# Interpretable Image Classification via Deep Supervised Clustering

**Marius Arvinte** [1] **Mai Lee Chang** [1] **(Ethan) Yuqiang Heng** [1]

## Abstract

Deep neural network classifiers have recently surpassed all other previous models, as well as humans, in terms of image classification and object recognition. In this work, we propose an interpretable classification architecture that combines the expressive power of a deep neural network with an ensemble of simple, geometrically motivated and interpretable classifiers. We use a deep network to shape the data to a clustered structure, with one cluster per class and then train a set of one-dimensional classifiers based on distance embeddings to the centers of each cluster. We experimentally show that we can replace each classifier with a simple decision set and that we can generate explanations for each classification output, since the distance embeddings are with respect to representative points from each class.

## 1. Introduction

Deep learning models have the potential to make positive societal impacts in various domains including medicine, criminal justice, education, and finance. However, their main drawback is that they are considered "black boxes". For instance, recent work in theoretically explaining how two-layer deep neural networks (DNNs) with ReLU activations work resulted in over a hundred theorems (Song et al., 2018). While this is one way to understand DNNs, this method is not eye-opening for the majority of users of DNN-based products since these domain experts do not come from machine learning backgrounds. One solution is to make models interpretable. Interpretability is defined as the mapping of an abstract concept (e.g., a predicted class) to a domain that the person can make sense of, such as images or sequences of words (Montavon et al., 2018). For classification, an interpretable model enables the user to understand the decision boundaries between classes and why the model made a certain prediction. This understanding is crucial as it will help domain experts appropriately trust and use the model (Lakkaraju et al., 2016). The challenge with building interpretable models is that accuracy and interpretability are competing goals.

Our work is motivated by interpretability and proposes a holistic approach that combines four aspects: algorithmic transparency, decomposability, case-based reasoning and rule-based reasoning. The work in (Li et al., 2018) proposes the use of a prototype layer that takes an input vector and returns the set of distances to a trainable collection of vectors. In (Chen et al., 2018), the authors take a step further and bind each prototype vector to a specific class.

We reuse the core concept of prototypes in (Li et al., 2018), but propose a novel, interpretable classifier architecture that combines supervised clustering with a geometrically formulated rule-based reasoning classification scheme, as well as an ensemble of experts method. We propose a criteria for deciding online (i.e., during training) which class to add a new prototype vector for, as well as how to find such a vector that is close to or belongs to the training data. Finally, after training is completed, we show that our classifier can be completely replaced by an ensemble of decision sets, allowing the user to generate semantic explanations for why a particular image is classified in a specific way. Note that by supervised clustering we do not mean that the deep network has access to the label of the input, but, rather, it is trained to produce a latent representation that is implicitly clustered according to the label, exactly as in (Makhzani et al., 2015).

The proposed method has the advantage of a decomposable structure, avoiding black-box like training methods as in (Wu & Tabak, 2017). We separate the clustering network from the prototype learning stage and restrict the prototype vectors to exactly match points in the training dataset, thus solving the issue of interpretability. We demonstrate the effectiveness of our method on carefully designed synthetic data, as well as images of handwritten images and real-world objects and show that the method is consistent even when the data is not perfectly clustered or even when there is strong overlap between different clusters corresponding to different classes.

[1]University of Texas at Austin. Correspondence to: Marius Arvinte <arvinte@utexas.edu>, Mai Lee Chang <mlchang@utexas.edu>, (Ethan) Yuqiang Heng <yuqiang.heng@utexas.edu>.

*Proceedings of the 36$^{th}$ International Conference on Machine Learning*, Long Beach, California, PMLR 97, 2019. 

## 2. Related Work

Interpretability broadly falls into two general types of methods, posthoc and built-in the model itself (non-posthoc). For posthoc interpretability, a trained model is given, and the goal is to understand how the model performs classification, without the possibility of modifying the model. One approach is activation maximization where the objective is to find inputs that maximize the outputs at different layers (Erhan et al., 2009; Lee et al., 2009; Oord et al., 2016; Yosinski et al., 2015). Other approaches used for image classification are deconvolution (Zeiler & Fergus, 2014) and gradient-based saliency visualization (Simonyan et al., 2013; Selvaraju et al., 2017). However, the main limitations of posthoc interpretability are that the explanations are only one of many possible reasons of how the model made its predictions and may not be the actual reasons, as well as the fact that extra modeling is required (Montavon et al., 2018).

Non-posthoc interpretability methods can be based either on extractive reasoning, case-based reasoning or rule-based reasoning. In extractive reasoning approaches, parts of an input are extracted and used for prediction (Pinheiro & Collobert, 2015; Lei et al., 2016) whereas in case-based reasoning, explanations are based on global similarity to prototypes (Bien et al., 2011; Kim et al., 2014; Branson et al., 2014; Wu & Tabak, 2017; Li et al., 2018; Chen et al., 2018). Here, a prototype is defined as being very similar to or identical to a training data point. Rule-based reasoning methods include decision trees and decision sets (Rivest, 1987; Breiman, 2017; Lakkaraju et al., 2016). The authors of (Lakkaraju et al., 2016) define and characterize a decision set by its size, length, cover, and overlap. They minimize size, length and overlap, and maximize cover to achieve interpretability. They show that for classification, decision sets are more interpretable in comparison to decision list since each rule can be applied independently. In addition, an ensemble of classifiers also uses rule-based reasoning such as majority vote and average vote (Dietterich, 2000; Rokach, 2010; Ruta & Gabrys, 2005).

A different perspective of interpretability is to look at it from the level of the entire model (simulatability), individual model components (decomposability), and training algorithm (algorithmic transparency) (Lipton, 2016). Poursabzi-Sangdeh et al. showed that a clear model, i.e., transparent model, is more understandable to participants (Poursabzi-Sangdeh et al., 2018). In our design of an interpretable model, we also incorporate decomposability and algorithmic transparency.

## 3. Proposed solution

Our approach consists of three components that are completely decoupled: deep supervised clustering, distance embedding, and an ensemble of interpretable classifiers. The ensemble of interpretable classifiers can be replaced by a decision set later. A high-level block diagram of our approach is shown in Figure 1.

Consider a high-dimensional, labeled dataset $\mathbf{x}_i$ and a low-dimensional, clustered (latent) representation of it $\mathbf{z}_i$, where each cluster only contains point with a single label. Let $d(\mathbf{a},\mathbf{b})$ be a quasi-metric, possibly not satisfying the triangle inequality. We define a distance embedding of a point $\mathbf{a}$ with respect to a point $\mathbf{b}$ as the function

$$f_{\mathbf{b}} : \mathbb{R}^{\mathrm{K}}, \quad f_{\mathbf{b}}(\mathbf{a}) = \mathrm{d}(\mathbf{a},\mathbf{b}), \tag{1}$$

where $d$ is any (quasi-)distance metric. In particular, we use Euclidean

$$d(\mathbf{a},\mathbf{b}) = \|\mathbf{a}-\mathbf{b}\|_2 \tag{2}$$

and cosine

$$d(\mathbf{a},\mathbf{b}) = 1 - cos(angle(\mathbf{a},\mathbf{b})) = 1 - \frac{\mathbf{a}^T\mathbf{b}}{\|\mathbf{a}\|_2 \|\mathbf{b}\|_2} \tag{3}$$

distances, as well as the L1 distance

$$d(\mathbf{a},\mathbf{b}) = \|\mathbf{a}-\mathbf{b}\|_1 . \tag{4}$$

Let $\mathbf{x}_i$ be the $i$-th training sample, $\mathbf{z}_i$ its latent representation and $y_i$ its label. After supervised clustering, a prototype vector $\mathbf{p}_i$ is selected out of the set of all cluster centers $C$ and the entire training data is projected on the one dimensional axis given by $d(\mathbf{z},\mathbf{p}_i)$. The cluster center for the $i$-th class is computed by averaging all the latent representations with the same label and finding the closest point in the training data, yielding

$$\mathbf{p}_i = \underset{j,y_j=i}{\arg\min} \left\| \frac{1}{N_k} \sum_{k,y_k=i} \mathbf{z}_k - \mathbf{z}_j \right\|_2 , \tag{5}$$

where $N_k$ is the number of points labeled as class $k$.

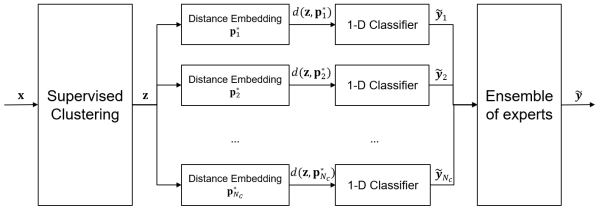


*Figure 1.* Block diagram of our proposed approach.

Our goal is, given clustered data and a metric $d$, finding the cluster center $\mathbf{p}_i$ that leads to a separable distance embedding in terms of the labels $y_i$. We propose the following criteria for ranking the embeddings, that attempts to measure the separability of the different classes on the projected one-dimensional embedding. Intuitively, we claim that the more an embedding causes the projected data to be spread in the one-dimensional space, the better it is for subsequent classification. The highest variance embedding is defined as

$$\mathbf{p}_1^* = \arg\max_{\mathbf{p}_j \in C} \mathrm{Var}(d(\mathbf{p}_i, \mathbf{p}_j)). \tag{6}$$

Similarly, the second-highest variance embedding is picked as the second best and defined as

$$\mathbf{p}_2^* = \arg\max_{\mathbf{p}_j \in C, \mathbf{p}_j \neq \mathbf{p}_1^*} \mathrm{Var}(d(\mathbf{p}_i, \mathbf{p}_j)). \tag{7}$$

Effectively, a single sorting is sufficient to rank the embeddings. Once the order of the embeddings is fixed, we train a deep neural network for classifying each embedding independently by using the cross-entropy loss, as well as a regularization term for the weights of the network as

$$L_i = -\sum_{k=1}^{N} \sum_{j=1}^{N_c} y_j \log f_{i,j}(d(\mathbf{z}_k, \mathbf{p}_i^*; \theta_i)) + \lambda \left\| \theta_i \right\|_2^2, \tag{8}$$

where $\theta_i$ are the weights of the $i$-th deep classifier, $f_{i,j}$ is its output predicted probability for the $j$-th class, $N_c$ is the total number of classes and $\lambda$ controls the regularization term importance.

Note that even though we train a potentially very deep and wide network for classification, its input and output dimensions (after selecting the maximum probability) will always be equal to one, thus we can replace the trained network with a one-dimensional decision set. In this sense, selecting the right embedding is critical to derive a decision set of minimal length and zero overlap, enabling us to produce succinct, interpretable explanations for each classifier. Finally, we use an ensemble approach to combine the outputs of different classifiers. Our complete source code is available online at https://github.com/mariusarvinte/RAI-FinalProject.

### 3.1. Aspects of interpretability

In this section, we describe different notions of interpretability found in our proposed solution and how each of them is achieved.

**Case-Based Reasoning** We incorporate case-base reasoning via computing distance to prototype vectors, Equation 5.

**Algorithm 1** The proposed algorithm

**Input**: Labeled training data $(\mathbf{x}_i, y_i)$.
Obtain clustered representation $(\mathbf{z}_i, y_i)$.
**for** Each label $i \in N_c$ **do**
  Compute cluster center $\mathbf{p}_i$ using Equation 5.
**end for**
**for** Each label $i \in N_c$ **do**
  Compute distribution of distance to all other cluster centers $d(\mathbf{p}_i, \mathbf{p}_j)$.
  Train classifier using Equation 8.
**end for**
Sort embeddings using Equation 6.
Add embeddings to ensemble of experts until validation accuracy stops increasing.

We ensure the prototype vectors are interpretable by forcing each cluster center in the latent space to match the closest training data point with the same class. During training the one-dimensional classifier, we allow the prototype vectors to be trainable, but heavily penalize their distance from the closest training point by using an R1 regularization term as in (Li et al., 2018).

**Rule-Based Reasoning** The rule-based reasoning component of our approach comes from both the one-dimensional classifiers and their ensemble. After training, each classifier is replaced with a decision set and the model can provide succint explanations for why each input is classified by only displaying the active rules. Minimizing length ensures that the rules are short and concise and minimizing overlap ensures that each rule covers an independent part of the feature space (Lakkaraju et al., 2016). By using the distance embedding, we derive decision sets that have a single feature (distance) to achieve interpretability. We also explore four rule-based strategies (majority vote, max vote, average vote, and median vote) to combine the scores from all the one-dimensional classifiers.

**Decomposability** We accomplish decomposability by designing the three components of our approach (deep supervised clustering, distance embedding, and an ensemble of interpretable classifiers) to be decoupled. In other words, each component operates on a stand-alone basis so the user can improve on or use any of these components individually.

**Algorithmic Transparency** Since our approach is decomposable, this helps the model also possess the quality of algorithmic transparency. The user can easily observe how training changes for each of the components, as well as the order in which components (classifiers) are added and how they boost performance. Since each of the one-dimensional classifiers is independent from the others, some can be trained for performance and others for interpretability (i.e., decision sets with minimal number of rules) without affecting any of

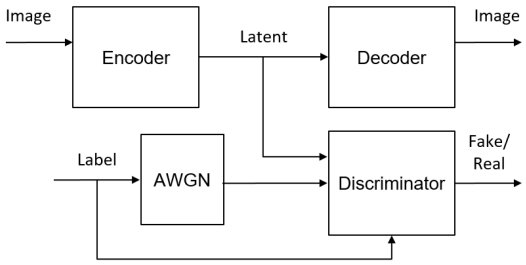


*Figure 2.* Block diagram of the supervised adversarial autoencoder architecture from (Makhzani et al., 2015).

the other components.

## 4. Experiments

### 4.1. Dataset 1: Toy Dataset

We first verify our idea of classification based on distance embeddings using two synthetic toy datasets. The objective is to investigate the impact of different distance metrics, including L1, L2 and cosine distance, as well as to validate our proposed ranking of distance embeddings. Specifically, we compare the final classification accuracies on validation datasets using different distance metrics and with different order of adding additional distance embeddings. Although an ensemble of 1-D classifiers is proposed in our decomposable architecture, such ensemble of experts might sacrifice accuracy for interpretability. Hence, we use a deep ReLu neural network with 3 hidden layers jointly trained on incrementally increased distance embeddings in this experiment to provide a better basis of comparison. The bigger capacity of this jointly trained deep classifier should minimize the effect of the choice of classifier on the final accuracy and allow us to investigate the effect of distance metrics and embedding rankings.

We consider two scenarios: small number of entangled clusters and large number of well-separated clusters, both in 10-D. We generate cluster centers by sampling from a uniform distribution over $[-1, 1]^{10}$. We generate $10^7$ sets of cluster centers and pick the set with the maximum minimum pairwise distance between the centers. This ensures that there is reasonable separation between the cluster centers. We obtain the data points by adding zero-mean Gaussian noise to the cluster centers independently in each dimension. For the small number of entangled clusters case, we consider 10 clusters with a noise variance of 0.5, while for the large number of well-separated clusters case, we consider 25 clusters in with a noise variance of 0.02. We report the average validation accuracy across many independent iterations, where we reinitialize and retrain the classifiers multiple times.

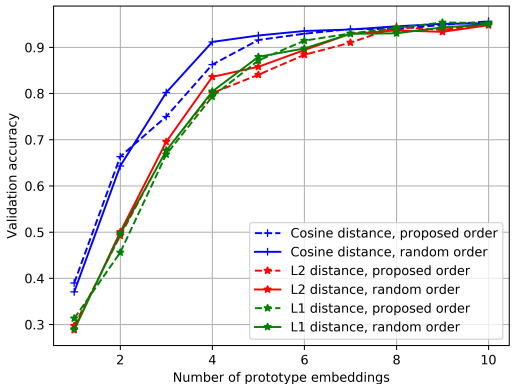


*Figure 3.* Average validation accuracy for synthetic data, 10 overlapping clusters.

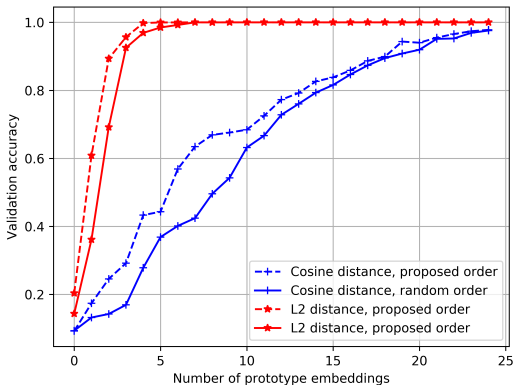


*Figure 4.* Average validation accuracy for synthetic data, 25 separated clusters.

With 10 clusters, the mean validation accuracy as a function of the number of distance embeddings for different distance metrics is shown in Fig. 3. Clearly, cosine similarity outperforms both L1 and L2 consistently in terms of its validation accuracies, especially early on with few distance embeddings. When we use 8 or more embeddings, the number of distance embeddings is big enough to capture most of the geometry in the dataset, regardless of the used distance metric. There is not a significant difference between the validation accuracies as we increase the number of distance embeddings with random order or our proposed ranking. Overall, in this scenario with small number of clusters and high noise, cosine distance performs consistently better compared to L1 and L2 distances. Our proposed embedding ranking does not offer a significant improvement over random ordering in terms of mean accuracy, but it does reduce the variance across multiple runs (since our ordering is fixed).

For 25 clusters, the mean validation accuracy with cosine or L2 distance, and with proposed or random ordering, is shown in Fig. 4. In this case, L2 distance performs significantly better than cosine distance does. Starting with a few distance embeddings, incrementally adding embeddings with L2 distance provides a significantly larger improvement in validation accuracy. Our proposed embedding ranking also proves to outperform random ordering, especially when we have a few embeddings and try to add another one. Overall, L2 distance achieves better accuracy when classifying well-clustered data. Our proposed ranking of the embeddings is thus able to reduce the number of required embeddings to achieve a certain target for accuracy when greedily adding embeddings.

The previous experiments show that our proposed ranking performs better and more consistently than random ordering. Cosine distance is better for classifying a small number of classes with high noise, while L2 distance is better for classifying data from a large number of well-separated clusters.

### 4.2. Dataset 2: MNIST

The second dataset we experiment on is the well-known MNIST (LeCun et al.), containing black-and-white handwritten images of the digits 0-9, with an image size of 28x28 pixels. Our goal is to show that our proposed architecture can cluster the data and classify the data with good accuracy, while achieving interpretability.

For real image datasets, we use an adversarial autoencoder (AAE) (Makhzani et al., 2016) for deep supervised clustering. The overall architecture of the AAE is shown in Fig. 2. The encoder maps the input images into a lower-dimension latent space. The decoder reconstructs the image from the latent space. The encoder also acts as a generator and forms a generative adversarial network (GAN) with the discriminator. We generate multimodal additive white Gaussian noise (AWGN) with different mean and variance for each class. The images in latent space, their labels as well as the labels with AWGN are fed into the discriminator. Overall this architecture forces the mapped images in the latent space to a multimodal prior distribution, thus effectively achieving deep supervised clustering. After training, we bind the prototype vectors to training data and can discard the decoder and discriminator. For the encoder, decoder and discriminator, we use a fully-connected neural network with 4 hidden layers. We consider a 10-D latent space and use cosine similarity as the distance metric.

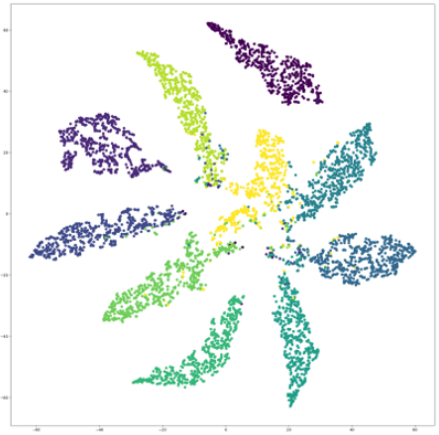

*Figure 5.* t-SNE visualization of the MNIST dataset after supervised clustering with a deep adversarial autoencoder.

*Figure 6.* Decoded prototype images for the MNIST dataset.

A 2-D visualization of the clustering using t-SNE is shown in Fig. 5. We achieve a silhouette score of 0.48. The silhouette score measures the similarity between a point and its cluster compared to other clusters and ranges from -1 to 1. Hence our clustering of MNIST is reasonably-separated. The images reconstructed from the cluster centers using the decoder are shown in Fig. 6. These prototypes are well-defined, easily interpretable images from each class. We then project the dataset onto 1-D by considering the cosine distance from one of the prototypes. and train a three hidden layer ReLU network for each distance embedding as its 1-D classifier.

Since each neural network only classifies on 1-D data, we can easily extract the decision boundaries, which are in-

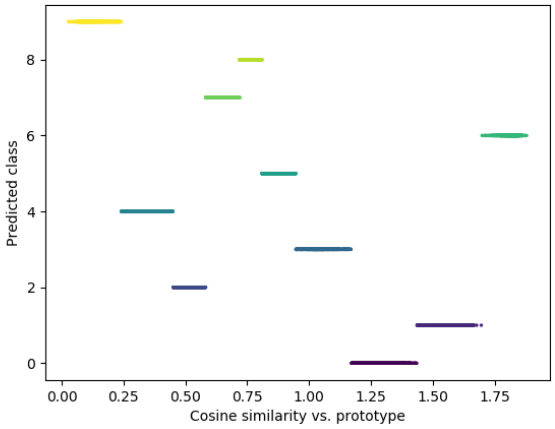


*Figure 7.* Predicted class for the classifier based on the distance embedding generated by the prototype '9'.

tervals, and replace the neural network with a decision set. However, if the intervals are fragmented for each class, we need to extract complex, long decision sets which are less interpretable. The cosine similarity of each class with respect to a prototype of class 9 is shown in Fig. 7. Each class corresponds to one interval in cosine similarity and there is no overlap between different regions. This shows we can replace the neural network with the simplest possible decision set: if distance from the prototype '9' is smaller than 0.2, then image is a '9', etc.

After training the 1-D classifiers, we essentially obtain an ensemble of classifiers and consider four strategies to combine their predicted outputs. Majority vote and max-vote require partial knowledge from the 1-D classifiers. The majority vote requires each classifier to predict and forward the class, after which the class with the majority vote is chosen as the final prediction, with ties broken randomly. Max-vote requires each classifier to predict and forward the class as well as its likelihood. The most confident prediction with the highest likelihood is chosen as the final prediction. In contrast, average vote and median vote require the classifiers to forward the predicted likelihood of each class. The average vote chooses the class with the highest average likelihood among all 1-D classifiers. The median vote chooses the class with the highest median likelihood. The final accuracy on the validation dataset with different score combination strategies as we incrementally increase the number of distance embeddings (i.e., number of experts) is shown in Fig. 9. As expected, the score combination strategies that require full knowledge from the classifiers achieve consistently better accuracies compared to the strategies that require partial knowledge. As we increase the number of distance embeddings, the final accuracy also increases. It is noteworthy that with just one distance embedding, we can achieve a validation accuracy of 85%, which is significantly better than the 10% achieved with the approach proposed in (Li et al., 2018).

Since each classifier can be replaced with a minimum length decision set, we can generate semantic explanations for each input during the testing phase of the type 'this image was classified as '9', since it is twice as close to a representative image of '9' than it is to a representative image of '8''.

Overall, we show that our decomposable and interpretable architecture can achieve good classification accuracy (94%) on MNIST. The final architecture for classification is shown in Fig. 8. We use the deep encoder to map the input images into a well-clustered latent space. We then project the dataset onto spaces in 1-D by calculating the distance to prototype images (cluster centers). Finally, we use simple decision sets and a score combination strategy to predict the class. Either the deep encoder, the distance embeddings, the 1-D classifiers or the score combination strategy can be replaced, thus achieving decomposability. The classification uses decision sets based on distance from well-defined prototypes, thus achieving interpretability.

### 4.3. Dataset 3: COIL-20

The COIL-20 dataset contains 1440, 128 x 128 grayscale images of 20 objects with 72 poses each (Nene et al., 1996). The images have high resolution and also have less data points per class which makes this dataset more challenging to reconstruct, as well as harder to find a well separated clustered representation for.

For this dataset, we train a 5-hidden layer convolutional encoder and decoder, with a hidden dimension of size 1200 and a latent dimension of size 10, same as MNIST and our synthetic experiments. All filter sizes have a dimension of $3 \times 3$. The discriminator is a fully connected ReLU network with the same number of layers and hidden dimension. Experimentally, we determine that Euclidean distance offers better separation than cosine or L1, which is also consistent with our synthetic experiment with a large number of clusters.

Figure 10 shows a 2-D visualization of the best obtained clustering using t-SNE. This achieves a silhouette score of 0.28 and, while it is not as good as MNIST, we can still notice some clusters are separated from others. We apply our method and rank the twenty prototype vectors according to our proposed criteria, and train a 3-hidden layer one-dimensional ReLU classifier for each, after which we implement the same voting strategies as in the previous section. Figure 11 plots the obtained validation accuracy, where we note that the peak performance (with all classifiers) is approximately 75% and this is achieved by the ensemble methods that leverage the complete prediction information.

We now investigate how the learned one-dimensional clas-

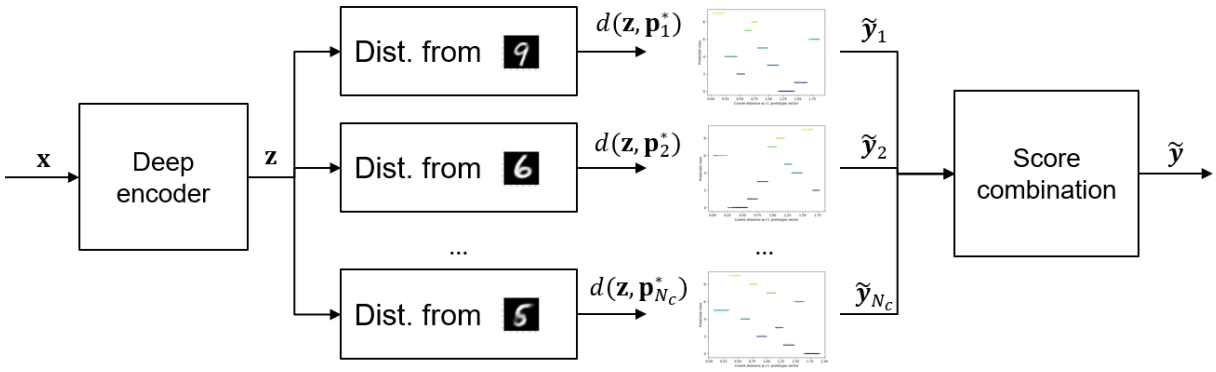


*Figure 8.* Interpretable architecture for the MNIST dataset. Each distance embedding is computed w.r.t. the latent representation of a training point and the classifiers are minimal length decision sets.

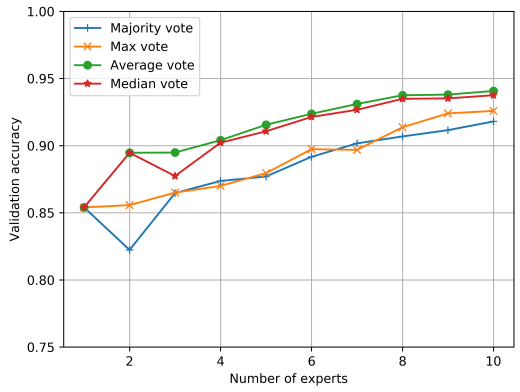


*Figure 9.* Validation accuracy for the ensemble of experts approach for MNIST with different voting strategies. All distance embeddings use the cosine distance.

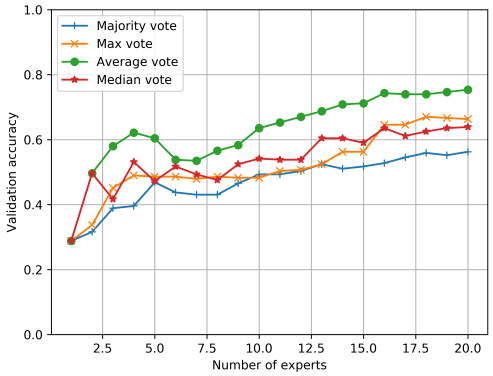


*Figure 11.* Validation accuracy for the ensemble of experts approach for COIL-20 with different voting strategies. All distance embeddings use the Euclidean distance.

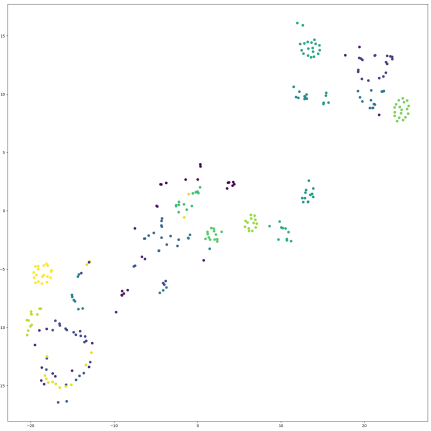

*Figure 10.* t-SNE visualization of the COIL-20 dataset after supervised clustering with a deep adversarial autoencoder.

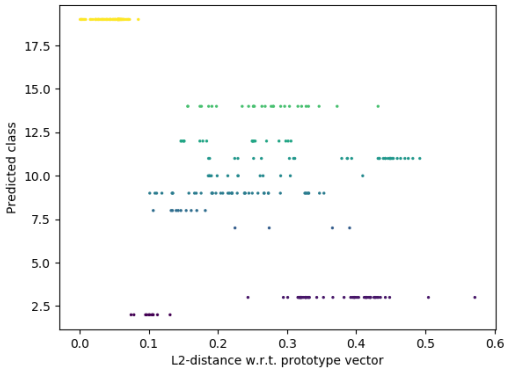


*Figure 12.* Predicted class for the classifier based on the distance embedding generated by the highest-ranked prototype according to our metric. This classifier achieves approximately 35% validation accuracy.

sifiers look like for this dataset. Figure 12 plots the input-output mapping for the highest-ranked expert according to our proposed ranking. In this case, we can notice that the decision regions are more fragmented than in the MNIST case, thus the corresponding decision set will have a significantly larger number of rules

## 5. Conclusions

We propose a decomposable classification method that combines deep learning, case-based and rule-based reasoning to achieve interpretability. We design an algorithmic rule for ranking the candidate prototype vectors. For classification, we use an ensemble of interpretable classifiers where classifiers are incrementally added according to their rank. We experimentally show that our ranking is consistent, and we can achieve up to 94% performance on MNIST with a mixture of simple decision sets. We have empirically shown that our method can handle more complicated datasets by running it for the COIL-20 dataset, where even though we obtain a weak clustering, we are still able to get up to 75% validation accuracy.

Future work includes extending our proposed ranking to multi-dimensional experts, since using two-dimensional classifiers would still be amenable to visualization and interpretability, as well as investigating regularization approaches for deep classifiers that result in low fragmentation of the decision regions.

## References


Bien, J., Tibshirani, R., et al. Prototype selection for interpretable classification. *The Annals of Applied Statistics*, 5(4):2403–2424, 2011.

Branson, S., Van Horn, G., Belongie, S., and Perona, P. Bird species categorization using pose normalized deep convolutional nets. *arXiv preprint arXiv:1406.2952*, 2014.

Breiman, L. *Classification and regression trees*. Routledge, 2017.

Chen, C., Li, O., Barnett, A., Su, J., and Rudin, C. This looks like that: deep learning for interpretable image recognition. *arXiv preprint arXiv:1806.10574*, 2018.

Dietterich, T. G. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pp. 1–15. Springer, 2000.

Erhan, D., Bengio, Y., Courville, A., and Vincent, P. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

Kim, B., Rudin, C., and Shah, J. A. The bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Advances in Neural Information Processing Systems*, pp. 1952–1960, 2014.

Lakkaraju, H., Bach, S. H., and Leskovec, J. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1675–1684. ACM, 2016.

LeCun, Y., Cortes, C., and Burges, C. J. C. The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pp. 609–616. ACM, 2009.

Lei, T., Barzilay, R., and Jaakkola, T. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*, 2016.

Li, O., Liu, H., Chen, C., and Rudin, C. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Lipton, Z. C. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*, 2016.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Makhzani, A., Shlens, J., Jaitly, N., and Goodfellow, I. Adversarial autoencoders. In *International Conference on Learning Representations*, 2016. URL http://arxiv.org/abs/1511.05644.

Montavon, G., Samek, W., and Müller, K.-R. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

Nene, S. A., Nayar, S. K., and Murase, H. Columbia object image library (coil-20. Technical report, 1996.

Oord, A. v. d., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

Pinheiro, P. O. and Collobert, R. From image-level to pixel-level labeling with convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1713–1721, 2015.

Poursabzi-Sangdeh, F., Goldstein, D. G., Hofman, J. M., Vaughan, J. W., and Wallach, H. Manipulating and measuring model interpretability. *arXiv preprint arXiv:1802.07810*, 2018.

Rivest, R. L. Learning decision lists. *Machine learning*, 2 (3):229–246, 1987.

Rokach, L. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, 2010.

Ruta, D. and Gabrys, B. Classifier selection for majority voting. *Information fusion*, 6(1):63–81, 2005.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. 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.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Song, M., Montanari, A., and Nguyen, P. A mean field view of the landscape of two-layers neural networks. *Proceedings of the National Academy of Sciences*, 115:E7665–E7671, 2018.

Wu, C. and Tabak, E. G. Prototypal analysis and prototypal regression. *arXiv preprint arXiv:1701.08916*, 2017.

Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., and Lipson, H. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.