Enhancing Interpretability of Self-Explaining Neural Networks

Edward Günther, Massimo Höhn, Carina Schnuck
{gedward, hoehnm, cschnuck}@student.ethz.ch

Abstract— **Recent literature has introduced a class of self-explaining neural networks that naturally output explanations for their reasoning. While much work has focused on those parts of the network architecture that create explanations, less focus has been put on the parts of the model on which these explanations are applied to. To address this issue, we present two network architectures that can simply be plugged into existing self-explaining neural networks: the first uses a two stage supervised variational autoencoder architecture, the second a variational siamese network. In experiments we show that the proposed models surpass standard architecture in key aspects of interpretability. Our work demonstrates that attention should be spent on all parts of a self-explaining neural network, especially to make it more useful than post-hoc explanation methods, and that a variational component can be advantageous for interpretability. Source code can be found on `https://gitlab.ethz.ch/gedward/senn-revisited`.**

I. INTRODUCTION

Demand for interpretability of complex machine learning models has increased over past years as these models have been adopted in a wide range of real-world applications, especially in increasingly critical domains such as health care [1, 2], criminal justice [3, 4] and autonomous driving [5, 6]. Naturally, interpretability in machine learning has also gained growing attention in research [7]–[11]. Two different paradigms on achieving interpretability of machine learning models can be found in the literature: post-hoc explanation methods and intrinsically interpretable models. Our proposed model falls into the latter category, however, in the following we will briefly outline both approaches for better understanding.

Post-hoc explanation methods aim to provide information on black-box models after training and do not pose any constraints or regularizations enforcing interpretability on the model of interest. These methods can be divided into gradient or reverse propagation methods that use at least partial information given by the primary model [12]–[16], and complete black-box methods that only use local input-output behavior for explanations [17]–[19].

Conversely, an intrinsically interpretable approach aims to make a model interpretable by its formulation and architecture and thus takes into account interpretability already during training. A first and natural approach to achieve model interpretability is to restrict oneself to the use of simple and often close to linear models (for nonlinear examples see [20, 21]) that can be understood by humans on a global level [2, 22, 23]. Other approaches aim to induce sparsity in the context of a neural network. [24] directly use a Lasso penalty while [25] introduce knock off features for deep feature selection. [26] use GANs with a similar goal. Lastly and most relevant for the approach taken here, we want to mention explanation generating models. These models produce explanations which are at the same time used for predictions and thus aim to be intrinsically interpretable by network architecture. One such architecture is used in [27]. The authors design a neural network that learns relatively few prototypical representations of the data in order to make predictions. The distance of a sample to these prototypes can then be used as explanations. [28] introduce contextual explanation networks which simultaneously make predictions and provide explanations leveraging a probabilistic model.

In this work, we use an explanation generating architecture called Self-Explaining Neural Network *SENN* from [29]. A *SENN* consists of two components: a conceptizer $h(x)$ and a parametrizer $\theta(x)$. The conceptizer aims to encode input data as meaningful concepts while the parametrizer learns explanations which are then applied to concepts to obtain predictions.[1] The design of both components is crucial for attaining interpretability and [29] propose different requirements on each of them. The parametrizer should be locally interpretable. Further, three desiderata are stipulated for the conceptizer: (i) grounding (human-interpretable concepts), (ii) diversity (non-overlapping concepts), and (iii) fidelity (relevance of concepts). [29] propose the use of an additional loss on the parametrizer and the use of an autoencoder for the conceptizer. While we think the former approach can be successful (given correct parameter setting as analysed in section IV) the above stated desiderata for the conceptizer, however, are not per se fulfilled by an autoencoder: (i) human interpretability of encodings can suffer e.g. from discontinuities in the latent space [30], (ii) autoencoders do not guarantee disentangled representations [31], and (iii) although autoencoders compress raw inputs to a lower dimensional space, these embeddings may still contain information irrelevant to the prediction task [32]. Further, [33] argue that a crucial part of interpretability for a *SENN* depends on the robustness of the conceptizer itself.

[1] In [29] the output of the parametrizer is called relevances rather than explanations and the word explanation is used to describe a combination of concepts and relevances. To be consistent with terminology used in the literature mentioned earlier in our introduction, we will not follow the terminology from [29] but that used in our description of explanation generating models.

That is, the conceptizer should also be relatively stable for close inputs with the same class label (similar to the local interpretability requirement demanded on the parametrizer).

In this report we propose two distinct network architectures, *VaeSENN* (II-B) and *VSiamSENN* (II-C) that aim to advance the fulfillment of all three desiderata mentioned above compared to the autoencoder used in [29]. While the network architecture of the two approaches itself tackles all three desiderata by design we are also able to show improvements in continuity of latent space (corrsponding to grounding) and disentanglement of concepts (corresponding to diversity) in experiments (III-B) for both approaches. Moreover, for *VSiamSENN* we show that our approach also enhances interpretability-robustness of the conceptizer as defined in [33] (III-C).

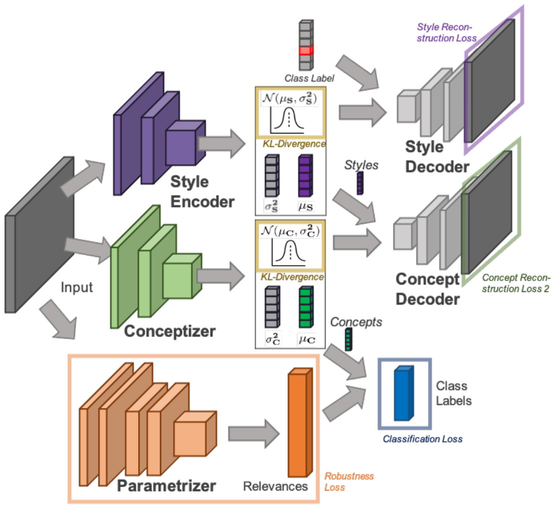

Fig. 1: VaeSENN network architecture

II. MODELS AND METHODS

A. SENN

A *SENN* [29] can be represented as follows:

$$f(x) = g(\theta_1(x)h_1(x), ..., \theta_k(x)h_k(x)) \tag{1}$$

with the following properties:

1) g is an aggregation function that is monotone and completely additively separable
2) For every $z_i := \theta_i(x)h_i(x)$, g satisfies $\frac{\partial g}{\partial z_i} \geq 0$
3) θ is locally difference bounded by h, i.e. $\forall x_0$, $\exists \delta > 0$ and $L \in \mathbb{R}$ such that $\|x - x_0\| < \delta$ implies $\|\theta(x) - \theta(x_0)\| \leq L \,\|h(x) - h(x_0)\|$
4) $h_i(x)$ is an interpretable representation of x
5) k (the number of basis concepts) is small

Property 3 poses a Lipschitz-like continuity on the parametrizer $\theta(x)$, which is aimed to achieve local linearity and thus ensures that for inputs x_0 and x_1 close to each other, $\theta(x_0)$ and $\theta(x_1)$ should not differ significantly. Together with the reconstruction loss on the autoencoder (used for $h(x)$ to ensure property 4 and 5) the following loss is minimized:

$$\underbrace{\mathcal{L}_y(f(x), y)}_{\text{Classification loss}} + \underbrace{\mathcal{L}_h(x, \tilde{x})}_{\text{Reconstruction loss}} + \underbrace{\lambda \cdot \mathcal{L}_\theta(f)}_{\text{Robustness loss}} \tag{2}$$

where $\mathcal{L}_\theta(f) = \left\|\nabla_x f(x) - \theta(x)^\top J_x^h(x)\right\|$. Having stated the formal properties of a *SENN* we can also reformulate our goal with respect to these: we want to enhance fulfillment of property 4. Hence, in the following we will introduce two network architectures that can be plugged into the *SENN* as a conceptizer instead of the originally proposed autoencoder.

B. VaeSENN

The first extension, *VaeSENN* (see Fig. 1), uses supervised autoencoders [32] implemented via a β-Variational Autoencoder [β-VAE] [34]. The idea is to use two-stage autoencoding to separately encode parts of the input irrelevant and relevant to the prediction task. A first supervised variational autoencoder learns embeddings for those parts of the input data not relevant to the prediction task (e.g., style or rotation in handwriting). In the following, we refer to this first autoencoder as the style autoencoder and to the learned embeddings as styles. Encoded styles are then given as additional input to a second variational autoencoder (the conceptizer in the framework of a *SENN*) which learns concepts used for the prediction task.

The use of a supervised β-VAE in both autoencoders addresses all three shortcomings of autoencoders mentioned in the introduction. First, the variational approach enhances continuity of the latent space and thus hopefully also human interpretability. Second, the variational approach in both autoencoders enhances disentanglement of styles but more importantly also of concepts. Third, the additional input of styles allows the conceptizer to learn relevant encodings since it does not have to embed information irrelevant to the prediction task.

In order to learn styles (the task of the style autoencoder), we run a β-VAE on the inputs with one crucial addition: we will not only give the embeddings produced by the style encoder to the style decoder but also the label of each instance (i.e., we add one-hot encoded labels to the style decoder's input). This supervision procedure follows supervised adversarial autoencoders [S-AAE] presented in [32], with the difference that we do not use a GAN but a KL-divergence loss to enforce a prior distribution on the latent space. Concretely, β-VAE introduces a new hyperparameter β that weights the KL-divergence in the model objective. For larger β, the latent space will be encouraged to look unit gaussian, thus enforcing independence in the latent vector. Using a supervised β-VAE we are not only enforcing disentanglement within the latent space but also disentanglement of label information from all other latent factors of variation.

In the second stage, the same architecture is used to learn concepts. Again, the input images are processed by a probabilistic encoder $q_\phi(z|x)$. However, instead of additionally including the label information, we now give the style information learned during the first stage to the concept decoder. In this way, concepts are disentangled from all other latent factors regarding information irrelevant to the prediction task (styles). Equally important, the use of a β-VAE for the conceptizer also encourages disentanglement between concepts.

Ultimately, the prediction task leads to the following loss function to be minimized:

$$\mathcal{L} = \underbrace{\mathcal{L}_y(f(x), y)}_{\text{Classification loss}} + \underbrace{\mathcal{L}_h(x, \tilde{x})}_{\text{Reconstruction loss}} + \underbrace{\lambda \cdot \mathcal{L}_\theta(f)}_{\text{Robustness loss}} + \underbrace{\beta \cdot \mathbb{D}_{KL}(q_\phi(z|x)||p(z))}_{\text{KL Divergence}} \tag{3}$$

C. VSiamSENN

The second extension, *VSiamSENN* (see Fig. 2), uses a siamese network architecture [35] to learn embeddings invariant to in-class noise, and thus, enhancing interpretability-robustness (see section III-C). A triplet loss is used to ensure that embddings corresponding to images of the same class are mapped close to each other while images of a different class are explicitly mapped far away from each other. Further, in order to better shape the latent space we additionally use a variational scheme comparable with β-VAE [34]. In that sense, the training objective here slightly differs from [35]: while [35] aims to learn informative embeddings in a lower-dimensional space we aim to learn an informative posterior distribution of embeddings in this lower dimensional space. The proposed network architecture and training procedure is described in the following. During training our architecture takes as an input three images: (i) the image x_1 to be classified, (ii) in-class example x_2, a sampled image of the same class, and (iii) out-of-class example x_3, a sampled image of a different class. The three images are processed by the same probabilistic encoder $q_\phi(z|x)$ (the conceptizer in the *SENN* framework). [35, 36] use a predictor g for estimating the expectation of the latent encoding over the space of data augmentations by minimizing the negative cosine distance to its estimated expectation. Our approach differs, in that the predictor g instead estimates the expectation of the posterior mean corresponding to all images of the same class by minimizing negative cosine distance between sampled latent encodings from the posterior distributions and expected posterior mean of images of the same class. Additionally, to ensure class separation in the latent space, the absolute cosine distance between sampled latent representations from the posterior distributions and expected posterior means of images of different classes is minimized. We define negative cosine distance for two inputs x_i and x_j with $i, j \in \{1, 2, 3\}$:

$$\mathcal{D}(z_i, \bar{\mu}_j) = -\frac{z_i}{||z_i||_2} \cdot \frac{\bar{\mu}_j}{||\bar{\mu}_j||_2} \tag{4}$$

where z_i is a sample drawn from the posterior distribution $q_\phi(z|x_i)$ with mean $\mu_\phi(x_i)$ and $\bar{\mu}_j = g(\mu_\phi(x_j))$ is the estimated expectation of the posterior mean[2].

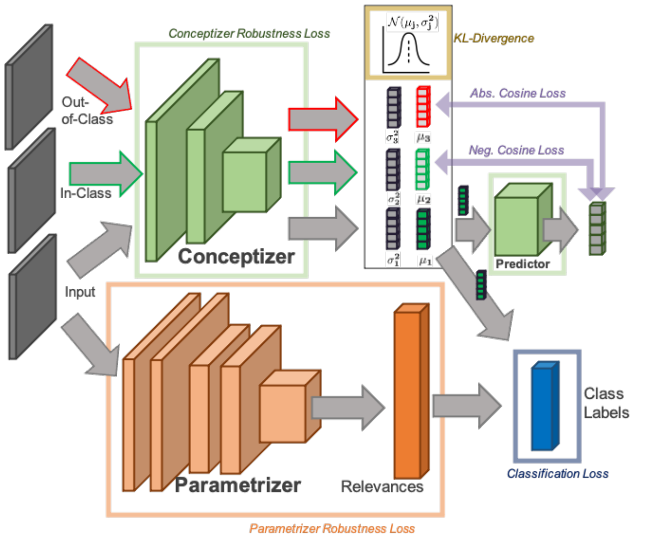

Fig. 2: VSiamSENN network architecture

In order to better shape the latent space, a KL-divergence loss is used to enforce a prior distribution (a unit Gaussian) on the latent space. Moreover, as suggested in [33], we introduce a new local-Lipschitz stability property for $\mu_\phi(x_1)$ with respect to x, in order to ensure that small changes in the input do not cause significant changes in concepts. This leaves us with the following loss to be minimized:

$$\begin{aligned}\mathcal{L} = &\frac{\mathcal{D}(z_1, \bar{\mu}_2)}{2} + \frac{\mathcal{D}(z_2, \bar{\mu}_1)}{2} + \frac{|\mathcal{D}(z_1, \bar{\mu}_3)|}{2} + \frac{|\mathcal{D}(z_3, \bar{\mu}_1)|}{2} \\ &+ \beta \cdot \left[\frac{\mathbb{D}_{KL}(q_\phi(z_1|x_1)||p(z))}{3} + \frac{\mathbb{D}_{KL}(q_\phi(z_2|x_2)||p(z))}{3} \right. \\ &\left. + \frac{\mathbb{D}_{KL}(q_\phi(z_3|x_3)||p(z))}{3}\right] + \eta \cdot ||\nabla_{x_1} \mu_\phi(x_1)||_2 \\ &+ \lambda \cdot \mathcal{L}_\theta(f) + \mathcal{L}_y(f(x_1), y_1)\end{aligned} \tag{5}$$

III. Experiments

For experiments we will evaluate SENN with default parameters as proposed in [29], and both *V-SiamSENN* and *VaeSENN* with comparable parameters where applicable. As datasets MNIST and CIFAR10 will be used. We found that MNIST worked best for comparing visualization of concepts due its limited complexity. To distinguish performance we found that it was useful to run experiments on a more complex data set, namely CIFAR10. In the following sections we will briefly evaluate accuracy of all three models (III-A). We will then elaborate in more detail on achieved interpretability with respect to continuity of latent space and disentanglement of latent factors to evaluate grounding and diversity of concepts (III-B). At last we

[2]As mentioned in [35, 36], a necessary component of the model is a stop-gradient operation. Therefore, (4) is modified by: $\mathcal{D}(s, p) = \mathcal{D}(s, \texttt{stopgrad}(p))$.

Robustness penalty	**0.0001**	**0.001**	**0.01**	**0.1**	**1**
SENN	98.99%	98.89%	98.68%	97.12%	10.30%
VaeSENN	99.24%	99.21%	98.91%	97.48%	10.30%
VSiamSENN	99.35%	99.15%	98.78%	97.54%	10.30%

TABLE 1: Test accuracy on MNIST

Robustness penalty	**0.0001**	**0.001**	**0.01**	**0.1**	**1**
SENN	83.65%	82.77%	79.11%	66.33%	10.00%
VaeSENN	84.27%	83.02%	79.04%	65.47%	10.01%
VSiamSENN	84.10%	82.83%	79.21%	67.24%	10.02%

TABLE 2: Test accuracy on CIFAR10

evaluate interpretability-robustness with experiments similar to [33] (III-C).

Training configuration We chose a similiar training configuration as [29]. For all three models: we set the number of concepts (the dimension of latent space) to be five, we trained 100 epochs on (i) MNIST dataset and 200 epochs on (ii) CIFAR10 dataset. For (i) we used the same convolutional architecture for the parametrizer and conceptizer with two convolutional and one fully connected layer[3]. For (ii) we used a VGG8 architecture [37] for the parametrizer and a convolutional network with two convolutional and one fully connected layer for the conceptizer[4]. For all models we used a learning rate of 2×10^{-4} and a batch size of 132. For *VaeSENN*: we used five styles on MNIST and 20 on CIFAR10. The style autoencoder was trained separately for 100 epochs with $\beta = 1 \times 10^{-2}$. Afterwards, we freezed the weights of the style autoencoder and trained the remaining model with $\beta = 1 \times 10^{-2}$. For *V-SiamSENN*: we set $\beta = 1 \times 10^{-3}$ and $\eta = 1 \times 10^{-4}$. In all models a sigmoid activation on the final layer of the parametrizer was used and a tanh activation on the concepts in *SENN*. No activation functions on the concepts are used in the variational settings.

A. Accuracy

We evaluated test accuracy of the above specified models on MNIST and CIFAR10 for robustness penalties ranging from $\lambda = 1 \times 10^{-4}$ to $\lambda = 1 \times 10^{0}$. Tables 1 and 2 show that both *VaeSENN* and *V-SiamSENN* have test accuracies comparable to *SENN*. Thus, in the following we will focus on arguing that our proposed models achieve enhanced interpretability while achieving similar accuracy compared to *SENN*.

B. Grounding and Diversity of Concepts

From Tables 1 and 2 one can see that the robustness penalty λ starts having greater influence on accuracy for $\lambda \geq 1 \times 10^{-2}$ on CIFAR10 while a conclusion of this kind is harder on MNIST. Thus, we will use the models

For: c - number of concepts, CL - concolutional layer, and FC - fully connected layer:

[3] $h(\cdot) : CL(10, 20) \rightarrow FC(c)$ and $\theta(\cdot) : CL(10, 20) \rightarrow FC(c \cdot 10)$

[4] $h(\cdot) : CL(10, 20) \rightarrow FC(c)$ and $\theta(\cdot) : CL(2^6, 2^7, 2^8, 2^9, 2^9) \rightarrow FC(2^8, 2^7, c \cdot 10)$

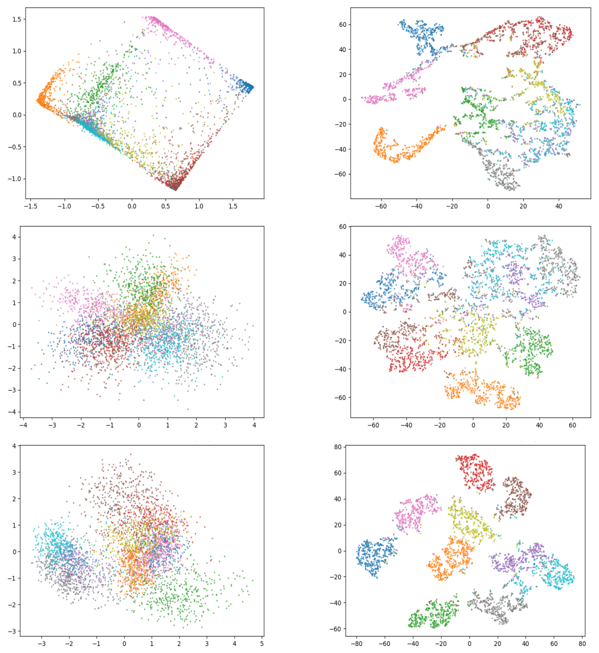

Fig. 3: Visualization of concepts in latent space for *SENN*, *VaeSENN* and *VSiamSENN* on MNIST. **Rows:** The first row shows results for *SENN*, the second for *VaeSENN*, the third for *VSiamSENN*. **Columns:** The first column uses PCA as a visualization method, the second t-SNE.

trained with $\lambda = 1 \times 10^{-2}$ on CIFAR10 and $\lambda = 1 \times 10^{-1}$ (the largest tested value before test accuracy deprecates to random guessing) on MNIST. We want to note, however, that the findings shown in this section hold for a much wider range of robustness penalties λ.

To analyze the diversity of concepts and continuity of latent space we qualitatively assess concepts using two visualization techniques: principal component analysis (PCA) [38] and t-distributed stochastic neighbor embedding (t-SNE) [39]. With using two distinct visualization techniques we hope to reduce the risk of coming to conclusions that result from the specific characteristics of the visualization technique rather than from the concepts themselves. Fig. 3 shows the results of visualizing 4000 learned concepts on a test data set for all three models.

It can clearly be seen that the latent space for *VaeSENN* and for *VSiamSENN* is of much nicer shape than that for *SENN*. Due to the use of a variational component for the former two this is not surprising. Especially the PCA plot for *SENN* indicates some discontiunities within the latent space. Overall, the use of a variational part within the conceptizer substantially increased interpretability of concepts by improving the continuity of the latent space. Further viusalizations can be found in appendix I and II.

Due to the distinct characteristics between the latent space of a *SENN* and *VaeSENN* and *VSiamSENN*, disentanglement

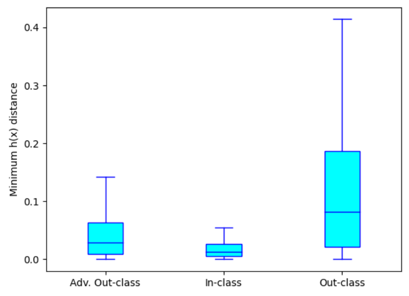

Fig. 4: Comparison on the distribution of adv. out-class, in-class and out-class distance for *SENN* on MNIST dataset with accuracy 98.89% ($\lambda = 1 \times 10^{-3}$).

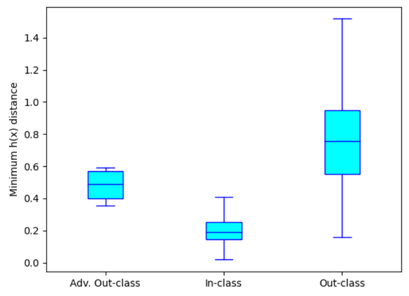

Fig. 5: Comparison on the distribution of adv. out-class, in-class and out-class distance for *VSiamSENN* on MNIST dataset with accuracy 99.03% ($\lambda = 1 \times 10^{-3}, \eta = 1 \times 10^{-4}$).

is harder to compare. On an absolute basis, Fig. 3 shows that both *VaeSENN* and *VSiamSENN* are clearly able to learn concepts that are disentangled by class (while not being disentangled too much which could jeopardize continuity of latent space).

C. *Interpretability-Robustness of Concepts*

[33] points out that interpretability-robustness is not fulfilled when small perturbations in input images cause significant changes in concepts while model output stays the same. Therefore, similar to [33], we compare the in-class, out-class, and adversarial out-class distance[5].

As shown in Fig. 4, for *SENN* we observe that in-class distance is smaller than out-class distance. Moreover, we observe that adv. out-class distance is similar to in-class distance. In comparison, for the *VSiamSENN* (Fig. 5) we observe that the adv. out-class as well as the (natural) out-class distances are significantly larger than the in-class distance. The results imply that adversarial perturbations are often successful in changing concepts of an image to concepts of an image belonging to a different class in the *SENN* but rarely for *VSiamSENN*. We observe that all distances for *VSiamSENN* are much higher than for *SENN*. We conclude that using a loss encouraging local-Lipschitz stability and a variational architecture for the conceptizer are able to highly improve interpretability-robustness.

IV. Discussion

We want to note that while it should be obvious that the choice for the robustness penalty λ is crucial for interpretability of the parametrizer it is also of high importance for interpretability of concepts. Setting λ too low allows the network to learn all relevant information via the parametrizer (see appendix IV) which diminishes interpretability of both the parametrizer (which in this case is no more interpretable than a usual neural network) and of the conceptizer (since the model does not need meaningful concepts for predictions). Further, sufficient modeling capacity of the conceptizer is central to learn concepts that can be disentangled. The conceptizer chosen in [29] seems to be too small to learn disentangled representations on CIFAR10 (see appendix I). We also observed for CIFAR10 that the conceptizer sometimes collapsed under poor initialization, resulting in constant concepts for images from different classes. At the same time, accuracy was unaffected due to the rich modeling capacity of the parametrizer. We suspect that when the conceptizer does not have sufficient capacity for class separation there exists a local minimum where the conceptizer tries not to interfere with the parametrizer and outputs close to constant concepts, making them uninterpretable. Thus, when using a *SENN* with the goal of interpretability we suggest to carefully inspect the model's training convergence and to also give greater attention to the complexity of the conceptizer.

During our process to find alternative conceptizers that achieve fulfillment of the three desiderata we also examined the use of invariant feature learning using adversarial training as in [41, 42]. This approach achieved similar (or only marginally worse) accuracy on MNIST and CIFAR10 and could be promising in terms of the third desiderata, relevance of concepts. Due to the limited capacity of this report and our focus on a nicely shaped latent space we aimed our attention on variational approaches here (implementation of the adversarial approach can also be found on the gitlab repository under the name InvarSENN).

While a *SENN* as originally proposed in [29] does not necessarily fulfill all desiderata stated for robust interpretability it gives rise to a ”plug-in” principle that could be used to build custom models with enhanced interpretability. The separation of conceptizer and parametrizer within the *SENN* allows researchers and practitioners to plug in different architectures and pose different interpretability requirements on each of them. In this report we showed two possible examples for such a ”plug-in” to enhance interpretability where including a variational element seemed to be especially advantageous.

[5]In-class distance: $\min_{x_t} ||h(x) - h(x_t)||_2$ s.t. x_t has the same label as x. Out-class distance: $\min_{x_t} ||h(x) - h(x_t)||_2$ s.t. x_t has a different label than x. Adversarial out-class distance: $\min_{x_t, x^*} ||h(x^*) - h(x_t)||_2$ s.t. x_t has a different label than x, $x^* \in B_\epsilon(x)$, and x^* has the same predicted label as x. We perform PGD attack [40] with $L_\infty(\epsilon = 0.2)$.

References

[1] G. Stiglic, P. Kocbek, N. Fijacko, M. Zitnik, K. Verbert, and L. Cilar, "Interpretability of machine learning-based prediction models in healthcare," *WIREs Data Mining and Knowledge Discovery*, vol. 10, no. 5, p. e1379, 2020.

[2] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad, "Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, (New York, NY, USA), p. 1721–1730, Association for Computing Machinery, 2015.

[3] C. Wang, B. Han, B. Patel, F. Mohideen, and C. Rudin, "In pursuit of interpretable, fair and accurate machine learning for criminal recidivism prediction," 2020.

[4] L. K. Jeff Larson, Surya Mattu and J. Angwin, "How we analyzed the compas recidivism algorithm," 2016.

[5] J. Chen, S. E. Li, and M. Tomizuka, "Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning," 2020.

[6] J. Kim and J. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," 2017.

[7] Z. C. Lipton, "The mythos of model interpretability," 2017.

[8] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," 2017.

[9] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," *Electronics*, vol. 8, p. 832, Jul 2019.

[10] C. Molnar, *Interpretable Machine Learning*. 2019. https://christophm.github.io/interpretable-ml-book/.

[11] R. Marcinkevičs and J. E. Vogt, "Interpretability and explainability: A machine learning zoo mini-tour," 2020.

[12] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLOS ONE*, vol. 10, pp. 1–46, 07 2015.

[13] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, p. 336–359, Oct 2019.

[14] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," 2019.

[15] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2014.

[16] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," 2017.

[17] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," 2016.

[18] D. Alvarez-Melis and T. S. Jaakkola, "A causal framework for explaining the predictions of black-box sequence-to-sequence models," 2017.

[19] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," 2017.

[20] F. Wang and C. Rudin, "Falling rule lists," 2015.

[21] C. Chen and C. Rudin, "An optimization approach to learning falling rule lists," 2018.

[22] T. Hastie and R. Tibshirani, "Generalized additive models," *Statist. Sci.*, vol. 1, pp. 297–310, 08 1986.

[23] C. Chen, K. Lin, C. Rudin, Y. Shaposhnik, S. Wang, and T. Wang, "An interpretable model with globally consistent explanations for credit risk," 2018.

[24] J. Feng and N. Simon, "Sparse-input neural networks for high-dimensional nonparametric regression and classification," 2019.

[25] Y. Y. Lu, Y. Fan, J. Lv, and W. S. Noble, "Deeppink: reproducible feature selection in deep neural networks," 2018.

[26] J. Jordon, J. Yoon, and M. van der Schaar, "KnockoffGAN: Generating knockoffs for feature selection using generative adversarial networks," in *International Conference on Learning Representations*, 2019.

[27] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions," 2017.

[28] M. Al-Shedivat, A. Dubey, and E. P. Xing, "Contextual explanation networks," 2020.

[29] D. Alvarez-Melis and T. S. Jaakkola, "Towards robust interpretability with self-explaining neural networks," 2018.

[30] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2014.

[31] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," 2016.

[32] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," 2016.

[33] H. Zheng, E. Fernandes, and A. Prakash, "Analyzing the interpretability robustness of self-explaining models," 2020.

[34] 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*, 2017.

[35] X. Chen and K. He, "Exploring simple siamese representation learning," 2020.

[36] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent: A new approach to self-supervised learning," 2020.

[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[38] H. Hotelling, *Analysis of a Complex of Statistical Variables Into Principal Components*. Warwick & York, 1933.

[39] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[40] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2019.

[41] A. Jaiswal, Y. Wu, W. AbdAlmageed, and P. Natarajan, "Unsupervised adversarial invariance," 2018.

[42] A. Jaiswal, Y. Wu, W. AbdAlmageed, and P. Natarajan, "Unified adversarial invariance," 2019.

Appendix I
Visualizing Models on CIFAR10

As mentioned in the discussion of the main report IV, visualization of concepts for all three models suggests that the default encoder chosen in [29] is too limited in its modeling capacity. PCA and t-SNE in Fig. 6 show that concepts are not disentangled. However, when we trained a *VSiamSENN* on CIFAR10 with a larger conceptizer $h(x)$ namely a VGG8 architecture [37] concepts are already disentangled after 25 training epochs (see Fig. 7).

Appendix II
Sampling from Latent Space for *VaeSENN*

In section III-B we have examined the shape of the latent space for all three models and have mainly looked at the distribution of concepts (mapped to a lower dimension via PCA or t-SNE). Another possibility to analyze interpretability of the latent space is to sample from it and see if the decoded samples resemble meaningful pictures. For *SENN* and *VaeSENN* (trained with robustness penalty $\lambda = 1 \times 10^{-1}$) we encoded 10 images randomly chosen from the test data set. We then perturbed concept vectors by adding gaussian noise (with $\mu = 0$ and $\sigma = 0.2$). Resulting decoded images can be seen in Fig. 8 and 10 for *SENN* and *VaeSENN* respectively. Sampling from the latent space for a *VaeSENN* visibly works better than for a *SENN* which is in line with our argumentation in III-B that the latent space of the former also is more interpretable.

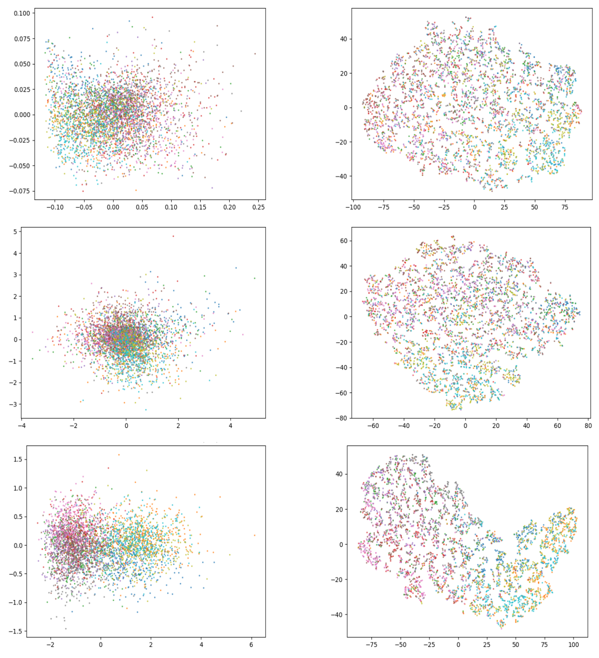

Fig. 6: Visualization of concepts in latent space for *SENN*, *VaeSENN* and *VSiamSENN* on CIFAR10. **Rows:** The first row shows results for *SENN*, the second for *VaeSENN*, the third for *VSiamSENN*. **Columns:** The first column uses PCA as a visualization method, the second t-SNE.

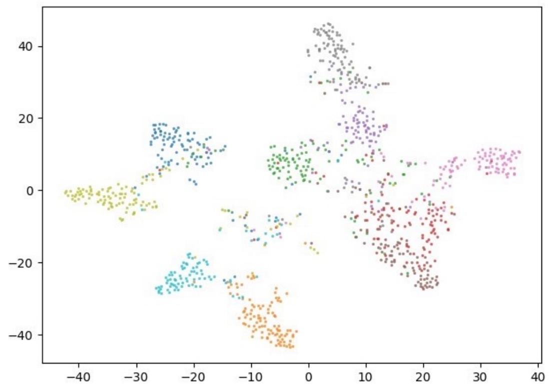

Fig. 7: t-SNE for *VSiamSENN* with a VGG8 encoder on CIFAR10 after 25 training epochs.

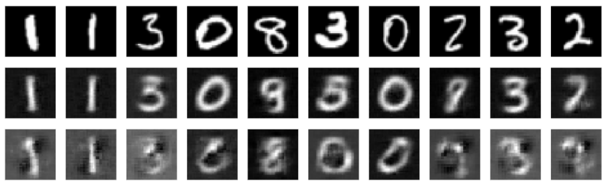

Fig. 8: Sampling from latent space in *SENN*: the first row shows the original input image, the second row shows the decoded image without added noise, and the third row shows the decoded image with added noise

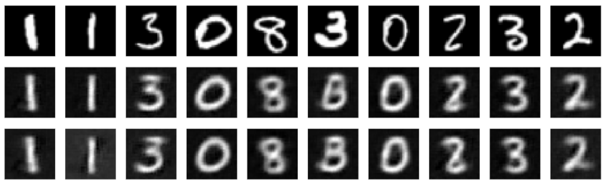

Fig. 9: Sampling from latent space in *VaeSENN*: the first row shows the original input image, the second row shows the decoded image without added noise, and the third row shows the decoded image with added noise

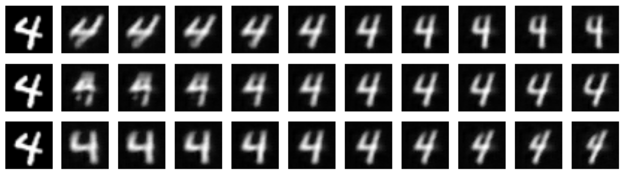

Fig. 10: Manipulating styles in *VaeSENN*: **Rows:** In the first, second, and third row we manipulated the first, second, and third entry of the latent vector of the style variational autoencoder. **Columns:** The first column shows the original image. From the second column onward we manipulate single latent value entries in a range from -2 to +2 (from left to right).

Appendix III
Manipulating Latent Vector of Styles Autoencoder

For the first stage of a *VaeSENN* we claimed that the style autoencoder learns different styles of an image of the same class. On MNIST we would expect this to be rotations or line thickness of handwritten numbers. To verify our claim we conducted the following experiment: a style vector with all entries except for one is set to zero and a class label (we arbitrarily chose ”4”) was given to the style decoder. We then varied the non-zero entry of the style vector in a range from -2 to 2. Decoded images are shown in Fig. 8. Indeed, the first and third row seem to show rotation of a number and the second the location of the horizontal line of a four (ranging from high to low from left to right).

Appendix IV
Training a Restricted *SENN*

To show that with a too low robustness penalty λ the separation of concepts and explanations through the conceptizer and parametrizer can fail we performed the following experiment: For a range of robustness penalties from $\lambda = 1 \times 10^{-4}$ to $\lambda = 1 \times 10^{0}$ we trained a *SENN* but set the concepts for every training instance all to one (keeping the number of concepts fixed at 5). Table 3 shows the test accuracy for both a normal *SENN* and a *SENN* restricted as just described. The results show that a too low robustness penalty can result in meaningless concepts even if test accuracy is high.

Robustness penalty	**0.0001**	**0.001**	**0.01**	**0.1**	**1**
SENN	98.99%	98.89%	98.68%	97.12%	10.30%
Restricted SENN	99.08%	99.08%	98.76%	97.04%	11.36%

TABLE 3: Test accuracy on MNIST. The first data row corresponds to a usual *SENN* (compare Table 1). The second data row corresponds to a restricted *SENN* where concepts are not learned but deterministically set to one for every train and test instance.