Explaining Black Box with visual exploration of Latent Space

F. Bodria[†1], S. Rinzivillo[2], D. Fadda[2], R. Guidotti[3], F. Giannotti[1], D. Pedreschi[3]

[1]Scuola Normale Superiore, Pisa, Italy, [2]ISTI-CNR,Pisa,Italy, [3]Università degli studi di Pisa, Dipartimento di Informatica, Pisa, Italy

Abstract

Autoencoders are a powerful yet opaque feature reduction technique, on top of which we propose a novel way for the joint visual exploration of both latent and real space. By interactively exploiting the mapping between latent and real features, it is possible to unveil the meaning of latent features while providing deeper insight into the original variables. To achieve this goal, we exploit and re-adapt existing approaches from eXplainable Artificial Intelligence (XAI) to understand the relationships between the input and latent features. The uncovered relationships between input features and latent ones allow the user to understand the data structure concerning external variables such as the predictions of a classification model. We developed an interactive framework that visually explores the latent space and allows the user to understand the relationships of the input features with model prediction.

CCS Concepts

• ***Human-centered computing*** → ***User interface design; Visualization techniques;***

1. Introduction

Recent years have witnessed the rise of decision support systems based on the adoption of opaque Artificial Intelligence models that have ensured high accuracy labeling [Kra91] by exploiting a mapping of the input instances into their internal representation, called *latent space*. However, the complexity of this mapping hides the internal decision criteria of the model. With the term *black box* we refer to an opaque decision model (e.g., random forest, ensemble classifiers, neural networks (NN), etc.). The diffusion of these models has boosted the research on eXplainable AI techniques [GMR*18, BGG*21]. In this paper, we present a classifier-agnostic approach for explaining the outcome of a black box based on two steps: 1) an autoencoder is used to map multi-dimensional input features to a bi-dimensional space; 2) a widely used explanation technique, i.e., SHAP [LL17], is exploited to measure the relevance of each attribute of the input to the position in the latent space. The combination of the two techniques enables the effective visualization and exploration of the contribution of each original feature to the latent space. Modifications on a single input feature generate a spatial offset on the latent space visualization, allowing cognitive friendly navigation of the joint modification of multiple attributes.

An autoencoder is a NN [TSK05] that mimics a data generator for the input data. It is composed of two parts: the *encoder*, which transforms an instance of the input space into a point in the latent space, and the *decoder* [GBCB16], which takes a point in the latent space and maps it to the input space. This latent space projection can be seen as a compression of the input space's original dimensions, focusing on valuable but unknown relationships among features. Nowadays autoencoders are employed in a wide variety of tasks ranging from recommendation systems [THZ*18] to image denoising [BSA20] and data generation [YLK20]. We chose a specific family of autoencoder architectures, called Conditional Variational Autoencoder (CVAE), which exploits the outcome of a classification model to generate a latent space where data are grouped by the similarity of their attributes and the prediction label assigned by the black box. We exploit this mapping to encode in the latent space positions the classification provided by the black box to be explained.

However, the reasons why an instance is mapped to a specific position remains unknown since the autoencoder model does not reveal linear and non-linear relationships with the data properties. Our contribution focuses on building an explanation for the latent space mapping. The explanation method is based on the SHAP (SHapley Additive exPlanations) [LL17] technique, which returns the contribution of each input feature to the final classification. We apply the SHAP method to the mapping model, yielding a sum of linear contribution for both the x and y components. If we consider a single input feature, its contribution is approximated by SHAP as a line segment passing through a reference point (corresponding to the average expected value): increasing or decreasing the input feature value produces an offset of the mapped point on the opposite

† This work has been partially supported by the European Community Horizon 2020 programme under the funding schemes: G.A. 834756 *XAI* (xai), G.A. 871042 *SoBigData++* (sobigdata), G.A. 952026 *Humane AI Net* (humane-ai), G.A. 825619 *AI4EU* (ai4eu).

DOI: 10.2312/evs.20221098

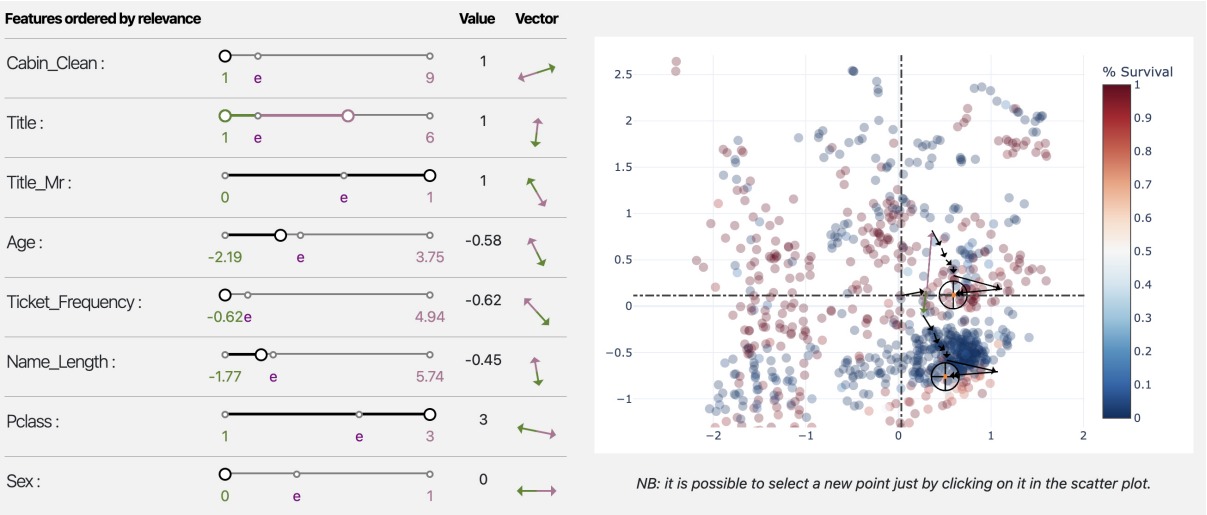

Figure 1: *Interactive framework for the titanic dataset. We have several sliders of the top ten most informative features on the left, ordered by relevance. The violet letter e on the bottom of every slider is the expected value for that particular feature. The two columns on the right of the sliders are in order: the feature's value and the shap score's vector representation. On the right, we have a graph of the latent space learned. The points are labeled red and blue according to the prediction label. The point taken into analysis is selected using a black viewfinder. The expected position is the origin of the two gray axes, and the black vectors represent the contributions of each feature, the sum of which lead from the expected value to the actual point. By modifying the Title feature from 1 to 4, we can see how the position of the point is changed in the latent space following the shap vector. By moving above the expected value of the feature, the effect in the latent space is given by the shap vector highlighted in pink.*

directions of the segment. Since we are considering multiple input attributes, the single contributions can be considered as a concatenation of offsets w.r.t. the previous features: the SHAP segment of feature i is translated to have the expected point aligned with the actual projected position of the contribution of feature $i-1$. The order of the features is a result of the SHAP method, going from the most relevant to the less relevant (more details in Section 4). These contributions are represented as vector arrows that sum to identify a point in the latent space. We leverage interactivity [SRFB*20] to let the user vary the values of the input space (i.e., the original attributes of the instance) to explore the resulting mapping in the latent space display. In this way, she can explore how variation in the input may influence the mapping of the autoencoder and, hence, the relevance of features. The variation of a feature f influences the final mapping of the point, since it modifies the magnitude of the vector corresponding to f.

The rest of the paper is organized as follows: Section 2 discusses related works. Section 3 recalls the notions needed to understand the proposed methodology which is illustrated in Section 4. Section 5 presents the interactive framework. Finally, Section 6 concludes the paper by discussing known limitations and proposing future research directions.

2. Related Work

In recent years several types of autoencoders models have been developed [N*11, KW13, SLY15, ZSE17] with different types of loss functions, integrating the use of labels or not. However, all these variations have not solved the interpretability of the latent space. A common technique to analyze the latent space is to do clustering on it and analyze the cluster properties [CHWH20, MALK19, XGF16]. Interpolation is another technique used to explore the latent space. It can be used to traverse between two known positions in latent space [RMC15, Whi16, CXTJ19]. However, the interpretation of the latent features is still not clear. Various approaches [XPYS18, SKGD18] adopted trial and error processes to find possible interpretations or a classifier to return the relevance of the latent space features. Several works have developed interactive frameworks to analyze the cluster formed in the latent space of dimensionality reduction methods [FKM19, CMK20, MJE21]. In addition to that, our proposal can unveil the importance of the input features in the latent space with a relevance score without any external help.

3. Background

Our contribution leverages the widely used XAI method named SHAP (SHapley Additive exPlanations) [LL17]. SHAP is a game-theoretic approach, based on *Shapley values*, to explain the output of any machine learning model. The explanation is provided as a sum of relevance from the input feature values. Each value is considered as a contribution to a cooperative game whose payout is the final prediction. Accordingly to the input instance, Shapley values estimate the payout among the input features. The computation of the Shapley values is exponential with the number of features. Kernel SHAP implements an approximated estimation of the Shapley values by using a weighted linear regression [LL17]. The other challenge to compute Shapley values is the correct estimation of suppressing one of the features. Since many machine learning

models are based on fixed-length input, suppressing a single feature is impossible. To overcome this problem, Kernel SHAP substitutes the value of a feature with the mean value observed over the whole dataset. If all the features are replaced with their corresponding means, it is possible to identify a mean input instance with the corresponding mean prediction value. This value is also called the *expected value*.

4. Connecting Data and Latent space

The scores returned by SHAP give us the ability to understand which real features have influenced the most the position in the latent space. Therefore, we can use these scores to explain the role played by the real features w.r.t the machine learning model. We organize this section in three parts: the first focuses on describing the latent space structure, the second focuses on how the explanation is created, and the third describes how to utilize the interactive framework to conduct this kind of analysis.

Let $\mathcal{D}\langle X,Y\rangle$ be a dataset where $X = \{x_1,\dots,x_n\}$ represents the features space, while $Y = \{y_1,\dots,y_n\}$, represents the label space. Let B a black box model trained on $\mathcal{D}$ which return the prediction labels $\hat{Y} = \{\hat{y}_1,\dots,\hat{y}_n\}$. Without loss in generality and for simplifying the presentation, we may consider a binary classification task and use the probability returned by the classifier as labels. Given a record $x \in X$, represented as a vector $x_i \in \mathbb{R}^m$, the corresponding classification of the black box $\hat{y} = B(x)$ and an autoencoder A, we indicate with the notation $z = A(x,\hat{y})$ the encoding of the instance x with A. We adopted a Conditional Variational Autoencoder (CVAE), i.e., an autoencoder trained on the feature space of X labeled with the prediction yielded by the black box. Typically, the representation of $\mathcal{D}$ in the latent space has a clustering-based structure [CHWH20], i.e., similar points are grouped in close areas in the latent space. In addition, CVAEs exploit the information given by $\hat{y}$ to place instances into the space accordingly with their labels. Then, with our proposal, we can observe the neighbors in the latent space, identifying common feature characteristics among the instance in the same group and catch the correlation with the target variable $\hat{y}$. With our framework, it is possible to define and select different neighborhoods in the latent space and compare the distributions of the real features of the points in the two groups. The features with the most different distributions among the two selected sets are the ones that characterize the belonging to the groups. In particular, if we select two clusters with points with different labels, it is possible to identify the discrimination features. For example, we can select a cluster with many points with target label L and another one with a lot of $\neg L$ and notice that the points in the L cluster are characterized by the value of the feature "Male" set to *true*.

Given an instance $x \in X$ and an autoencoder A trained on $[x,\hat{y}]$, our goal is to find the explanation e unveiling why x has been encoded in the latent representation $z = A(x)$. The proposed explanation is built from the Kernel SHAP method to estimate the Shapley values of z. We can interpret these values in terms of comparison with the *expected prediction*. The expected prediction is the outcome of the classifier when all the features are assigned their respective mean value. Thus we can project this feature in the latent space at point $\hat{z}$ and use that location as a reference position. Given the Shapley value of the instance x, we can add the contribution of each value starting from the reference position $\hat{z}$. Thus, starting from the expected prediction $\hat{z}$, we can sum to it the contribution of every input feature in the dimension k and obtain the position in the latent space, i.e., the latent representation $z = \hat{z} + \sum_m \vec{\phi}(x_m)$.

An explanation for a classified instance is built around exploring the Shapley values contributions. First, the contributions of each feature are sorted by order of magnitude. The user can then change the value of one of the input features to observe how the position of z in the latent space changes. Those features with higher relevance will produce a broader impact on the latent space since their corresponding magnitude is larger. We leverage an exploration strategy based on a low dimensional latent space to allow an efficient and comprehensible visualization of the induced space, i.e., two or three dimensions. When the user explores a new instance, the corresponding latent position is determined, and the vectors representing the contributions of the feature are computed and mapped on the visual space as vector arrows (Figure 1). The variation of one of the attributes affects changing the magnitude of the vector arrows displayed on the visualization. We clarify this aspect by highlighting that, since SHAP values can be approximated with linear regression, they also have some linear properties concerning the expected value. For example, consider a feature like *age* ranging from 0 to 100 with expected value 40, and consider that the value 80 has a SHAP score of 0.6. If we substitute the value of the feature to 40, then the SHAP score would be 0 since the contribution of *age* equal to the expected value is null. However, if we substitute the value with 20, the SHAP score will be negative due to the linear propriety. Since the position in the latent space is given by the SHAP values, we can use them as a guide to increase or decrease feature values in the real space to move points to different positions in the latent space. SHAP vector directions are reported on the right of the sliders in Figure 1 gives the user the ability to see how the change of an input feature will result in a modification of the position in the latent space. This allows the user to explore the latent space guided by the direction of the SHAP scores and analyze the prediction in different space portions.

5. What-If Analysis - The Interactive Framework

Moving into the latent space guided by SHAP scores can provide insights to identify homogeneous groups of points, outliers, or close points with different predictions despite their similar latent characteristics. In particular, this last case can be helpful to identify misclassifications or malicious use of the black box, for example, through adversarial instances. Since it is interesting to analyze different latent positions by playing with the input features, we realized an interactive framework, acting as a front-end for our latent space explanation proposal, that allows the user to analyze the distribution of latent points and interact with their SHAP values.

The central component of our solutions is the visualization of the latent space. The two dimensional visualization of the latent space is built as a scatter plot, where latent instances are represented as points. The projection of the expected feature is highlighted with a triangular shape, and it is used as a reference point for positioning the other instances. The contribution of SHAP values is visualized as arrows oriented accordingly to the coefficients returned by the explanation method. The destination location of z is high-

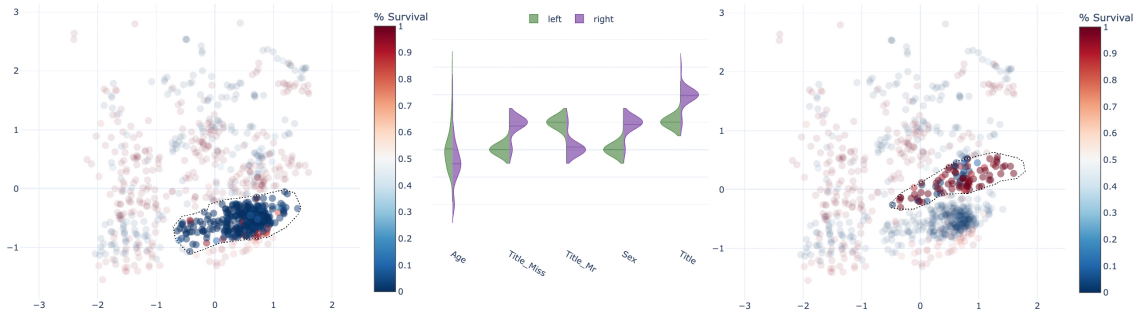

Figure 2: *Here, we have the distribution analysis of the two clusters analyzed in the Titanic dataset. The violet distribution is the distribution of the points belonging to the cluster highlighted in the right figure, while the dark green is the distribution of the left one. The features are sorted in descending order from the most separated distributions on the right to the least separated one to the left. The feature "Sex" and "Title" are the ones that characterize the clusters: all the points in the blue cluster are men with a low title.*

lighted with a cross shape. The latent space visualization is linked with a set of sliders where the user can specify values for each attribute. Each slider shows relevant information about the SHAP values for the corresponding feature. In particular, the slider highlight the minimum and maximum values and the position of the expected value for that attribute. Thus, the user can move the selection on the left or on the right of the expected value to change the magnitude of the corresponding vector arrow. We also provide directional vectors for each feature to help the user to predict the direction and verse of the translation in the latent space. When the user changes one of the values on a slider, a new instance is generated, the black box classifies it, and the SHAP values are computed. The SHAP values are then visualized on the latent space visualization. Since the number of features of the dataset may be large, we decided to visualize only the sliders of the top 10 most representative features selected in decreasing order of magnitude (from top to bottom) of the SHAP scores. The two displays are mutually linked: when a slider is changed, the latent space visualization is updated; when a point in the scatter plot is clicked, the corresponding feature values are loaded in the set of sliders.

The interface is divided into two parts. On the top, the sliders and the latent space visualization allow the exploration of latent space by changing the value of the input feature. On the bottom, the user can select groups of points from two instances of the latent space visualization (Figure 2). The two groups of points are then analyzed to determine which features distinguish one group from the other. The differences are visualized as a set of violin plots, comparing the distribution of observed values in the two groups. The violin plots are sorted accordingly with features that maximize the distance of the mean of distributions. We tested our framework with a public available tabular dataset, i.e.the *Titanic* dataset, which contains the data of real Titanic passengers. Each passenger is labeled as a *survivor* or *not-survivor*. Every categorical feature is transformed as a one-hot vector of dimension equal to the number of possible categorical values. A web application that implements our framework is accessible at the following link: `https://kdd.isti.cnr.it/LSE/`. The source code is also available on GitHub: the link is available from the web application URL. The continuous variables are normalized in the range [-1,1]. The dataset contains 1310 passengers described by attributes such as age, passenger class, sex, fee, etc. As a classification model, we trained an xgboost model [CG16]. The latent space created by the autoencoder is illustrated on the right of Figure 1. Most of the data labeled as *non-survivor* are concentrated in the lower right part of the latent space. The survived people, instead, are more sparse. Firstly, we selected two clusters in the interface to analyze the difference in distributions. We selected the one on the right with many not survived people and the one closer at the top. We can see that the feature "Title" is the most representative of the two clusters (Figure 2). In the blue cluster, the people have lower titles than those in the red. However, the people on the blue cluster are all men. We conclude that in the dataset, there is a correlation between "Sex" and "Title". Male people with a lower title are the ones with meager survival chances. In Figure 1 we took a point in the blue cluster on the bottom and tried to modify its most relevant features to move it to the top cluster. In particular, we can see that the feature "Title" has contributions in the direction of the cluster and has a lower value than the expected one. By modifying this feature to a value greater than the expected value (i.e., on the right of the e placeholder), we discovered that it is possible to move the point from the original cluster to the one in the center, which is mainly labeled as *survivor*. The more we increase the value, the more the point will be in the top part of the space. Interestingly the feature "Sex" has a low SHAP value meaning that our model is only looking at the feature "Title" for distinguishing points.

6. Discussion and Conclusions

This paper proposes an explanation methodology that exploits the combination of latent spaces learned by an autoencoder with a feature relevance method, namely SHAP. We have extracted beneficial information about the data under analysis by designing an interactive exploration of the latent space to evaluate the position of single instances or groups of points. We are working to introduce visual-based animation to bind the changes on the sliders with the updates on the visualization. We plan to organize a user-based survey for the evaluation of the interface, in particular for solving specific explanation tasks.

References

[BGG*21] BODRIA F., GIANNOTTI F., GUIDOTTI R., NARETTO F., PEDRESCHI D., RINZIVILLO S.: Benchmarking and survey of explanation methods for black box models. *arXiv preprint arXiv:2102.13076* (2021). 1

[BSA20] BAJAJ K., SINGH D. K., ANSARI M. A.: Autoencoders based deep learner for image denoising. *Procedia Computer Science 171* (2020), 1535–1541. 1

[CG16] CHEN T., GUESTRIN C.: Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (2016), pp. 785–794. 4

[CHWH20] CHEN M., HUANG L., WANG C.-D., HUANG D.: Multi-view clustering in latent embedding space. In *AAAI* (2020), pp. 3513–3520. 2, 3

[CMK20] CHATZIMPARMPAS A., MARTINS R. M., KERREN A.: t-visne: Interactive assessment and interpretation of t-sne projections. *IEEE transactions on visualization and computer graphics 26*, 8 (2020), 2696–2714. 2

[CXTJ19] CHEN Y.-C., XU X., TIAN Z., JIA J.: Homomorphic latent space interpolation for unpaired image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 2408–2416. 2

[FKM19] FUJIWARA T., KWON O.-H., MA K.-L.: Supporting analysis of dimensionality reduction results with contrastive learning. *IEEE transactions on visualization and computer graphics 26*, 1 (2019), 45–55. 2

[GBCB16] GOODFELLOW I., BENGIO Y., COURVILLE A., BENGIO Y.: *Deep learning*, vol. 1. MIT press Cambridge, 2016. 1

[GMR*18] GUIDOTTI R., MONREALE A., RUGGIERI S., TURINI F., GIANNOTTI F., PEDRESCHI D.: A survey of methods for explaining black box models. *ACM computing surveys (CSUR) 51*, 5 (2018), 1–42. 1

[Kra91] KRAMER M. A.: Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal 37*, 2 (1991), 233–243. 1

[KW13] KINGMA D. P., WELLING M.: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013). 2

[LL17] LUNDBERG S. M., LEE S.-I.: A unified approach to interpreting model predictions. In *Advances in neural information processing systems* (2017), pp. 4765–4774. 1, 2

[MALK19] MUKHERJEE S., ASNANI H., LIN E., KANNAN S.: Clustergan: Latent space clustering in generative adversarial networks. In *Proceedings of the AAAI conference on artificial intelligence* (2019), vol. 33, pp. 4610–4617. 2

[MJE21] MARCÍLIO-JR W. E., ELER D. M.: Explaining dimensionality reduction results using shapley values. *Expert Systems with Applications 178* (2021), 115020. 2

[N*11] NG A., ET AL.: Sparse autoencoder. *CS294A Lecture notes 72*, 2011 (2011), 1–19. 2

[RMC15] RADFORD A., METZ L., CHINTALA S.: Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015). 2

[SKGD18] SPINNER T., KÖRNER J., GÖRTLER J., DEUSSEN O.: Towards an interpretable latent space: an intuitive comparison of autoencoders with variational autoencoders. In *IEEE VIS 2018* (2018). 2

[SLY15] SOHN K., LEE H., YAN X.: Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems 28* (2015), 3483–3491. 2

[SRFB*20] SMITH-RENNER A., FAN R., BIRCHFIELD M., WU T., BOYD-GRABER J., WELD D. S., FINDLATER L.: No explainability without accountability: An empirical study of explanations and feedback in interactive ml. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–13. 2

[THZ*18] TRAN D. H., HUSSAIN Z., ZHANG W. E., KHOA N. L. D., TRAN N. H., SHENG Q. Z.: Deep autoencoder for recommender systems: parameter influence analysis. *arXiv preprint arXiv:1901.00415* (2018). 1

[TSK05] TAN P., STEINBACH M. S., KUMAR V.: *Introduction to Data Mining*. Addison-Wesley, 2005. 1

[Whi16] WHITE T.: Sampling generative networks. *arXiv preprint arXiv:1609.04468* (2016). 2

[XGF16] XIE J., GIRSHICK R., FARHADI A.: Unsupervised deep embedding for clustering analysis. In *International conference on machine learning* (2016), PMLR, pp. 478–487. 2

[XPYS18] XU K., PARK D. H., YI C., SUTTON C.: Interpreting deep classifier by visual distillation of dark knowledge. *arXiv preprint arXiv:1803.04042* (2018). 2

[YLK20] YOO J., LEE H., KWAK N.: Unpriortized autoencoder for image generation. In *2020 IEEE International Conference on Image Processing (ICIP)* (2020), IEEE, pp. 763–767. 1

[ZSE17] ZHAO S., SONG J., ERMON S.: Infovae: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262* (2017). 2