Explaining Image Classifications with Near Misses, Near Hits and Prototypes

Supporting Domain Experts in Understanding Decision Boundaries

Marvin Herchenbach[1,2(✉)], Dennis Müller[1,2], Stephan Scheele[1,3], and Ute Schmid[1,3]

[1] Sensory Perception and Analytics | Comprehensible AI, Fraunhofer Institute for Integrated Circuits IIS, Erlangen, Germany
{stephan.scheele,ute.schmid}@iis.fraunhofer.de
[2] Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
{marvin.herchenbach,dennis.mueller}@fau.de
[3] Otto-Friedrich-Universität Bamberg, Bamberg, Germany

Abstract. We propose a method for explaining the results of black box image classifiers to domain experts and end users, combining two example-based explanatory approaches: Firstly, *prototypes* as representative data points for classes, and secondly, contrastive example comparisons in the form of *near misses* and *near hits*. A prototype *globally* explains the relevant characteristics for a entire class, whereas near hit and near miss explain the *local* decision boundary of a specific prediction. To combine both types of explanations within one framework is novel and we propose that presenting both types of explanations is especially helpful for domain experts in visual domains. To improve the faithfulness of the explanations, we investigated an unbiased, generic embedding and a model-related (model-specific) embedding for handling the images. The proposed approaches are evaluated regarding parameter selection and suitability on two different data sets – the well-known MNIST and a real-world industrial quality control data set. Finally, it is shown how global and local example-based explanation can be combined and realized within a demonstrator.

Keywords: Explainable AI · Example-based explanation · Prototypes · Near misses · Near hits

1 Introduction

Machine learning (ML) based image classification algorithms, such as deep neural networks, are increasingly employed in settings where transparency and comprehensibility of decisions are crucial such as medical diagnostics or industrial quality control. Research on *explainable artificial intelligence* (XAI) is addressing

M. El Yacoubi et al. (Eds.): ICPRAI 2022, LNCS 13364, pp. 419–430, 2022.
https://doi.org/10.1007/978-3-031-09282-4_35

these requirements [1] by providing techniques to support the decision making of ML black-box models and thereby allow users to develop justified trust [14]. Many XAI methods identify the most relevant information in the input for the classifier decision. While this information is helpful for model developers, e.g., to detect overfitting [13], it might be not expressive enough to explain model decisions for domain experts such as medical experts or quality engineers [14].

Cognitive science research provides theories as well as empirical evidence that explanations by examples are highly effective for humans to grasp complex concepts [7,11,12]. Therefore, we consider in this paper two kinds of examples, with the specific goal of explaining *image classifier* AI models to end users and domain experts without expertise in machine learning:

1. *Prototypes*, representing *typical* representative instances of some image class as a *global* explanation of the model, and
2. *Near hits and misses* of some given input, representing examples from the training data similar to the input image and from the same (or opposite, respectively) class, as *local* explanations.

In combination, prototypes, near hits and near misses allow users to get a better understanding of information considered relevant as well as of the decision boundaries of a given classification algorithm.

Numerous algorithms for computing prototypes of a given data set exist. In this paper, we primarily use [8], a widely used state-of-the-art approach based on *Maximum Mean Discrepancy* (see Subsect. 3.1 for details). ProtoDash [4] builds on the former, but at time of writing, no adequate implementation with sufficient adaptability for our experiments could be found. We additionally use *Partitioning around Medoids* [15] as a baseline approach for comparison; an improved version of a simple k-Medoids clustering algorithm [6], where we interpret the associated medoids of each cluster as prototypes.

Near hits and misses (*NHM*s) as relating to classified data are much less well covered by the existing literature, especially as an explanatory tool. One notable exception is [11], where NHMs are computed specifically for Prolog clauses to explain classifications in the context of *Inductive Logic Programming*. Conceptually however, finding close matches of a given input according to some metric is a ubiquitous tool in many distinct areas, such as in feature selection [17] or – more closely related to our purposes – in *content-based image retrieval* [5].

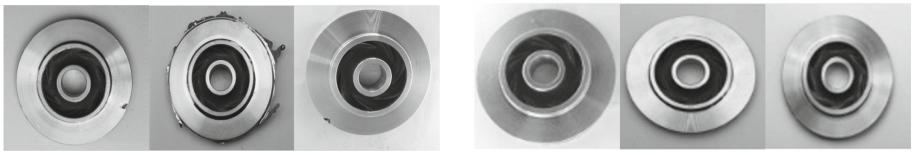
For providing more faithful explanations, we differentiate between two vector embeddings for handling images: a *model-specific* relying on the CNN-based classification model to be explained, and a *model-agnostic* allowing obtain another embedding unbiased by our data sets and unrelated to our classification model.

In the following, we describe the algorithms used for example-based explanations with focus on their evaluation for two data sets – the classic MNIST and a real-world data set of casting manufacturing image data for industrial quality control [2]. We start in Sect. 2 with describing the setup for our experiments, i.e. the data sets and classifier models used, and a brief overview of the final user-centric architecture. Sections 3 and 4 deal with prototypes and near hits and misses, respectively, the algorithms used, their parameters and our evaluations thereof. Lastly, in Sect. 5 we present our demonstrator implementation.

2 Methodology

2.1 Data Sets

We primarily use [2] for our experiments; a data set consisting of 1100 grayscale images of cast metal components of size 512×512 labelled with one of two classes, "ok" (419 entries) and "defective" (681 entries), see Fig. 1. The entries of the latter class show various kinds of defects, e.g., blow holes, abrasions, scratches etc. (see Fig. 1a). Notably, the data set is highly homogeneous in that the objects in the images are very similar to each other (except for the defects, which are usually subtle), but differ with respect to features that are irrelevant for purposes of classification, e.g., lighting conditions and angle (see Fig. 1b). The data set occasionally contains multiple images from different angles of the same object, which makes it especially interesting for the purpose of evaluating near hits and misses.



(a) defective. (b) ok.

Fig. 1. Some examples from the casting data set [2].

For comparison, we additionally use the MNIST data set of handwritten digits [9]. Since the casting data set is restricted to two classes, we correspondingly restrict MNIST to two classes – namely "1" and "7" (each consisting of 7877 and 7293 entries, respectively), which are uniformly white digits on black background, but differ significantly in their shapes within their respective classes.

2.2 Models and Embeddings

For each of our two data sets, we trained a small standard convolutional neural network (CNN) with three convolutional and two fully connected layers on the respective classification tasks, with resulting accuracies of 96.82% and 99.72% respectively.

These models actually serve two purposes: Firstly, they naturally serve as toy classifier models to be explained by our overall approach. Secondly, we can use feature extraction on the models to obtain embeddings for our images, which should be sensitive to those aspects of an image that relate to its inferred class. We consequently expect these embeddings to map images with similar *class-relevant* features near each other, leading to more informative near hits and misses. However, it should be noted that by using embeddings depending on the

classifier model, our approach is *model-specific*. That is, it is required that the model to be explained is a neural network (or otherwise induces a suitable vector embedding). We therefore additionally use a generic state-of-the-art image classification model (VGG16 [16]) to obtain a second embedding unbiased by our data sets and unrelated to our classifier model, allowing us to remain *model-agnostic*. We refer to the embedding obtained via feature extraction on our classifier models as $\mathtt{E}_C$, and the one using VGG16 as $\mathtt{E}_{\mathtt{VGG}}$. We will occasionally use the raw image vectors for comparison, which we denote as the (trivial) embedding $\mathtt{E}_0$.

2.3 Architecture Overview

Figure 2 shows our approach as envisioned in practice. A user selects an image, for instance, of an industrial manufacturing component, which is classified by a CNN (or other black-box model). The inferred label is used to obtain a set of prototypes with the same label from the training data set. Both the label and the input image – under some vector embedding – are used to select a number of comparable near hits and misses from the training set. All three combined are provided to the user, allowing to better comprehend both the returned classification by comparing it to prototypical samples and the most similar (ground-truth labelled) elements from the training data (near hits), as well as the decision boundary in a contrastive manner via the near misses.

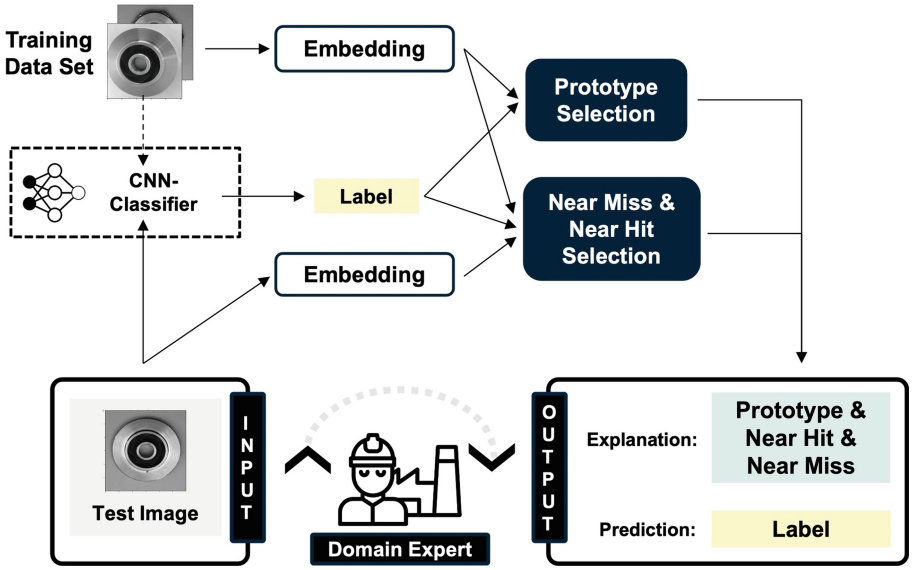


Fig. 2. Overview of the implemented architecture.

3 Prototype Selection

3.1 Prototype Selection Using Maximum Mean Discrepancy

Kim et al. [8] propose an approach for prototype selection based on *Maximum Mean Discrepancy* (MMD), a similarity measure on distributions, rather than individual data points: Given (finite approximations for) distributions X, Y, then the expression

$$\begin{aligned}\texttt{MMD}^2(X,Y) \;:=\; & \frac{1}{|X|^2}\sum_{x_1,x_2\in X} k(x_1,x_2) + \frac{1}{|Y|^2}\sum_{y_1,y_2\in Y} k(y_1,y_2) \\ & - \frac{2}{|X|\cdot|Y|}\sum_{x\in X,y\in Y} k(x,y)\end{aligned}$$

approaches 0, as X and Y become more similar with respect to a Hilbert space of testing functions with reproducing kernel k. For our purposes, we use the radial basis kernel function $k(x,y) := e^{-\gamma||x-y||}$ for a real-valued scaling parameter γ. We can use this for selecting prototypes as follows:

Given a set of embedded data points X with $|X| = n$ and a kernel function $k : X \times X \to \mathbb{R}$, our objective is to find a subset $S \subseteq X$ with $|S| = m$ such that $\texttt{MMD}^2(X,\varnothing) - \texttt{MMD}^2(X,S)$ is maximized, which can be simplified to the following cost function:

$$J(S) := \frac{2}{nm}\sum_{x\in X,s\in S} k(x,s) - \frac{1}{m^2}\sum_{s_1,s_2\in S} k(s_1,s_2)$$

The remaining aspects of the selection algorithm are straight-forward (see Algorithm 1).

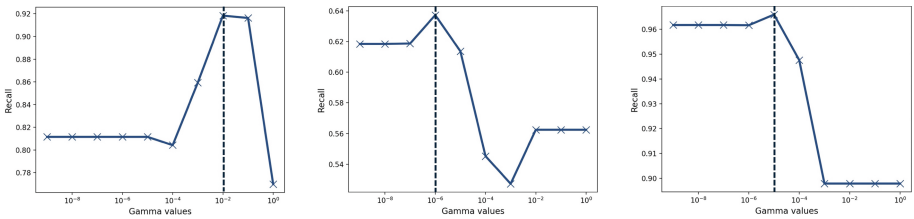
Algorithm 1. Prototype selection algorithm, adapted from [8].

Input: m, X
  $S = \varnothing$
  while $|S| < m$ **do**
    foreach: $x \in X \setminus S, j_x = J(S \cup \{x\}) - J(S)$ **do**
      $S = S \cup \{\text{argmax}\ \{j_x | x \in X\}\}$
  end while
Return: S

Surprisingly, [8] suggests using raw image data as input for the algorithm. While this works well for some data sets, e.g., MNIST, we agree with [10] that feature embeddings should yield better results in general. Nevertheless, we compare both variants.

3.2 Parameter Selection

Algorithm 1 depends on two parameters: The number m of desired prototypes and the scaling factor γ of the kernel function. To determine the optimal value for the latter, [10] suggests training a 1-Nearest-Neighbour (1-NN) algorithm on the selected prototypes to classify an test set. Additionally, we used a k-fold cross-validation averaged for robustness. Notably, the best value for γ seems to depend on both the underlying data set and the embedding used (see Fig. 3).



(a) $\mathtt{E}_C$ on casting data. (b) $\mathtt{E}_{\mathtt{VGG}}$ on casting data. (c) $\mathtt{E}_{\mathtt{VGG}}$ on MNIST data.

Fig. 3. γ-Values plotted against recall on two data sets and two embeddings while using 1-NN.

Regarding m, the natural but expensive approach would be a survey on end-users. Instead, we opt for an analytical approach by considering two methods:

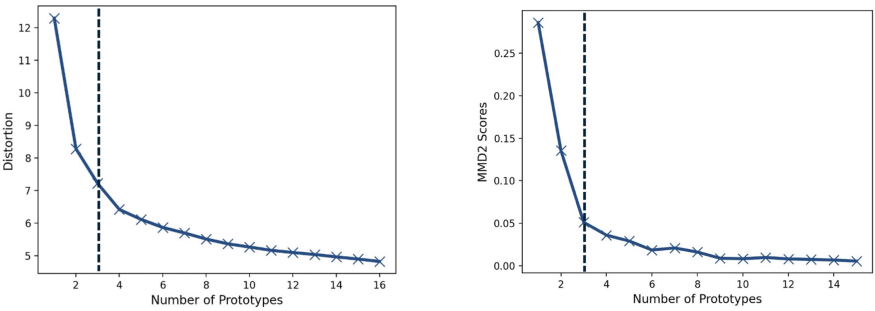
1. We perform an *Elbow method* based on a k-Means algorithm, plotting the *distortion* (i.e. the sum of square distances from each point to its assigned cluster center) wrt. the number of clusters/prototypes (see Fig. 4a).
2. We use the fact that $\mathtt{MMD}^2(X, S)$ gives us a measure of how "representative" the elements of S are with respect to our full data set X. We therefore consider a *Scree plot* (see Fig. 4b) of the $\mathtt{MMD}^2$-value against m.

We applied both methods on all three embeddings E_0, E_C and $E_{\mathtt{VGG}}$, and on both data sets, and observe at which values the respective curves flatten. In all cases, this happens noticeably at a value of about $m = 3$, which strongly suggests that more than three prototypes do not convey significantly more information. However, note that the optimal number should vary depending on the specific data set under consideration.

3.3 Evaluation

Similarly to our strategy for selecting parameters, we evaluate the resulting prototypes using a 1-NN approach with respect to the training data set. We additionally use an off-the-shelf *Partitioning around Medoids* [15] k-Medoids clustering algorithm as a baseline approach. The results are shown in Table 1.

Overall, the MMD-based approach performs mostly better than k-Medoids, although the difference is surprisingly small. As expected, the embedding $\mathtt{E}_C$



(a) Elbow method. (b) Scree Plot.

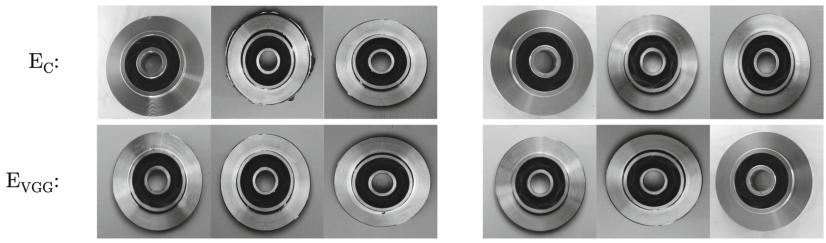
Fig. 4. Representative plots using an Elbow method (a) and a Scree plot (b).

Table 1. Results of 1-NN algorithm trained selected prototypes, evaluated with respect of the embedding based on accuracy (best performance in bold).

	Casting			MNIST		
	$\mathtt{E}_0$	$\mathtt{E}_C$	$\mathtt{E}_{\mathtt{VGG}}$	$\mathtt{E}_0$	$\mathtt{E}_C$	$\mathtt{E}_{\mathtt{VGG}}$
MMD	0,6682	**0,9273**	**0,6591**	**0,9649**	**0,9995**	**0,9750**
k-Medoids	**0,6864**	0,9136	0,6136	0,9598	**0,9995**	0,9653

obtained via feature extraction on the classifier model itself shows consistently better results than the alternative embeddings. Surprisingly, the unbiased embedding ($\mathtt{E}_{\mathtt{VGG}}$) is also superior to the raw data ($\mathtt{E}_0$) on the casting data set, while not on MNIST - probably due to the simple structure and uniform background.

Figure 5 shows the resulting prototypes of our primary data set. Notably, the defective prototypes using $\mathtt{E}_C$ cover exactly the three primary kinds of defects occurring in the data set - a blowhole in the first, abrasions in the second, and a scratch in the third, whereas the $\mathtt{E}_{\mathtt{VGG}}$ based prototypes are noticeably less diverse



(a) defective (b) ok

Fig. 5. Selected prototypes of the casting data set.

in that respect. Like the corresponding data samples themselves, the prototypes for the "ok" class are largely very similar, regardless of the embedding used.

4 Near Miss and Hit Selection

Regarding NHMs, our algorithm is conceptually straight-forward (see Algorithm 2). Given a data sample e, we choose as a subset X of our *training* data either those samples with the same inferred label as e (near hits) or those with a different label (near misses). Then we compare each element of X to e by some given metric $m : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$: (i) The euclidean metric $\sqrt{\sum_i (x_i - y_i)^2}$, (ii) the manhattan metric $\sum_i |x_i - y_i|$ and (iii) the cosine metric $1 - x \cdot y / \|x\| \cdot \|y\|$, again using all three of our embeddings.

Algorithm 2. Near Miss/Hit Algorithm.

Input: e data sample, X data set, m metric
  $L = \varnothing$
  foreach: $x \in X \setminus \{e\}$ **do**
    $L = L \cup \{\langle x, m(x, e)\rangle\}$
  sort L by second component
Return: L

4.1 Evaluation

Evaluating the accuracy of NHMs *analytically* is considerably difficult in that no *objective* measure of similarity – especially with respect to those features that are relevant for classification – exists, which could serve as a baseline comparison. While this applies equally to prototypes, this problem becomes a lot more prominent here, where comparisons between individual pairs of data samples need to be considered. Ideally, we would evaluate the possible vector embeddings and metrics in a large-scale user study. In lieu of that, we opted for manually inspecting random samples of NHMs on both data sets with varying parameters.

One clear and unsurprising result is the superiority of the classifier embedding $\mathtt{E}_C$. This is particularly noticeable with *near misses* on the MNIST data set. Figure 6 shows some near misses for the class "7" for both embeddings. Notably, the near misses obtained using $\mathtt{E}_C$ all have something resembling a corner at the upper end, which could indicate a number 7, whereas using $\mathtt{E}_{\mathtt{VGG}}$ quickly yields plain lines, much more reminiscent of a 1.

Furthermore, the near misses using $\mathtt{E}_C$ seem to differ much more rarely depending on the metric used, or even the input image used. This makes sense, assuming the data samples are distributed such that near misses reduce to those data samples which most closely resemble the opposite class. For example, Fig. 7 shows the first five near hits for an image of class "1", which are notably similar for all three metrics and for several input images.

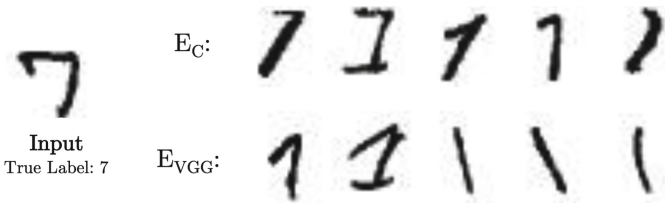


Fig. 6. Near misses for an input of class "7".

Fig. 7. Regularly occurring near hits for the class "1".

The advantage of $\mathtt{E}_C$ over $\mathtt{E}_{\mathtt{VGG}}$ is much less noticeable on the casting data set (see Fig. 8), however. We conjecture that the homogeneity of the data set allows for either embedding to primarily focus on the relevant differences, i.e. exactly the defects, since even generic embeddings should largely be able to abstract from rotation, angle and similar unimportant variations.

With respect to the metric used, different choices yield different, but very similar results (equal in only ≈30% of cases). In fact, we could not notice a clear advantage of either over the others, regardless of the choice of embedding, with possibly a slight advantage of euclidean and manhattan distances over cosine.

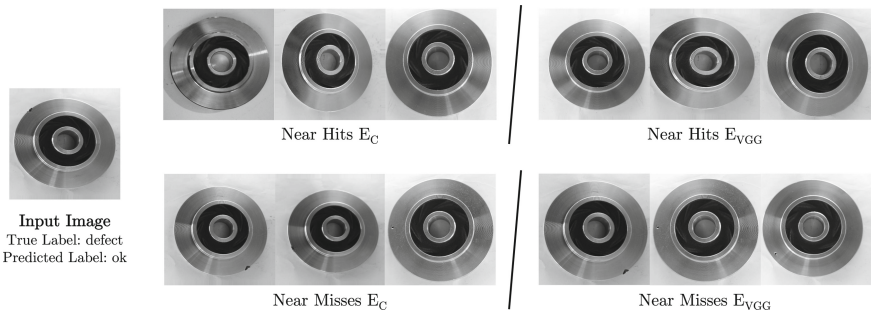


Fig. 8. $\mathtt{E}_C$-based NHMs exhibits more striking features regarding the example input image, to indicate faster this misclassified image of class "defective"

5 Demonstrator

We implemented a web-based interactive demonstrator (see Fig. 9). A user can choose a model-specific ($\mathtt{E}_C$) or model-agnostic ($\mathtt{E}_{\mathtt{VGG}}$) embedding, one of our two example data sets, a metric (cosine, euclidean, manhattan), the number

of near hits and misses to show, and an input image from the test sets. The system then displays the input image itself, its classification according to the CNN, the corresponding probability, prototypes for the classes and near hits and misses with their corresponding distances according to the metric chosen. The demonstrator, and all code relating to our evaluations is available online[1].

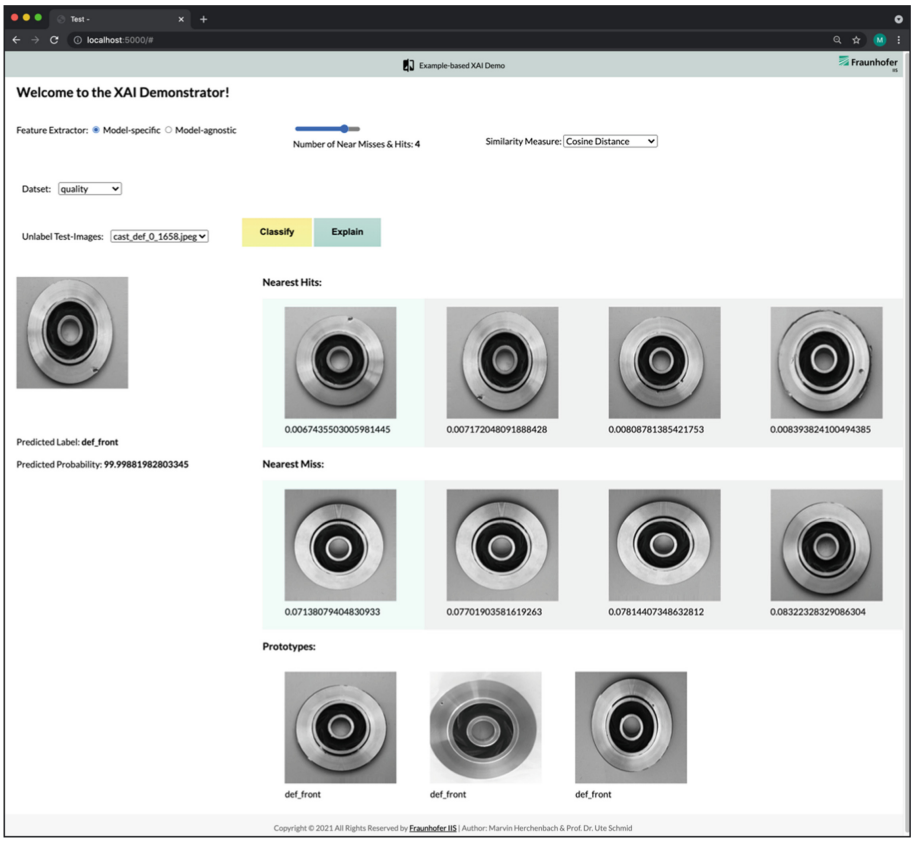


Fig. 9. Screenshot of our XAI demonstrator.

6 Conclusion and Future Work

We presented an example-based XAI approach for image classification models providing prototypes as global explanation, as well as near misses and hits to explain the local decision boundary of a prediction. Our experiments showed that model-specific embeddings are more informative with respect to decision boundaries than model-agnostic ones. In a next step, more advanced prototype

[1] https://gitlab.cc-asp.fraunhofer.de/sees/vis-ml2022-mh.

selection algorithms can be evaluated, e.g., re-implementing the ProtoDash algorithm from [4].

Although there already exists some empirical evidence showing that humans can profit from these types of example-based explanations [7,11], we plan to conduct user studies to evaluate the helpfulness of our demonstrator for the visual quality control task. Performance accuracies for predictions of class decisions of the CNN models will be compared for (a) visual highlighting, (b) prototype explanations, (c) near hit and miss explanations, and both prototype and near hit/miss explanations. Another useful enhancement could be highlighting the dissimilarities or similarities between the test image and the near miss or hit by using saliency maps – e.g. similarity based saliency maps stemming from CBIR [3] – to enable much more precise and faster indication of the decision boundaries to the domain expert. Finally, the user interface of the demonstrator can be improved with respect to intuitive interaction, ease of information acquistion, and positive user experience.

References

1. Adadi, A., Berrada, M.: Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). IEEE Access **6**, 52138–52160 (2018). https://doi.org/10.1109/ACCESS.2018.2870052
2. Dabhi, R.: Casting product image data for quality inspection (2020). https://www.kaggle.com/ravirajsinh45/real-life-industrial-dataset-of-casting-product
3. Dong, B., Collins, R., Hoogs, A.: Explainability for content-based image retrieval. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp. 95–98 (2019)
4. Gurumoorthy, K.S., Dhurandhar, A., Cecchi, G., Aggarwal, C.: Efficient data representation by selecting prototypes with importance weights. In: Proceedings IEEE International Conference on Data Mining (ICDM), pp. 260–269 (2019). https://doi.org/10.1109/ICDM.2019.00036
5. Hameed, I.M., Abdulhussain, S.H., Mahmmod, B.M.: Content-based image retrieval: a review of recent trends. Cogent Eng. **8**(1) (2021). https://doi.org/10.1080/23311916.2021.1927469
6. Kaufmann, L., Rousseeuw, P.: Clustering by means of medoids. In: Data Analysis Based on the L1-Norm and Related Methods, pp. 405–416 (1987)
7. Kenny, E.M., Ford, C., Quin, M., Keane, M.T.: Explaining black-box classifiers using post-hoc explanations-by-example: the effect of explanations and error-rates in XAI user studies. Artif. Intell. **294**, 103459 (2021)
8. Kim, B., Khanna, R., Koyejo, O.: Examples are not enough, learn to criticize! criticism for interpretability. In: Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS 2016, pp. 2288–2296. Curran Associates Inc., Red Hook, NY, USA (2016). https://doi.org/10.5555/3157096.3157352. ISBN 9781510838819
9. LeCun, Y., Cortes, C., Burges, C.J.: The MNIST database of handwritten digits (1998). http://yann.lecun.com/exdb/mnist/
10. Molnar, C.: Interpretable Machine Learning. A Guide for Making Black Box Models Explainable, p. 247 (2019). https://christophm.github.io/interpretable-ml-book

11. Rabold, J., Siebers, M., Schmid, U.: Generating contrastive explanations for inductive logic programming based on a near miss approach. Mach. Learn., 1–22 (2021, online first). https://doi.org/10.1007/s10994-021-06048-w
12. Renkl, A.: Toward an instructionally oriented theory of example-based learning. Cogn. Sci. **38**(1), 1–37 (2014). https://doi.org/10.1111/cogs.12086
13. Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K., Müller, K.-R. (eds.): Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. LNCS (LNAI), vol. 11700. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-28954-6
14. Schmid, U.: Interactive learning with mutual explanations in relational domains. In: Muggleton, S., Chater, N. (eds.) Human-Like Machine Intelligence, pp. 338–354. Oxford University Press (2021)
15. Schubert, E., Rousseeuw, P.J.: Faster k-medoids clustering: improving the PAM, CLARA, and CLARANS algorithms. In: Amato, G., Gennaro, C., Oria, V., Radovanović, M. (eds.) SISAP 2019. LNCS, vol. 11807, pp. 171–187. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32047-8_16
16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, pp. 1–14 (2015)
17. Urbanowicz, R.J., Meeker, M., La Cava, W., Olson, R.S., Moore, J.H.: Relief-based feature selection: introduction and review. J. Biomed. Inf. **85**, 189–203 (2018). https://doi.org/10.1016/j.jbi.2018.07.014