Contrastive Attribution with Feature Visualization

Eleanor Quint [1] Garrett Wirka [1] Stephen Scott [1] N. V. Vinodchandran [1] Tao Yao [1]

Abstract

Towards the goal of presenting contrastive explanations for the output of a classifier, we present a method for finding contrastive foils in feature space and then visualizing in data space for easy interpretation. To produce a simple classifier for this method, we present a model that combines decision trees with supervised variational autoencoders using our new *differentiable decision tree*. This allows for end-to-end optimization of a deep network to perform feature extraction from structured, high-dimensional data for classification by a single decision tree. Our experiments demonstrate that the resulting model is satisfactory in both classification and image generation and can explain a model's reasoning in answering a number of questions on classification and instance generation. Further experiments demonstrate that the model produces explanations using features that align with human-produced labels without any prior access to those labels.

1. Introduction

As use of machine learning techniques become more widespread, it is critical for developers, users, and other stakeholders to understand how learned models operate. This helps gauge trust in a model, utilize the model more effectively, and uncover its flaws (32). While many advances have been made to improve the interpretability of such models (13; 35; 29), existing approaches largely focus on attribution which limits the ability to explore contrastive cases, inputs that are similar to the original input data but have been modified to change the model output. Alternative methods study the internal mechanics of a model (14; 5), which leaves the onus with the human user to reconstruct and follow the model's reasoning. We propose that contrastive examples, presenting explanations in data space rather than in terms of the internals of the model, offer this kind of explanation (28). Literature in social science places emphasis on contrastive examples and "what if" analysis as a primary method of explainability (28).

Many methods exist for minimally perturbing data such that model output is changed, but their main purpose is adversarial attacks and thus make changes that are invisible to the human eye. By contrast, we propose a method of discovering contrastive examples by making changes in feature space using features that are both important in classification and whose changes are easy for a user to visualize in data space. To this end, we leverage single decision trees, which are simple, explainable classifiers. Since they are unsuitable for use with complex, high-dimensional data, we introduce a differentiable decision tree (DDT) to classify the latent variable of a variational autoencoder (VAE), which learns a factored, low-dimensional representation of data. This combined model is trained with gradient descent using a weighted sum of the VAE ELBO objective and the classification cross entropy. We call this combined model *Classifier+VAE,* or C+VAE, which we find to have good classification accuracy and generative log-likelihood, while optimizing both objectives simultaneously and using simple, feed-foward encoder and decoder.

The notion of explainability that we work with is similar to that of Ribeiro et al. (32). Both their work and ours focus on explaining a model's reasoning per instance as well as for the overall model. A key difference is that they focus on classification, whereas our results are for both classification and instance generation. Further, their approaches, while applicable to many types of learning models, require sampling instances near x (the instance to be explained) in terms of *interpretable features*, build an interpretable local model over those samples, and then use this proxy to explain the prediction on x. In contrast, we use the DDT as our classifier, which allows us to directly read out an explanation. Specifically, our use of the DDT helps guide a walk in latent space, and the results of the walk are utilized to explain classifications. Our model also offers explanations in a generative model, again using walks in latent space.

[1] Department of Computer Science and Engineering, University of Nebraska, USA. Correspondence to: Eleanor Quint <equint@cse.unl.edu>.

Proceedings of the Proceedings of the ICML 2020 Workshop Extending Explainable AI Beyond Deep Models and Classifiers, Copyright 2020 by the author(s).

1.1. Examples

Below are example questions that an explainable model might answer, categorized based on whether they relate to classification or generation and whether they concern individual instances or a model's behavior. We are interested in finding contrastive cases to answer each of these questions.

1. **Classification**
   (a) Instance-based: (i) Why is instance x classified as y? (ii) Why is instance x classified as y rather than y'? (iii) What changes to x would change its predicted class from y to y'?
   (b) Model-based: (i) What constitutes a class-y instance? (ii) What differentiates class y from class y'? (iii) Which classes most resemble each other to the discriminator?
2. **Generation**
   (a) Instance-based: (i) Why does instance x have low (or high) likelihood? (ii) Why does instance x, truly of class y, appear like class y'?
   (b) Model-based (i) What is the effect of latent variable z_i on generated instances, e.g.: (A) What happens if z_i is large (small)? (B) What features describe a class-y instance? (C) What does an ambiguous, class y/class y' hybrid look like? (ii) What classes y, y' are likely to have generated instances that resemble each other?

1.2. Our Contributions

We propose methods for generating contrastive answers to a representative subset of these questions in Sec. 3.3 and experimental results in Sec. 4.3. We propose one method to answer both 1(a)i and 1(a)ii together in order to make explicit the contrastive foil (28), which is implicit in the question. Specifically, we contribute the following. (1) A combined model, composed of a SVAE and our new differentiable decision tree, that both autoencodes input data, calculating a latent featurization $z = F(x)$ as well as an approximate inverse $g(z) \approx F^{-1}(z)$, and classifies using the latent featurization as input to the decision tree. Together, these features enable human-recognizable feature discovery using the features of the decision tree and allows for changes to data in feature space to be visualized in data space. (2) An interpretability algorithm *Walk* for generating contrastive examples in feature space (which is made straightforward if the classifier is simple) and then visualizing the contrastive examples in data space. We propose specific uses of the algorithm for answering *why*-questions a user might have about a classifier. (3) Experiments on data sets MNIST, Fashion-MNIST, and CELEBA, where we demonstrate feature discovery and apply *Walk* to answer questions locally about classifying particular data and globally about the classifier as a whole.

The rest of this paper is organized as follows. In Section 2 we give relevant background. Then in Section 3 we describe the SVAE and present our differentiable decision tree and our combined model, as well as approaches for generating explanations. Our experimental results appear in Section 4. Finally, we present related work in Section 5, and conclude in Section 6 with a discussion of future work.

2. Background

As part of our work, we extend the variational autoencoder (VAE) of Kingma and Welling (23) that optimizes the evidence lower bound (ELBO)

$$\mathcal{L}(\mathbf{x};\theta) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}(\log p(\mathbf{x} \mid \mathbf{z})) - KL(q(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})) \tag{1}$$

to a supervised VAE that optimizes

$$\mathcal{L}'(\mathbf{x}, y;\theta) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}(\log p(\mathbf{x} \mid \mathbf{z})) - KL(q(\mathbf{z} \mid \mathbf{x}) \| \mathcal{N}(\mathbf{z} \mid \mu_y, \mathbf{I})) \tag{2}$$

where y is a class label, μ_y is the posterior mean of class y, and π is a probability vector that may be pre-computed with the assumption that the training labels are iid. As μ_y is calculated empirically from the posterior, it can be initialized to small random values for all classes and updated regularly throughout training. This amounts to utilizing a Gaussian mixture as a prior distribution, similar to that of Dilokthanakul et al. (10). A key difference between their work and ours is that our use of class labels enhances training, obviating the need to marginalize over all classes to compute the K-L divergence. This helps avoid the over-regularization problem that they discuss in their paper, while achieving high sample quality in our generated images.

3. Model and Algorithm[1]

Now we describe our model and the **Walk** algorithm. The main contributions of our model are: (1) a differentiable decision tree (DDT), where we describe how to compute the expected probability distribution over predicted labels and use this to differentiate the expected loss of the tree; and (2) a combined VAE model, using the DDT and a supervised VAE, designed to learn a latent variable distribution suitable simultaneously for classifying data, generating data, and interpretable latent embedding (C+VAE).

3.1. Differentiable Decision Tree (DDT)

Decision trees are simple, explainable models. To utilize them in image analysis, we train a VAE for non-linear dimen-

[1] Code can be found at https://github.com/anonymous2020icml/icml2020submission

sionality reduction so the tree can classify a low-dimensional embedding. We develop a probabilistic generalization of decision trees, where each leaf returns a distribution over all classes: if instance $\mathbf{z}$ lands in leaf ℓ of tree T, then T returns distribution $P_T(y \mid \ell)$. As part of this generalization, we take a user-specified loss function $\text{loss}_T(\mathbf{z}, y)$ and compute the gradient of the expected loss $L_T = \mathbb{E}_{\mathbf{z}\sim D}[\text{loss}_T(\mathbf{z}, y)]$, where D is the distribution $q(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\mathbf{x}; \mu_{\mathbf{x}}, \sigma_{\mathbf{x}} I)$, where $\mu_{\mathbf{x}}$ and $\sigma_{\mathbf{x}}$ are the outputs of the encoder on input $\mathbf{x}$. This allows optimization of the distribution parameters for maximum likelihood w.r.t. an existing decision tree T. Thus, an embedding of the data may be learned in an EM-style manner, alternately learning a tree on the embedding produced by the parameters of a deep encoder and optimizing the embedding parameters to better fit the class-based partitioning induced by the learned decision tree.

3.2. Our Combined Model

The supervised VAE and decision tree inference can be used to both classify *and* reconstruct data from the encoded parameters of its latent distribution. Although an embedding could be learned by only optimizing classification accuracy of the decision tree, the additional reconstruction objective ensures that the learned representation is contains information for other, downstream uses, and is suitable for visualization. Our new architecture C+VAE (Classifier+VAE) uses a deep encoder network to parameterize a Gaussian distribution, which is then used as the input for classifying with the DDT and to reconstruct the encoded data with a deep decoder network. Generally, these modifications can also be applied to existing VAE architectures when label information is available. The C+VAE training procedure begins by randomly initializing the encoder/decoder parameters and encoding the training data to initialize the decision tree and aggregate posterior class means. Training then proceeds by running several epochs of gradient updates before re-training the decision tree and updating the aggregate posterior class means until the model converges. The optimization function of our combined model consists of a linear combination of the objective of the supervised VAE and the expected error of the current decision tree T. The modified VAE objective of the C+VAE to be minimized is

$$f(\mathbf{x}, y; \theta) = -\mathcal{L}'(\mathbf{x}, y; \theta) + \gamma L_T \ . \tag{3}$$

3.3. Explaining Model Reasoning

To answer Section 1.1's questions, our model uses the base algorithm **Walk**(z, z', δ), which visualizes data between latent points z and z' using step size[2] δ. After each step, **Walk** decodes the intermediate latent point to data space. This sequence of intermediate instances is used in the model's explanation. E.g., by highlighting image differences along the walk, the model can explain what the important differences are in data space, especially those regarding tree decision boundaries. For example, to answer **Question 1(a)(iii)** (What changes to x would change its classification from y to y'?), first identify the (class-y) tree leaf ℓ that z filters to, then find the lowest common ancestor of ℓ and any leaf ℓ' predicting class y'. Let z' be the mean of the training instances that filter to ℓ'. Then walk from z to z' using the dimensions tested in the tree on a shortest path from ℓ to ℓ'.

To answer **Question 1(b)(ii)** (What differentiates classes y and y'?), first find a pair of leaves (ℓ, ℓ') such that ℓ predicts y and ℓ' predicts y' (if there are multiple such pairs, choose the pair with minimal tree distance from ℓ to ℓ'). Let z and z' be points in latent space from ℓ and ℓ' that are nearest each other. Walk from z to z' using the dimensions tested in the tree on a shortest path from ℓ to ℓ'. To answer **Question 2(a)(i)** (Why does x have high/low likelihood?), first encode x as its latent representation z, then find the class mean μ_y nearest z. Walk from z to μ_y across all latent dimensions. To answer **Question 2(b)(i)(C)** (What does a class y/class y' ambiguous instance look like?), walk from μ_y to $\mu_{y'}$ across all latent dimensions and return the point z midway between them. To answer **Question 2(b)(ii)** (What classes y and y' are likely to have similar instances?), first choose the pair of classes (y, y') with the closest class means μ_y and $\mu_{y'}$. Walk from μ_y to $\mu_{y'}$ across all latent dimensions. It is possible to answer the other questions presented in Section 1.1 using similar calls to **Walk** using latent features the decision tree makes the relevant choices on. An exception is Question 1(b)(iii) (Similar classes to the discriminator), in which the answer simply comes from identifying a pair of classes whose leaves are closest to their LCA.

4. Experiments

Our experiments are designed to empirically study the following claims: (1) the supervised VAE effectively takes advantage of class labels to improve generative performance; (2) the C+VAE classifies competitively with other tree-based embedding methods while simultaneously maintaining a generative model competitive with the literature; (3) the differentiable decision tree is an explainable classifier that, when used in C+VAE and with **Walk**, can explain its relevant discriminitive features in terms of data space attributes; and (4) the generative model of our approach can be explained by running **Walk** in latent space.

We used MNIST and Fashion-MNIST. We applied the C+VAE modifications to a standard VAE (23) with two-layer MLPs of 500 hidden units as encoder and decoder models and a 50-dimensional latent variable $\mathbf{z}$ without importance sampling or an autoregressive prior. CART from scikit-learn was used to train the decision tree, regularized by limiting the decision tree depth to 8. Unless otherwise noted, we

[2] δ is calculated as the distance between z and z' divided by the number of steps in all cases except when addressing questions 1(b) and 2(b), when standard deviations of the aggregate posterior in question are used.

used $\gamma = 100$ in the objective function of the C+VAE (Equation (3)) and $n = 3$ epochs of gradient steps between each update of both the decision tree and the aggregate posterior parameters. Adam (21) was used for optimization and the data was not pre-processed or augmented.

4.1. Evaluating the Supervised VAE

Table 1 lists classification results of a number of tree-based and VAE-based models on MNIST (C+VAE's error on Fashion MNIST was 7.12%). The M1:SVAE+CART model trains the supervised VAE to convergence, and then trains a standard decision tree with CART to classify its latent code in the style of M1 (22). The intent is to highlight the effect of training without the backpropagated classification loss from the DDT. C+VAE sans reconstruction zeros the reconstruction loss term of the objective function to highlight the effect of training a model that only learns an embedding suitable for classification with the DDT. We compare our results to those of boundary trees (BT) with embedding (39), another tree-based interpretable classifier, as well as M1+M2 (22), and the Ladder Network (30), which are other VAE-based classifiers.

We then evaluate the efficacy of leveraging label data in a supervised VAE in generation, equivalent to using C+VAE with $\gamma = 0$. The flexibility of a Gaussian mixture and the fact that the data is clearly multi-modal both contribute to the SVAE log-likelihood of -102.77, better than the log-likelihood of -109.56 using our implementation of the VAE, which uses an unmodified Gaussian prior. We expect this difference to be the result of using a flexible prior that is more faithful to the true prior. This flexibility is similar to that seen in techniques like normalizing flows (31), but modifies the prior rather than the posterior and uses the additional information provided by label information.

4.2. Evaluating the C+VAE

We next empirically evaluate C+VAE for classification and generation. As a baseline, we first examine how well a standard (non-differentiable) decision tree from CART can classify when the data is encoded by a supervised VAE (but with no error feedback: $\gamma = 0$). This is similar to M1 (22) with a different VAE. In Table 1, row M1:SVAE+CART shows that without the error feedback from the tree, it is unlikely that the embedding will be useful in classification by a decision tree. This motivates our use of the DDT. To test the benefit of reconstruction in learning an embedding that can be classified well, we ran a test in which we switched off the reconstruction error feedback in learning (removing the first term of Equation (2)). In Table 1, row C+VAE sans reconstruction shows a significant improvement in classification error over M1:SVAE+CART, but still quite high.

Table 1. MNIST classification error for fully supervised tree- and VAE-based models. C+VAE's error on Fashion MNIST was 7.12%.

Model	Error
M1:SVAE+CART	37.09%
C+VAE sans reconstruction	7.30%
C+VAE	1.98%
BT w/embedding	1.85%
M1+M2	0.96%
Ladder Network	0.57%

Row C+VAE in Table 1 shows our combined method's performance with $\gamma = 1000$. We see a large improvement in classification error over C+VAE sans reconstruction, demonstrating the importance of both types of feedback in training. While C+VAE's classification performance is worse than in the literature, it's still competitive, despite simultaneously optimizing both classification and log-likelihood. Also, C+VAE's log-likelihood of -106.83 is comparable to the -109.56 from our parallel implementation of the VAE, which uses the same encoder-decoder pair as C+VAE. A more powerful encoder or the use of more recent techniques (e.g., normalizing flows, importance weighting, etc.) might improve both error and log-likelihood even further.

4.3. Evaluating the Explainability of the DDT

The decision trees learned by C+VAE on MNIST and Fashion-MNIST (both omitted for space) were then used to answer answer some of the questions from Section 1.1.

Question 1(a)(iii) (What changes to x would change its predicted class from y to y'?) In our example, x is an MNIST digit "8" mispredicted as a "3", and the question is what it would take for the model to correctly classify it. The leftmost digit in Figure 1 is the original digit x and the remaining ones are the result of walking from x's representation z along dimension 20 (the attribute tested by the parent of Class 8's leaf and Class 3's leaf) to the mean of the training instances in the leaf predicting class "3". The model's answer to Question 1(a)(iii) is, "To change x from class "3" to "8", subtract the pixels colored green and add those colored magenta."

Question 1(b)(ii) (What differentiates class y from class y'?) In this example, class y is "9" and class y' is "4". Figure 2 shows the results of walking from the mean of the training instances in the "9" leaf of the tree to its "4" leaf. Thus, the model's answer to Question 1(b)(ii) is, "The differences transitioning from class "9" to class "4" are to subtract the pixels in green and add those in magenta."

Question 1(b)(iii) (What classes resemble each other to the discriminator?) Here, we find pairs of classes near their lowest common ancestor in the tree, e.g., "4"/"9".

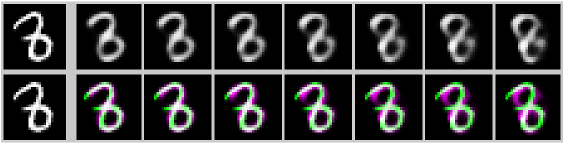

Figure 1. What changes to this input (left) would change its predicted class from “3” to “8” under the model? The bottom row of images show which regions of the image are salient for discriminating between instances of “3” and “8”.

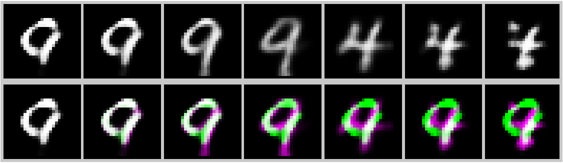

Figure 2. What differentiates class “9” from class “4”? The center image is the reconstruction of the point halfway between μ_4 and μ_9 in latent space. The left and right images walk along dimension 26 towards $\mu_\mathbf{4}$ and $\mu_\mathbf{9}$, respectively (image changes highlighted).

Question 2(a)(i) (Why does x have low likelihood?) Figure 3 shows instance x whose latent representation z is not near any class mean in the generative model. Thus, the model’s answer to Question 2(a)(i) is, “Instance x is unlikely since it does not resemble its most similar class nor the class of its label.”

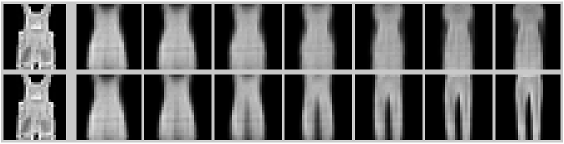

Figure 3. Why does this instance of “trouser” have low likelihood? The model’s prediction of “dress” and the actual label “trouser” are both ill-fitting. The rightmost images show the generated average “dress” (top) and “trouser” (bottom). The authors note the similarity of the intermediate images with womens’ jumpsuits.

Question 2(b)(i)(C) (What does a $y\,/\,y'$ class hybrid look like?) Figure 4 shows decoded images from the means of classes “dress” and “coat”, along with the hybrid decoded from the mean of means.

4.4. Correlation to Human Explanations

Our final experiment is to determine how well C+VAE can, without the use of auxiliary information, discover explanations that can be couched in terms originally derived by human users. Specifically, we investigated the following questions: (1) Does a z (latent) dimension and a threshold correlate well with a human-defined attribute, on its own?; and (2) How do these discovered explanations match with CART’s? We used the CelebA[3] dataset (37), which has over $200,000$ images of over $10,000$ celebrities. Further, each image has values of each of 40 binary attributes. For each binary attribute $y_c \in \{$“No Beard”, “Smiling”, “Male”, “Brown Hair”$\}$, we let y_c be the class label to be learned, training CART on the 39 other binary attributes as well as training C+VAE on only the images (no binary attributes). We used the same train/test split that was already in place in the CelebA dataset. For each y_c, let T_c^C denote the CART tree learned and T_c^V denote the C+VAE tree. Table 2 presents the test set accuracy of T_c^C vs T_c^V for each c. Observe that T_c^V is at least as accurate as T_c^C. We then measured how well the binary attributes correlated with the latent-space features evaluated by T_c^V. For each c, let $\bar{Y}_c$ be the set of 39 other attributes, excluding y_c. Then for each $y_o \in \bar{Y}_c$, we computed across the test set the ϕ coefficient (38) between y_o and the test at the root of T_c^V. The ϕ coefficient, also known as the mean square contingency coefficient, measures how much two binary variables associate with each other. Table 2 presents for each y_c, all $y_o \in \bar{Y}_c$ with $|\phi| \geq 0.4$, which is considered a strong relationship[4] (sign of ϕ is irrelevant, since it changes with the inequality at the root of T_c^V). We now discuss each of the y_c class attribute values with respect to Questions (1) and (2).

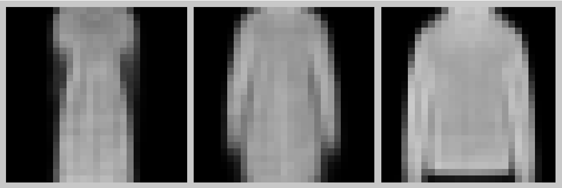

Figure 4. What does an ambiguous, class “dress”/class “coat” hybrid look like? Left image decoded from μ_dress, right from μ_coat, and the center from the point halfway between them.

“No Beard”: (1) *(Correlation to human-defined attributes)* All four y_o attributes with $|\phi| \geq 0.4$ have obvious relationships to facial hair. (2) *(Correlation to T_c^C)* All four y_o attributes with $|\phi| \geq 0.4$ appeared in T_c^C within distance 3 of the root, and all except Mustache within distance 2. The only CART attribute near the root and absent from Table 2 is 5 o’clock Shadow, which had $\phi = 0.38$.

“Smiling”: (1) Clearly, “Mouth Slightly Open” is a valid justification to classify as “Smiling”. Further, “High Cheekbones” is arguable as an explanation since a large smile gives the appearance of high cheekbones. (2) T_c^C had “High Cheekbones” at the root and “Mouth Slightly Open” tested at both the root’s children. This exactly correlates with the

[3] http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[4] https://www.statisticshowto.datasciencecentral.com/phi-coefficient-mean-square-contingency-coefficient/

Table 2. Test set ϕ coefficient between top values of y_o and z value at root of C+VAE-trained tree, for select values of y_c. "CART" accuracy is for T_c^C and "C+VAE" is accuracy for T_c^V.

			Accuracy	
y_c	y_o	ϕ	CART	C+VAE
No Beard	Sideburns	-0.47	0.94	0.94
No Beard	Goatee	-0.46		
No Beard	Male	-0.45		
No Beard	Mustache	-0.41		
Smiling	High Cheekbones	0.62	0.85	0.93
Smiling	Mouth Open	0.52		
Male	Wearing Lipstick	-0.79	0.93	0.96
Male	Heavy Makeup	-0.64		
Male	No Beard	-0.49		
Brown Hair	Black Hair	-0.51	0.82	0.82

only two y_o attributes with $|\phi| \geq 0.4$.

"Male": (1) "Wearing Lipstick" and "Heavy Makeup" are solid explanations of predicting "Male" as false in the CelebA dataset. On the other hand, "No Beard" is likely a false indicator, since it is true in 83% of the data set. (2) T_c^C had "Wearing Lipstick" at the root and "No Beard" and "Heavy Makeup" at the root's children. This exactly correlates with the only three y_o attributes with $|\phi| \geq 0.4$.

"Brown Hair": (1) Clearly, the presence of "Black Hair" is an explanation of not "Brown Hair". Interestingly, what is lacking in Table 2 is "Blond Hair", which would also be a good explanation, but had only $\phi = 0.29$. (2) T_c^C had "Black Hair" at its root, with Blond Hair and a leaf as the root's children. The root's attribute matches our top correlates strongly with our y_o attribute.

Note that trees T_c^V were learned **with no knowledge of the attributes in $\bar{Y}_c$,** i.e., C+VAE trained with only images. Despite this, C+VAE was still capable of learning trees that can explain in terms of human-defined attributes in $\bar{Y}_c$.

5. Related Work

Work using deep networks for representation learning with decision trees includes Deep Neural Decision Forests (24), which stochastically make routing decisions through a tree according to the outputs of a deep convolutional network. This achieved good classification performance, but it is unclear how to interpret the proposed classification process. To make the tree differentiable, our method of integrating a distribution over the tree's decision regions is a novel approach. Another tree-based method uses differentiable boundary trees to learn an embedding suitable for k-nearest neighbor (39). The learned representation allows a small, interpretable boundary tree to classify effectively, similar to our technique. The classification accuracy of the technique marginally outperforms our combined model, but the C+VAE also acts as a generative model and does not suffer from the significant complexity of having to use dynamically constructed computation graphs. Decision sets have been shown to be effective interpretable classifiers (26). It would be interesting to adapt our DDT approach to differentiable decision sets to train and operate in the latent layer.

Other work in classifying the latent codes produced by a VAE includes Kingma et al. (22), whose M1 semi-supervised model learns to classify from the latent embedding similarly to our combined classifier. However, M1 trains the discriminator separately from the VAE and lacks explainability as the class separation is performed solely by a black-box discriminator. The M2 model is similar to the supervised VAE, but doesn't change the VAE prior. Dilokthanakul et al. (10) present a Gaussian Mixture Variational Autoencoder to learn a class-focused latent representation. Our work assumes a supervised, rather than the GMVAE's unsupervised environment, allowing the classifying modification to the VAE framework to be more explainable.

Related to explainability, our work most resembles that of Ribeiro et al. (32), described in Section 1. Much other recent work in deep model interpretability is rooted in learning *disentangled latent representations* (2; 6; 18; 17; 4; 17; 12; 19; 9; 25; 11; 3; 8; 33; 27; 34; 20; 7; 15). Generally, such approaches attempt to train models towards maintaining latent representations where a single latent variable ties to a single attribute in data space, such as object class, color, position, rotation, size, etc. Related approaches ILVM and JLVM (1) specify their interpretable representations outside of the latent variables used in the core model, and then learns a bijection between them. Finally, Vedantam et al. (36) specify attribute vectors defined over data space and train a product-of-experts model to generate instances based the presence of subsets of these attributes. These disentanglement approaches work to capture semantics of the latent representation in terms of features in the data space, but in contrast to our work, they do not attempt to directly *explain* how such features relate to the model's reasoning in classification or generation. It should be straightforward to combine some of these (e.g., β-VAE) with our model.

Analyzing an instance to determine how to change its predicted class is related to adversarial learning (16), where one modifies an instance in a way imperceptible to humans, but the classifier changes its prediction. A key difference is that our approach aims to make perceptible changes to the input, so the user can follow the explanation.

6. Future Work

Future work includes applying our approach to other data with more powerful encoders and decoders to see how performance is affected. We will also look into extending our approach to handle unlabeled data in applications such as semi-supervised learning and clustering. Other future work includes applying disentanglement of latent variables such as β-VAE.

Acknowledgements

This work was supported by DoD contract FA4600-18-D-9001-FA460018FU904. It was completed utilizing the Holland Computing Center of the University of Nebraska, which receives support from the Nebraska Research Initiative. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

References

[1] T. Adel, Z. Ghahramani, and A. Weller, *Discovering interpretable representations for both deep generative and discriminative models*, in ICML, 2018, pp. 50–59.

[2] Y. Bengio, *Learning deep architectures for AI*, Foundations and Trends in Mach. Learning, 2 (2009), pp. 1–127.

[3] D. Bouchacourt, R. Tomioka, and S. Nowozin, *Multi-level variational autoencoder: Learning disentangled representations from grouped observations*, in AAAI, 2018.

[4] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, *Understanding disentangling in β-VAE*, CoRR, abs/1804.03599 (2018).

[5] N. Cammarata, S. Carter, G. Goh, C. Olah, M. Petrov, and L. Schubert, *Thread: Circuits*, Distill, (2020). https://distill.pub/2020/circuits.

[6] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, *InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets*, in NIPS 29, 2016, pp. 2172–2180.

[7] J. Chou, C. Yeh, H. Lee, and L. Lee, *Multi-target voice conversion without parallel data by adversarially learning disentangled audio representations*, in Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, 2018, pp. 501–505.

[8] E. L. Denton and V. Birodkar, *Unsupervised learning of disentangled representations from video*, in NIPS 30, 2017, pp. 4417–4426.

[9] G. Desjardins, A. C. Courville, and Y. Bengio, *Disentangling factors of variation via generative entangling*, CoRR, abs/1210.5474 (2012).

[10] N. Dilokthanakul, P. A. M. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, *Deep unsupervised clustering with Gaussian mixture variational autoencoders*, 2017.

[11] C. Donahue, A. Balsubramani, J. McAuley, and Z. C. Lipton, *Semantically decomposing the latent spaces of generative adversarial networks*, CoRR, abs/1705.07904 (2017).

[12] E. Dupont, *Joint-VAE: Learning disentangled joint continuous and discrete representations*, CoRR, abs/1804.00104 (2018).

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

[14] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, *Explaining explanations: An overview of interpretability of machine learning*, in 2018 IEEE 5th International Conference on data science and advanced analytics (DSAA), IEEE, 2018, pp. 80–89.

[15] Y. Gong and C. Poellabauer, *Towards learning fine-grained disentangled representations from speech*, CoRR, abs/1808.02939 (2018).

[16] I. J. Goodfellow, J. Shlens, and C. Szegedy, *Explaining and harnessing adversarial examples*, in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

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

[18] T. Hinz and S. Wermter, *Inferencing based on unsupervised learning of disentangled representations*, CoRR, abs/1803.02627 (2018).

[19] W. Hsu, Y. Zhang, and J. R. Glass, *Unsupervised learning of disentangled and interpretable representations from sequential data*, in NIPS 30, 2017, pp. 1876–1887.

[20] S. Jain, E. Banner, J. van de Meent, I. J. Marshall, and B. C. Wallace, *Learning disentangled representations of texts with application to biomedical abstracts*, CoRR, abs/1804.07212 (2018).

[21] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, CoRR, abs/1412.6980 (2014).

[22] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, *Semi-supervised learning with deep generative models*, 2014.

[23] D. P. Kingma and M. Welling, *Auto-encoding variational Bayes*, in ICLR, 2014.

[24] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulò, *Deep neural decision forests*, in ICCV, 2015, pp. 1467–1475.

[25] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum, *Deep convolutional inverse graphics network*, in NIPS 28, 2015, pp. 2539–2547.

[26] H. Lakkaraju, S. H. Bach, and J. Leskovec, *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, San Francisco, CA, USA, August 13-17, 2016, 2016, pp. 1675–1684.

[27] M. Mathieu, J. J. Zhao, P. Sprechmann, A. Ramesh, and Y. LeCun, *Disentangling factors of variation in deep representations using adversarial training*, CoRR, abs/1611.03383 (2016).

[28] T. Miller, *Explanation in artificial intelligence: Insights from the social sciences*, Artificial Intelligence, 267 (2019), pp. 1–38.

[29] A. Nguyen, J. Yosinski, and J. Clune, *Deep neural networks are easily fooled: High confidence predictions for unrecognizable images*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 427–436.

[30] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko, *Semi-supervised learning with ladder networks*, in NIPS 28.

[31] D. J. Rezende and S. Mohamed, *Variational inference with normalizing flows*, in ICML 2015.

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

[33] A. Saha, M. Nawhal, M. M. Khapra, and V. C. Raykar, *Learning disentangled multimodal representations for the fashion domain*, in 2018 IEEE Winter Conf. on App. of Comp. Vision, 2018, pp. 557–566.

[34] N. Siddharth, B. Paige, J. van de Meent, A. Desmaison, F. D. Wood, N. D. Goodman, P. Kohli, and P. H. S. Torr, *Learning disentangled representations with semi-supervised deep generative models*, CoRR, abs/1706.00400 (2017).

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

[36] R. Vedantam, I. Fischer, J. Huang, and K. Murphy, *Generative models of visually grounded imagination*, CoRR, abs/1705.10762 (2017).

[37] S. Yang, P. Luo, C. C. Loy, and X. Tang, *From facial parts responses to face detection: A deep learning approach*, in IEEE International Conference on Computer Vision (ICCV), 2015.

[38] G. U. Yule, *On the methods of measuring association between two attributes*, Journal of the Royal Statistical Society, 75 (1912), pp. 579–652.

[39] D. Zoran, B. Lakshminarayanan, and C. Blundell, *Learning deep nearest neighbor representations using differentiable boundary trees*, 2017.