text{patches}(\boldsymbol{\Phi}(\boldsymbol{x}_n))} \|\boldsymbol{z} - \boldsymbol{p}_j\|_2^2,$$

where $\boldsymbol{P}_n \subset \mathcal{P}$ is the set of prototypes corresponding to class $\boldsymbol{y_n}$ and $\boldsymbol{\Phi}(\cdot)$ denotes the convolutional backbone.

ProtoPNet, as opposed to PrototypeDL, can be used on real-world images. The prototype-to-image is explicit and uses images from the training set, this is the $\text{Clst}(\cdot)$ term in the error function. The anchoring to a given input image and the "localized footprint" of the prototype vector leads to an increased interpretability level – contrasting PrototypeDL, where we have global representations of the inputs.

3) BagNet: these models were introduced by Brendel and Bethge [19], inspired by the *bag-of-words* (BoW), or the *bag-of-features* (BoF) models; these are popular models in information retrieval (IR) and natural language processing (NLP) [20]. BoW or BoF count the occurrence of a feature in an entity, e.g. words in a document, thus the representation of a document becomes a bag data structure collecting the number of word occurrences in a document. The architecture of BagNet resembles BoW: individual features of the input are put together – in this case, averaged – to obtain the global representation of the entire image.

The idea is simple yet effective: we use an FCN to generate features for the input image and use the average of the feature vectors to output the logits. Each feature vector corresponds to a window of the same size as the receptive field of the network (patch). Using logits, the output of the linear classifier will be the same as the average of the logits output for each input image patch.

BagNet is able to generate a heatmap for the input images: for each patch of the input image the model outputs the class logits and these will correspond to one pixel of the heatmap. These heatmaps can be interpreted as explanations for the classification: the part of the image, where the heatmap has a high activation has more importance during classification. Moving the window with stride 1, we are able to generate one heatmap for each class.

This model is built on a simple idea inspired from IR/NLP, yet we found that it produces competitive results for self-explaining image classification tasks.

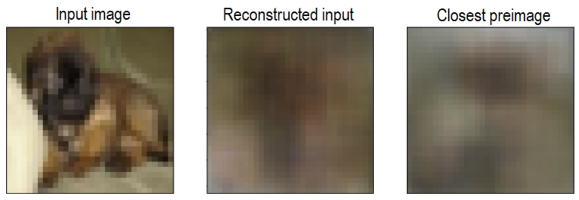


Fig. 1. The PrototypeDL architecture in action on the CIFAR10 data: the input (left), its reconstruction (middle), and the closest pre-image (right). It can be observed that both the reconstruction and the pre-image are blurry, therefore cannot serve as an explanation.

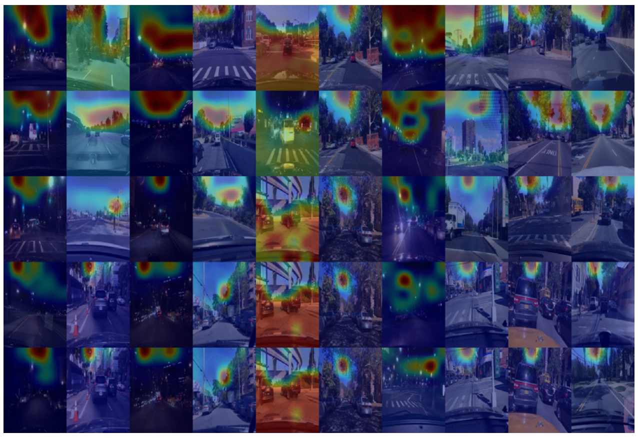


Fig. 2. Evolution of the 10 prototypes assigned to *clear* class – from top to bottom. It is visible that most of the prototypes "focus" on the upper part of the image, the blue sky. There are abnormal behaviours, e.g. in the fifth column the region is not "interpretable" w.r.to the class semantics.

C. Experimental comparison of the models

In our research, we conducted several experiments using different datasets. First, we used a simpler dataset (MNIST – Modified National Institute of Standards and Technology[1] [21]) and then the models were fine-tuned for a more complex one (BDD100K – Berkeley DeepDrive[2] [22]). Working with BDD100K the *weather condition* labels were used, containing 7 classes: *clear*, *foggy*, *overcast*, *partly cloudy*, *rainy*, *snowy*, and *undefined*. This dataset is highly unbalanced, e.g. we have 37 344 images with label *clear* as opposed to 130 images labeled *foggy*.

1) PrototypeDL: In the original paper [16] this method was tested on three datasets: MNIST (99.22% test accuracy), 3D cars [23] (93.5% test accuracy), and Fashion MNIST[3] (89.95% test accuracy), the obtained results being comparable with that of non-interpretable models (within 2.55% margin). We also tested the model on CIFAR10 dataset, but the output prototypes were not interpretable, see Figure 1. Due to its poor interpretability, this model was not tested on BDD100K dataset.

2) ProtoPNet: In our experiments, VGG-19 CNN was used as a feature extractor. The number of prototypes per class was

[1]http://yann.lecun.com/exdb/mnist/
[2]http://bdd-data.berkeley.edu
[3]https://github.com/zalandoresearch/fashion-mnist

TABLE I
COMPARISON OF THE PERFORMANCE OF PROTOPNET AND THE EQUIVALENT BACKBONE MODEL ON BDD100K DATASET.

Metric	**ProtoPNet**		**Backbone model**	
	Train	Test	Train	Test
Accuracy	0.8564	0.8264	0.8457	0.8284
F_1 score (micro-avg.)	0.8523	0.8218	0.8457	0.8284
F_1 score (macro-avg.)	0.7082	0.6422	0.6725	0.6510

TABLE II
COMPARISON OF THE PERFORMANCE OF BAGNET-17 AND THE EQUIVALENT RESNET-50 ON THE BDD100K DATASET.

	BagNet-17		**Equivalent ResNet-50**	
	Train	Test	Train	Test
F_1 score (micro-avg.)	0.7880	0.7953	0.8098	0.8093
F_1 score (macro-avg.)	0.5973	0.6038	0.6494	0.6242
Training epochs	81		84	

set to 10 as in [17]. Besides trying to reproduce the result on CUB-200-2011[4] the model was trained on MNIST and BDD100K datasets. We analyzed the result of our experiments from two aspects. Firstly, the performance of ProtoPNet was compared against its equivalent backbone, here we aimed at assessing the cost of interpretability. Secondly, we performed a subjective analysis of the prototypes and their evolution over the learning epochs.

The backbone model has a feature extractor layer followed by a fully connected one, i.e. the prototype layer was removed from the model. On MNIST dataset the differences were below 0.1%, both interpretable and non-interpretable models had an accuracy of 99.8% on the train and 99.6% on test data. With BDD100K dataset the interpretable model achieved better accuracy on train data, as shown in Table I. We also measured F_1 scores on the BDD100K dataset because of its uneven distribution. We can observe a gap between the models on train data by analyzing these scores. For every class, we collected the images from which the ProtoPNet learned its 10 prototypes in different epochs and aggregated them into an image collage. In the original images, we can see the saliency map of input pixels indicating their contribution to the learned prototype vector. The prototypes were observed form every 10th epoch up to the 50th. Figure 2 shows an example for the class *clear*. As expected, most of the learned prototypes are parts from the blue sky regions. The evolution of prototypes is noticeable (with uninterpretable prototypes at the start). Using visual inspection, we can have a subjective assessment of the prototype, namely the region it focuses on during prediction. This way we may exclude prototypes from the model. For example, in the 5th column, there is a prototype of the road. If we want our model to “focus” only on the sky, we can remove it from the model.

3) BagNet: In [19] the model was tested on ImageNet[5]. Using a 17×17 patch size, the model reached the performance of AlexNet (80.5% top-5 accuracy) [24], while using 33×33 patches a top-5 accuracy of 87.6% was achieved.

[4]https://www.vision.caltech.edu/datasets/cub_200_2011/
[5]https://www.image-net.org/

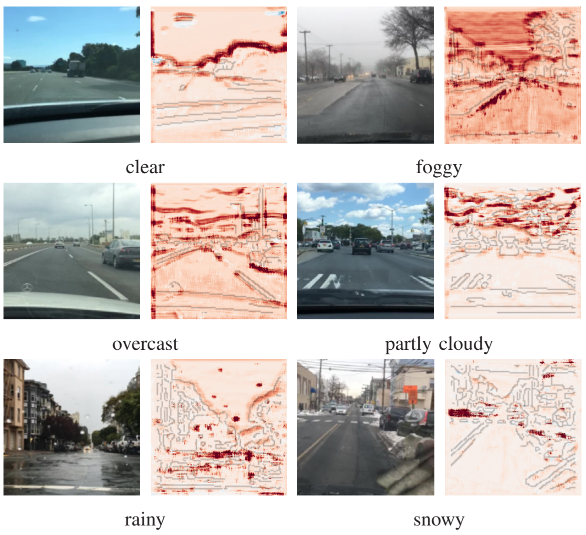


Fig. 3. One sample from each weather class from the BDD100K dataset (left) and their corresponding heatmaps (right) obtained with the BagNet-17 model. The darker the region, the more important it was in assigning the given label to the image.

For conducting the experiments on the BDD100K dataset, we used the initial architecture, based on ResNet-50 [25] by replacing most of the 3×3 by 1×1 convolutions, thus limiting the receptive field size of the topmost convolutional layer to $q \times q$. We experimented with both $q = 9$ and $q = 17$ and found that $q = 17$, that is Bagnet-17, yielded better results.

We compared the results obtained with the BagNet-17 with the corresponding ResNet-50 architecture (backbone model). The model was trained using SGD with momentum (0.9), a batch size of 16, and a learning rate initially set to 0.1 and decayed by a multiplicative factor of 0.35 every 30 epochs. For evaluation, F_1 score was used. The results are summarized in Table II, which shows that BagNet-17 achieved a micro-averaged F_1 score of 0.7880 on the training set and a micro-averaged F_1 score of 0.7953 on the test, respectively, after 81 epochs. Surprisingly, the equivalent ResNet-50 has outperformed the BagNet-17 by only 0.014 on the test set when evaluated with the micro-averaged F_1 score, and by 0.0204 with the macro F_1 score.

To visually assess the interpretability of the BagNet-17 model on BDD100K, we plotted the heatmaps generated by the model, as shown in Figure 3 along with the original images. Our conclusion is that the strong emphasis is in partial agreement with our intuition but it is often not conclusive: e.g it is unclear why in the clear class example – top left – the highest importance is rendered to the line separating the blue sky background from the rest of the image.

D. Measuring interpretability

While measuring prediction accuracy is usually simple, unless labeling data becomes a costly process [26], determining interpretability is much more complicated, however, the assessment of explainability is similarly important to evaluate the method and compare it to other approaches from this perspective as well. Following the works [27]–[29] we define

TABLE III
EVALUATION RESULTS OBTAINED ON MNIST (RESNET-20 BACKBONE) USING DIFFERENT HYPERPARAMETERS.

	λ_1	λ_2	λ_3	**Training accuracy**	**Test accuracy**
(a)	0.8	-0.08	0	0.9902	0.9814
(b)	0.8	0	0	0.9951	0.9853
(c)	0.8	-0.08	-0.1	0.9895	0.9826
(d)	0.8	-0.08	-0.15	0.9893	0.9808

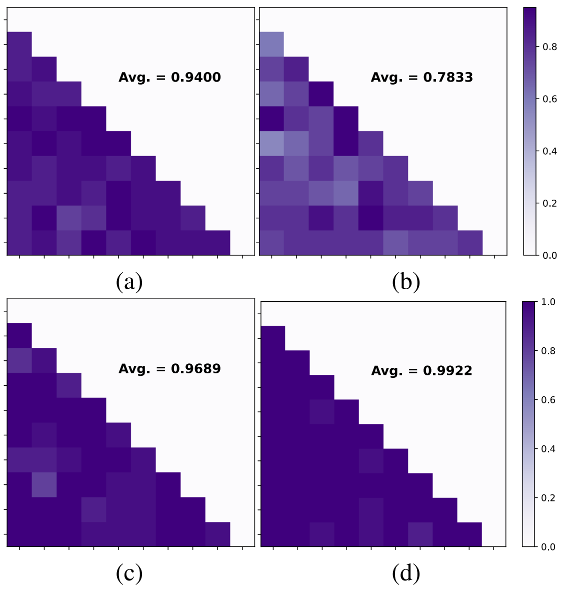


Fig. 4. Matrix of 1-vs-1 average training accuracies obtained using linear SVMs to separate prototype vectors assigned to different classes (the four scenarios correspond to the models displayed in Table III): (a) original ProtoPNet model, (b) omitting the separation cost, (c)–(d) using the additional separation cost.

interpretability as the linear separability of the feature vectors and incorporate a new separation cost to improve on this in the ProtoPNet model[6]. Thus, we measure interpretability by the macro-averaged 1-vs-1 training classification accuracy obtained by a linear classifier considering the prototype vectors assigned to different classes as training examples. In our experiments, linear support vector machines (SVM) [30] were used.

In [17], the separation cost enforces every latent patch of a training image to be distant from the prototypes not of its class, which is a requirement of explainability. If no such condition is imposed on the patches and/or prototypes it may harm the interpretability of the model, since similar prototypes can be obtained for different classes. While discarding separability may improve classification performance [31], it can hurt the visual explanation process.

The additional separation cost introduced in our model pushes away the prototype vectors from each other,

$$\mathrm{Sep}_2(\mathcal{P}) = \frac{-2}{K(K-1)} \sum_{\substack{i,j=1 \\ j>i}}^{K} \min_{\substack{\boldsymbol{p}\in\mathcal{P}_i, \\ \boldsymbol{q}\in\mathcal{P}_j}} \|\boldsymbol{p}-\boldsymbol{q}\|_2^2 \tag{3}$$

[6]The linear separability of the features increases/should increase monotonically with the depth of the model, explained by the fact that the last layer of deep neural networks is usually a linear classifier.

and thus the total loss function of the optimization problem becomes

$$\begin{aligned} L(\mathcal{D}) = {} & \mathrm{CrossEnt}(\boldsymbol{Y}, \widehat{\boldsymbol{Y}}) \\ & + \lambda_1 \mathrm{Clst}(\mathcal{P}, \boldsymbol{X}) \\ & + \lambda_2 \mathrm{Sep}(\mathcal{P}, \boldsymbol{X}) + \lambda_3 \mathrm{Sep}_2(\mathcal{P}), \end{aligned} \tag{4}$$

where CrossEnt, Clst, and Sep represent the same loss components as in the original ProtoPNet model. In Figure 4 the interpretability values are shown as macro-averaged 1-vs-1 training accuracies obtained for separating the prototypes of different classes using a linear SVM. We also display the lower triangular portion of the confusion matrices obtained, where a darker color of a square denotes a better separation. For conducting the experiments, ResNet-20 was used as the backbone of ProtoPNet [25], and the methods were evaluated on the MNIST dataset. The number of epochs was set to 50. Table III shows the hyperparameter settings of the models together with the training and test accuracies obtained. From Table III and Figure 4 one observes that by omitting the separation costs, it is possible to obtain better accuracy scores (model (b)) compared to the base model (model (a)), meanwhile, separability decreases. The introduction of the new separation cost yields slightly better test accuracies and better linear separation of the prototypes (model (c)). Furthermore, setting a higher weight for the newly introduced cost (model (d)) can slightly decrease prediction accuracy but at the same time significantly increase the separability.

II. CONCLUSION

In this article, we presented a comparison of three interpretable image classification models. All models use convolutional neural networks, CNNs, to represent image features via prototype vectors, and the use of prototypes provides a good performance. The interpretability of the models is achieved via the connection of the prototype vectors to the outputs on one hand, and the connection of the prototype vectors to a region in the input image on the other hand; model interpretability is achieved by linking the decision – i.e. the model output – and the region in the input image.

With the measurements on “backbones”, we assessed whether the addition of interpretability to the model, similar to extensions towards robustness, impacts accuracy and we conclude that there is no significant drop in performance when interpretability is added to the model. The conclusion is that it was essential to properly adjust model hyperparameters, such as receptive field size or the parameters of the cost function; this fitting of model parameters indicates a potential instability when using the models in real situations.

As a summary, we assessed that the first tested *PrototypeDL* model uses prototypes that are global to the whole image, therefore the pre-images of the prototypes are blurry, as such cannot be used as an explanation. Our second model, *ProtoPNet*, an improved PrototypeDL, provides localized prototypes, and the manual inspection of the results confirms its highly interpretable nature, and this model achieves the best evaluation results on the tested datasets. The third model we tested, the BagNet model, had the simplest architecture of all and it produces good results despite not using prototypes.

An important conclusion was that these models need large datasets to perform well – namely, we emphasize that for ProtoPNet the multiplication of inputs using augmentation had a huge positive impact on the performance.

Since almost all methods studied so far rely on deep features extracted by CNNs, we might ask about the validity of our base assumption about these, namely that similar patches generate similar features. As one-pixel attacks show [32], it suffices to change one pixel in the input image to obtain very dissimilar features. At the same time, other obfuscation techniques, e.g. JPEG compression, resulting in differences imperceptible to the human eye greatly influence the output of the network as well [33]. Therefore, we plan to study the influence of robustness on the predictions and explanations in interpretable methods [34], as well as explore other approaches that could increase interpretability in such self-explainable deep neural network models.

Acknowledgment

The authors acknowledge the financial support through Bosch grant *"Explainable artificial intelligence (xAI) for automated driving systems"*.

References

[1] O. O'Neill, "Linking trust to trustworthiness," *International Journal of Philosophical Studies*, vol. 26, no. 2, pp. 293–300. DOI: 10.1080/09672559.2018.1454637, 2018.

[2] J. A. McDermid, Y. Jia, Z. Porter, and I. Habli, "Artificial intelligence explainability: the technical and ethical dimensions," *Philosophical Transactions; Series A*, vol. 379. DOI: 10.1098/rsta.2020.0363, 2021.

[3] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," in *DSAA*. IEEE, 2018, pp. 80–89. DOI: 10.1109/DSAA.2018.00018

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should I trust you?" Explaining the predictions of any classifier," in *KDD*, 2016, pp. 1135–1144. DOI: 10.1145/2939672.2939778

[5] T. Hastie, R. Tibshinrani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Verlag, 2016.

[6] S. Thrun, "Extracting rules from artificial neural networks with distributed representations," in *Advances in Neural Information Processing Systems*, G. Tesauro, D. Touretzky, and T. Leen, Eds., vol. 7. MIT Press, 1994, pp. 505–512.

[7] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Springer International Publishing, 2014. ISBN 978-3-319-10590-1 pp. 818–833. DOI: 10.1007/978-3-319-10590-1_53

[8] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *Workshop at International Conference on Learning Representations*, 2014, pp. 1–8. DOI: 10.48550/arXiv.1312.6034

[9] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," arXiv preprint *arXiv:1409.0473*, 2015.

[10] S. Chaudhari, V. Mithal, G. Polatkan, and R. Ramanath, "An attentive survey of attention models," *ACM Transactions on Intelligent Systems and Technology*, vol. 12, no. 5, pp. 1–32. DOI: 10.1145/3465055, 2021.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," arXiv preprint *arXiv:2010.11929*, Tech. Rep., 2020.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, vol. 30, 2017. DOI: 10.48550/arXiv.1706.03762

[13] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013.

[14] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-VAE: Learning basic visual concepts with a constrained variational framework," in *ICLR*. OpenReview.net, 2017.

[15] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *NeurIPS*, vol. 29, 2016. DOI: 10.48550/arXiv.1606.03657

[16] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions," in *AAAI*, vol. 32, 2018, pp. 3530–3537. DOI: 10.1609/aaai.v32i1.11771

[17] C. Chen, O. Li, C. Tao, A. J. Barnett, J. Su, and C. Rudin, "This looks like that: deep learning for interpretable image recognition," in *NeurIPS*, 2019, pp. 8930–8941. DOI: https://dl.acm.org/doi/10.5555/3454287.3455088

[18] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. ISBN 9780262035613. [Online]. Available: http://www.deeplearningbook.org

[19] W. Brendel and M. Bethge, "Approximating CNNs with bag-of-local-features models works surprisingly well on ImageNet," 2019.

[20] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge University Press, Cambridge, 2008. [Online]. Available: https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf

[21] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: Extending MNIST to handwritten letters," in *IJCNN*, 2017, pp. 2921–2926. DOI: 10.1109/IJCNN.2017.7966217

[22] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "BDD100K: A diverse driving video database with scalable annotation tooling," 2018. [Online]. Available: https://www.arxiv-vanity.com/papers/1805.04687

[23] S. Fidler, S. Dickinson, and R. Urtasun, "3D object detection and viewpoint estimation with a deformable 3D cuboid model," in *NeurIPS*, vol. 25. Curran Associates, Inc., 2012. DOI: 10.13140/2.1.2839.7440

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84–90. DOI: 10.1145/3065386, 2017.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*. IEEE, 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90

[26] X. J. Zhu, "Semi-supervised learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep. 1530, 2005.

[27] G. Alain and Y. Bengio, "Understanding intermediate layers using linear classifier probes," in *ICLR*, 2017. DOI: 10.48550/arXiv.1610.01644

[28] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *CVPR*. IEEE, 2017, pp. 6541–6549. DOI: 10.1109/CVPR.2017.354

[29] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV)," in *ICML*, 2018, pp. 2668–2677. DOI: 10.48550/arXiv.1711.11279

[30] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *COLT*, 1992, pp. 144–152. DOI: 10.1145/130385.130401

[31] W. Xiao, Z. Ding, and H. Liu, "Learnable visual words for interpretable image recognition," *CoRR*, vol. abs/2205.10724. DOI: 10.48550/arXiv.2205.10724, 2022.

[32] D. V. Vargas and J. Su, "Understanding the one-pixel attack: Propagation maps and locality analysis," 2019.

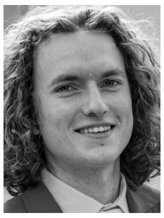
[33] A. Hoffmann, C. Fanconi, R. Rade, and J. Kohler, "This looks like that... does it? shortcomings of latent space prototype interpretability in deep networks," 2021.

[34] C. Etmann, S. Lunz, P. Maass, and C. Schönlieb, "On the connection between adversarial robustness and saliency map interpretability," in 6 ICML 2019, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. *PMLR*, 2019, pp. 1823–1832. https://doi.org/10.48550/arXiv.1905.04172

Adél Bajcsi is a Ph Dstudent at the Faculty of Mathematics and Computer Science, Babeş-Bolyai University, under the supervision of Camelia Chira. Her current research interest includes image processing and classification, and machine learning. Adél is currently a member of a research team that investigates different self-interpretable deep learning models applied in the field of image recognition.

Anna Bajcsi is currently a member of a research group at the Babeş-Bolyai University investigating self-explainable deep learning models for image classification problems. She obtained her master's degree at the same university in 2022 studying Data Analysis and Modelling.

Szabolcs Pável is an assistant professor at the Faculty of Mathematics and Computer Science of the Babeş-Bolyai University (Cluj, Romania). His research interests include classical computer vision, machine learning (including deep learning), focusing on applications in driver assistance systems and self-driving cars. He also works as a machine learning engineer in the automotive industry, developing video perception systems.



Ábel Portik is a master's student in Data Analysis and Modelling at the Faculty of Mathematics and Computer Science, Babeş-Bolyai University. He currently investigates self-explaining deep neural models applied for image classification as a member of a research group.

Csanád Sándor is an assistant professor at the Faculty of Mathematics and Computer Science, Babeş-Bolyai University, as well as a machine learning engineer in the automotive industry. His main research interest is neural network compression, focusing on structured parameter pruning and post-training quantization.

Annamária Szenkovits is an assistant professor at the Department of Math- ematics and Computer Science of the Hungarian Line of the Babeş-Bolyai University (Cluj, Romania). Her main areas of interest include information retrieval and machine learning. Within machine learning, her work focuses on image recognition and natural text processing. She is currently a member of a research team that investigates different self-explainable deep learning models applied in the field of image recognition.

Orsolya Vas is an assistant professor at the Faculty of Mathematics and Computer Science of the Babeş-Bolyai University of Cluj-Napoca. Her previous research interests and results are related to critical point analysis and differential calculus. She is a member of a research team investigating self-explainable deep learning models for classification.

Zalán Bodó is currently an associate professor at the Faculty of Mathematics and Computer Science, Babeş-Bolyai University, teaching, among other subjects, Information Theory and Natural Language Processing. He obtained his PhD degree from the same faculty in 2010, studying and analyzing kernel methods for semi-supervised learning. Since then, he participated in several research projects, and his main interests include data-efficient machine learning algorithms, natural language processing, information retrieval, and recently deep learning methods.

Lehel Csató is a professor at the Faculty of Mathematics and Computer Science. He holds a PhD from Aston University (UK), his research was the use of probabilistic non-parametrics as latent variables. He is interested in the mathematical aspects of machine learning, in providing approximate solutions for inverse problems, performing sparse inference on large systems with applications to robotics, classification of complex data, and active learning. After the onset of the new deep learning era, he is interested in a better understanding of the working of these systems; the exploration of self-explainable deep learning methods, and simplification possibilities, e.g. pruning.