Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 53-2 (2020) 7903–7908

β-Variational Classifiers Under Attack

Marco Maggipinto [*] Matteo Terzi [*] Gian Antonio Susto [**]

[*] Department of Information Engineering (DEI), University of Padova, Italy (e-mail: marco.maggipinto@phd.unipd.it, terzimat@dei.unipd.it))
[**] DEI and Human-Inspired Technology Center, University of Padova, Italy (e-mail: gianantonio.susto@dei.unipd.it)

Abstract: Deep Neural networks have gained lots of attention in recent years thanks to the breakthroughs obtained in the field of Computer Vision. However, despite their popularity, it has been shown that they provide limited robustness in their predictions. In particular, it is possible to synthesise small adversarial perturbations that imperceptibly modify a correctly classified input data, making the network confidently misclassify it. This has led to a plethora of different methods to try to improve robustness or detect the presence of these perturbations. In this paper, we perform an analysis of β-Variational Classifiers, a particular class of methods that not only solve a specific classification task, but also provide a generative component that is able to generate new samples from the input distribution. More in details, we study their robustness and detection capabilities, together with some novel insights on the generative part of the model.

Keywords: Adversarial Training, Computer Vision, Deep Learning, Machine Learning, Robustness

1. INTRODUCTION

The astounding performance that Deep Neural Networks (DNNs) provide when dealing with large amounts of complex data has recently led to extensive research in Deep Learning (DL) technologies. Empirical evidence shows that, in contrast to standard Machine Learning (ML) methods, DNNs are able to generalize well in the over-parametrized regime (Belkin et al. (2018a,b)), i.e. when the number of parameters of the model is much higher than the number of data used to train it; hence, there is basically no limit, other than the computational capabilities, to the complexity of the hypothesis class of functions that is of practical use. Despite this property, that is still not well understood and object of an entire line of research, the high complexity of the input-output relationship comes at a cost: the predictions provided by DNNs are not interpretable, making it difficult to understand what caused the model to take a particular decision; moreover, (Szegedy et al. (2013)) discovered that DNNs are susceptible to adversarial perturbations, small changes in the input space that result in high changes in the output space. This allows the creation of *Adversarial Examples* (Goodfellow et al. (2014)): for example, in image classification, it is possible to synthesise artificial images that, while visually identical for the human eye to a correctly classified sample, they are confidently misclassified. While such problem is common in ML, it is emphasized in DNNs by the high dimensionality of the input space and the complexity of the function described by the DNN that may be subject to high curvature directions that can be exploited, even in a small neighborhood of a point, to significantly change the response of the network.

The discovery started a completely new research trend that tries to understand the phenomenon (see Liu et al. (2016); Shaham et al. (2018)) or find ways to defend against it. The most common approach to train robust networks is *Adversarial Training* (Goodfellow et al. (2014); Madry et al. (2017); Terzi et al. (2020)) that consists in generating adversarial examples and feeding them to the network during training along with the correct label. While effective, such method comes at the cost of reduced prediction accuracy (Tsipras et al. (2018)) and increased training time compared to standard models. Other approaches have been proposed to obtain robust models such as gradient regularization (Ross and Doshi-Velez (2018)), Lipschitz regularization (Finlay et al. (2018)) and curvature regularization (Moosavi-Dezfooli et al. (2019)). A different research line focuses on developing methods to detect adversarial examples, without requiring the model to be robust. In (Feinman et al. (2017)) it is proposed to combine Kernel Density Estimation on the hidden layer of the network (Botev et al. (2010)) and MC-Dropout (Gal and Ghahramani (2016)) that provides an estimate of the prediction uncertainty of the network. The rationale behind the method is that adversarial examples should have lower likelihood according to the estimated density and higher prediction uncertainty. (Gong et al. (2017)) propose to train a binary classifier as a detection method: it is shown that such approach is able to detect 99% of adversarial examples and it is robust to a second attack that aims at fooling the detection method. (Grosse et al. (2017)) uses the kernel-based two-sample test (Gretton et al. (2012)) to detect statistical differences between adversarial examples and normal data.

[⋆] Part of this work was supported by MIUR (Italian Minister for Education) under the initiative "Departments of Excellence" (Law 232/2016). We would like to also thank Nvidia for donating the Nvidia Titan V GPU used in for this research.

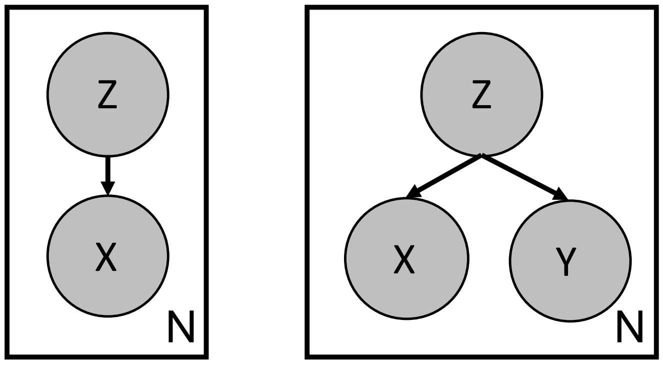


Fig. 1. Bayesian network of a VAE (left) and the Variational Classifier analyzed in this work (right).

Recently, (Li et al. (2018)) proposed a study on the robustness of Generative Classifiers and their detection capabilities. They propose three detection methods whose rejection policies are respectively: 1) reject samples with likelihood of the input lower than a certain threshold; 2) Reject samples with joint input/output likelihood lower than a certain threshold; 3) Reject over/under confident predictions. The methods have proven effective on object recognition tasks. Moreover, they show that models with lower capacity are more robust to adversarial examples.

We build upon (Li et al. (2018)) to provide an analysis of β-Variational Classifiers, a similar approach but based on β-Variational Autoencoders (Higgins et al. (2017)) that are able to provide a disentangled representation of the input (Burgess et al. (2018)) and give an alternative method to control the model capacity. In particular, the contributions of our paper are as follows:

- We analyze the robustness of β-Variational Classifiers combined with sparse regularization;
- We analyze the detection capabilities of β-Variational Classifiers;
- We analyse the effects of adversarial perturbations on the decoder network.

The remainder of this paper is organized as follows: In section 2 and 3 we provide a description of β-Variational Autoencoders and their β-Variational Classifiers. In Section 4 we explain the phenomenon of adversarial examples and the main method used to synthesise them. In Section 5 and 6 we describe the experimental settings and outline the obtained results. Finally, in Section 7 conclusions and future works are reported.

2. β-VARIATIONAL-AUTOENCODERS

Generative modeling aims at learning a parametrized model of the probability distribution underlying the data in order to obtain new realistic samples from it. In this context, Variational Autoencoders (VAEs) (Kingma and Welling (2013)) are a well know approach to develop a complex latent variable model that can be learned by Stochastic Gradient Descent (SDG).

Given a dataset of independent identically distributed samples $\{\boldsymbol{x}_i\}$ $i = 1, \cdots, N$ with distribution $p(\boldsymbol{x})$, we introduce hidden variables $\{\boldsymbol{z}_i\}$ $i = 1, \cdots, N$ of dimension d distributed as a multivariate Gaussian $p(\boldsymbol{z}) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ with $\boldsymbol{I} \in \mathcal{R}^{d \times d}$ the identity matrix.

VAEs model the joint distribution of the random variables $\boldsymbol{X}$ and $\boldsymbol{Z}$ as $p_\theta(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$ corresponding to the bayesian network in Figure 1(left) where the mean of $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is the output of a Decoder Neural Network $D_{\boldsymbol{\theta}}(\boldsymbol{z})$ parametrized by parameters $\boldsymbol{\theta}$, typical choices are Gaussian $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) \sim \mathcal{N}(D_{\boldsymbol{\theta}}(\boldsymbol{z}), \boldsymbol{I})$ for continuous output or Bernoulli $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) \sim \mathcal{B}(D_{\boldsymbol{\theta}}(\boldsymbol{z}))$ for binary output. Being the $\boldsymbol{z}_i$ unknown, we aim at finding the parameters value that maximizes the marginal likelihood; however, this requires evaluating an intractable integral to compute $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{z}}\left[p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right]$, which is also difficult to approximate by means of Monte Carlo methods due to the dimensionality of the hidden factors and the amount of data that is often very high in Deep Learning settings. In a similar scenario approximate inference is typically very effective; more in details, introducing an approximate posterior distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{x}, \boldsymbol{\Sigma}(\boldsymbol{x}))$ where $\boldsymbol{\mu}(\boldsymbol{x}), \boldsymbol{\Sigma}(\boldsymbol{x})$ are output of an Encoder network. We define the Expectation Lower bound (ELBO) $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x})$ as:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}) = &- D_{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z})\right) + \\ &\mathbb{E}_{q_{\boldsymbol{\phi}}}(\boldsymbol{z}|\boldsymbol{x})\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right]\end{aligned} \tag{1}$$

Where D_{KL} is the Kullback–Leibler divergence. It is always true that (for a detailed proof see Bishop (2006)):

$$\log\left(p_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}) \tag{2}$$

(2) has important implications when the parametrized distributions are extremely complex, such as Neural Networks. We can in fact maximize the ELBO instead of the intractable marginal likelihood. In particular, if we analyze the expression in (1), the Decoder network is trained to minimize an expected reconstruction error, while the Encoder network distribution is pushed to be close to the prior, this acts as a regularizer, tuning the capacity of the Encoder. Such regularizer impacts the type of representations that the Encoder can learn, in particular (Belkin et al. (2018a)) showed that by controlling this term using the following modified ELBO:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}) = &\beta|D_{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x} \,||\, p(\boldsymbol{z})\right) - C| + \\ &\mathbb{E}_{q_{\boldsymbol{\phi}}}(\boldsymbol{z}|\boldsymbol{x})\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right]\end{aligned} \tag{3}$$

The model is able to produce disentangled representations that are related to different characteristics of the image, e.g. color, shape etc. Here β and C are hyperparameters, the first one is usually kept very high around 1000 while the second directly control the capacity and is linearly increased at training time from 0 to a predefined value (this procedure has been shown to provide better representations). This modified version, is called β-VAE.

During optimization, $\mathbb{E}_{q_{\boldsymbol{\phi}}}(\boldsymbol{z}|\boldsymbol{x})\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right]$ is computed using a Monte Carlo approximation $\mathbb{E}_{q_{\boldsymbol{\phi}}}(\boldsymbol{z}|\boldsymbol{x})\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right] \approx \frac{1}{M}\sum_{m=1}^{M} log\, p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}^{(m)})$ $\{\boldsymbol{z}_m\}_{m=1}^{M} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$. Since it is not possible to back-propagate through the sampling operation, a reparametrization trick is used i.e. $\boldsymbol{z}_m = \mu(\boldsymbol{x}) + \xi_m \boldsymbol{\Sigma}(\boldsymbol{x})$ with $\xi_m \sim \mathcal{N}(0, 1)$. The detailed optimization procedure is reported in Algorithm 1.

3. β-VARIATIONAL-CLASSIFIERS

β-VAEs provide an interesting method to perform variational inference in the presence of complex parametrized models of probability distributions with hidden factors. A

Algorithm 1 Training Procedure

```
Input: ${\boldsymbol{x}^{(i)}\}_{i=1}^{n}, B, M, \beta, C, n_{iter}$
1: for $j = 1 \dots n_{iter}$ do
2:     Sample $B$ examples from the training set $\{\boldsymbol{x}^{(i)}\}_{i=1}^{B}$
3:     Sample $B \cdot M$ latent variables $\boldsymbol{z}_m^{(i)}$
4:     $c = \text{linearSchedule}(C, j)$
5:     Compute the gradient with respect to $\boldsymbol{\Phi}, \boldsymbol{\theta}$
       $$\boldsymbol{\delta_\theta} = \nabla_{\boldsymbol{\theta}} \frac{1}{B \cdot M} \sum_{i=1}^{B} \sum_{m=1}^{M} \log\, p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}|\boldsymbol{z}_m^{(i)})$$
       $$\boldsymbol{\delta_\Phi} = \nabla_{\boldsymbol{\Phi}} \frac{1}{B} \sum_{i=1}^{B} \Big[\beta\, |D_{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(i)})\,||\,p(\boldsymbol{z})\right) - c| + \frac{1}{M} \sum_{m=1}^{M} \log\, p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}|\boldsymbol{z}_m^{(i)})\Big]$$
6:     Ascend the gradient $\boldsymbol{\theta} = \text{ascendRule}(\boldsymbol{\theta}, \boldsymbol{\delta}_\theta)$
7:     Ascend the gradient $\boldsymbol{\Phi} = \text{ascendRule}(\boldsymbol{\theta}, \boldsymbol{\delta}_\Phi)$
8: end for
```

similar optimization procedure can be employed to learn a more complex Bayesian model that includes a random variable $\boldsymbol{Y}$ representing a class which the input belongs to. The resulting model is called β-Variational Classifier (β-VAC) . In this work, we focus our study on a particular β-VAC represented by the Bayesian network in Figure 1 (right), we assume that the dataset is composed by couples $\{\boldsymbol{x}^{(i)}, y^{(i)}\}_{i=1}^{n}$ where $y^{(i)}$ are the true labels (i.e. object classes) associated to the input. Introducing the conditional distribution $p_{\boldsymbol{\omega}}(y|\boldsymbol{z})$. The ELBO for such model is:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}, y) = -\,D_{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)\,||\,p(\boldsymbol{z})\right) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},y)}\left[\log\, p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p_{\boldsymbol{w}}(y|\boldsymbol{z})\right] \quad (4)$$

From now on, we assume that $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y) = q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ which states that all the information about $\boldsymbol{z}$ is contained in $\boldsymbol{x}$. The resulting ELBO, including the modified regularizer of Section 2 can be expressed as follows:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}, y) = \beta |D_{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\,||\,p(\boldsymbol{z})\right) - C| + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log\, p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p_{\boldsymbol{w}}(y|\boldsymbol{z})\right] \quad (5)$$

The optimization procedure to learn the model parameters is analogous to Algorithm 1.

4. ADVERSARIAL EXAMPLES

In object recognition, an adversarial example is an image that, while being visually indistinguishable or very similar to a correctly classified input, is confidently mis-classified by the model. The most common approach to synthetize an adversarial example $\boldsymbol{x}_{adv}$ is the Projected Gradient Descent (PGD) attack, where a normal data $\boldsymbol{x}$ is perturbed by following an ascending direction of the loss function while remaining in an ϵ-ball $B_\epsilon(\boldsymbol{x}) = \{\boldsymbol{x}_{adv}\, s.t.\, ||\boldsymbol{x} - \boldsymbol{x}_{adv}||_p \le \epsilon\}$ centered at the original sample. More in details, let $L(\boldsymbol{x}, y)$ be the value of the loss at $\boldsymbol{x}$ where y is the correct label, $\epsilon > 0$ the maximum distance from the original input relative to the maximum input value (e.g. 255 for 8bit images), k the number of iterations and α the step size, PGD works as desctibed in Algorithm 2.
$Proj(\boldsymbol{x}_j, B_\epsilon(\boldsymbol{x}))$ is the projection operator that varies depending on the chosen norm, typical choices are ℓ_2 and ℓ_∞. The value of ϵ identifies the strength of the applied perturbation. Typically, for simple tasks such as handwritten digit recognition, an high value is necessary to fool the network while for more complex tasks, the value can be much smaller. Figure 2 shows an adversarial example on a digit recognition task dataset, with $\epsilon = 0.1$.

Algorithm 2 PGD

```
Input: $\boldsymbol{x}, y, k, \epsilon, \alpha$
1: $\boldsymbol{x}_0 = x$
2: for $j = 1 \dots k$ do
3:     $\boldsymbol{x}_j = \boldsymbol{x}_j + \alpha \nabla L(\boldsymbol{x}_j, y)$
4:     $\boldsymbol{x}_j = Proj(\boldsymbol{x}_j, B_\epsilon(\boldsymbol{x}))$
5: end for
6: $\boldsymbol{x}_{adv} = \boldsymbol{x}_j$
7: return $\boldsymbol{x}_{adv}$
```

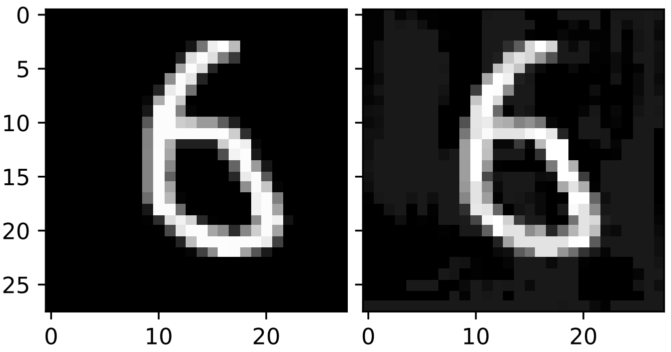


Fig. 2. Adversarial example on a digit recognition task: the 6 on the left is correctly classified while the one on the right is classified a 4.

5. EXPERIMENTAL SETTINGS

In order to analyze the robustness of β-VAC to adversarial examples we train the model on two popular datasets in Computer Vision for handwritten digit classification [1] (MNIST) and for clothes recognition [2] (Fashion-MNIST or FMNIST by Zalando research). They both are composed by 60 thousand labeled images for the train set and 10 thousand for testing, the images are in grayscale of size 28×28.

The structure of the model is as follows:

- Encoder: is an adapted Allcnn (Springenberg et al. (2014)) to provide an hidden representation of size 100 i.e. $\boldsymbol{z} \in \mathbb{R}^{100}$;
- Decoder: has a structure symmetric to the encoder, in order to provide as output an image of the same dimension of the input;
- Classifier: is a 2 layer perceptron with 64 hidden neurons per layer and ReLu (Nair and Hinton (2010)) activations.

We train for 60 epochs using an SGD optimizer with momentum 0.9 and learning rate 0.01 that is decreased at the 10th and 30th epoch by a factor of 10. The strength of the momentum is a common choice in literature while the other attributes have been chosen in order to properly

[1] http://yann.lecun.com/exdb/mnist
[2] https://github.com/zalandoresearch/fashion-mnist

train the network. We employ a small weight decay of 1e-6 (typical values are around 1e-3 for the Allcnn) so its influence on the capacity is limited and we are free to control it using the KL regularization term in the ELBO. We use PGD to compute the adversarial examples with variable strength, in particular for MNIST we use $\epsilon \in [0, 0.3]$ while for FMNIST $\epsilon \in [0, 0.1]$; the choice is justified by the more difficult classification task of the second dataset so a smaller perturbation is sufficient to effectively attack the network. We keep the number of iterations equal to 40 and the step size 0.01, the value of ϵ is referred to the ℓ_∞ norm.

6. RESULTS

In this section we outline the obtained results and, in particular, we analyze the effect of the capacity and sparse regularization on the robustness and detection capabilities of the β-VAC described in the previous section. To conclude, we visually inspect the reconstructed images of adversarial examples, showing some interesting properties.

6.1 Adversarial detection

The decoder part of the model can be extremely useful to detect adversarial examples: by definition, they provide an high variation of the network output given small variations of the input; hence, if the reconstructed image is strongly affected by the adversarial perturbation, it will probably be very different to the input fed to the network. An example of this phenomenon is shown in Figure 3(a) and 4(a), it is noticeable how the the reconstruction error considerably increases at the strengthening of the attack. We can thus train a classifier on the reconstruction error to obtain an effective attack detection methods. In the following, we will use a logistic classifier trained on adversarial examples computed on the training set and we analyse its performance by computing adversarial examples on the test set, and classifying both clean input and perturbed ones. We use the classification rate as performance metric being the dataset well balanced due to the high effectiveness of adversarial attacks.

6.2 Capacity effect

In Figure 3(b) and 4(b) we report the robustness of the β-VAC, measured in terms of accuracy at classifying adversarial examples, as a function of the parameter C controlling the capacity of the encoder network. We don't notice particular correlation between the capacity and the accuracy. On the FMNIST dataset, for small attack strength, there are some values of C that provide better robustness i.e. 0.01 and 1.0, however the first one has low accuracy on clean samples ($\epsilon = 0$) due to the limited capacity of the model. In general, we cannot conclude to be able to control the robustness by changing the capacity.

For the detection rate Figure 3(c) and 4(c), the capacity seems to have a more strong effect, in particular when values are too low, such as 0.01, the reconstruction error is high also for normal samples, making it difficult to distinguish them from the adversarial ones. On the other hand, we don't see a positive correlation between capacity and detection rate, so the parameter has to be fine tuned to get the best result. Overall, the detection accuracy is very good on the MNIST dataset also due to the higher attack strength used, for FMNIST the detection rate is well above 70% with the right capacity for every attack strength.

6.3 Sparse regularization effect

The idea of adding sparse regularization is motivated by the fact that the encoder network of β-VAE has been shown to provide disentangled representations. Hence, if the classifier selects only the ones that are useful for the classification task and discard the others, it should be more robust to adversarial perturbations. In Table 1 and 2 we show the effect of the ℓ_1 regularization on the robustness of the classifier trained using the best performing capacity, i.e 1.0 for MNIST and 10.0 for FMNIST. The ℓ_1 regularization does not seem to provide improved robustness. This result provides us the following insight: while it is easy to see that for linear models, obtaining l_∞-robustness is equivalent to applying ℓ_1-regularization, for non-linear model this relation is not true. Thus, our results give the evidence that the Encoder network is not providing robust latent embeddings.

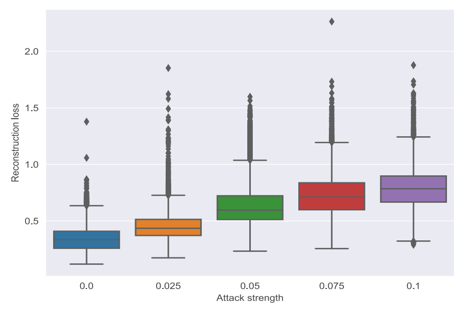
Overall, the detection accuracy is lower than the one for the non regularized method. This would be a fair price to pay in case of improved robustness but from the results obtained it is probably better to avoid using ℓ_1 regularization and choose the correct model complexity with C.

6.4 Decoded images inspection

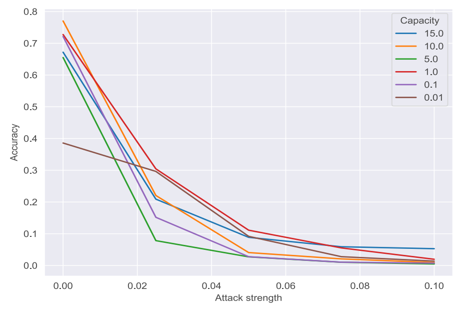
We analyse here the effect of an adversarial perturbation on the decoded images of the autoencoder. Interestingly, from Figure 5 we notice that the decoder is fooled to reconstruct an image that seems to belong to the same incorrect class provided the classifier. For example in the first row the true class is 0, the mistaken class is 6 and the decoder produces an image similar to a 6. This suggests that a similar model may be used for other vision tasks such as conditional image generation and style transfer. We reserve this analysis as a future work.

7. CONCLUSIONS

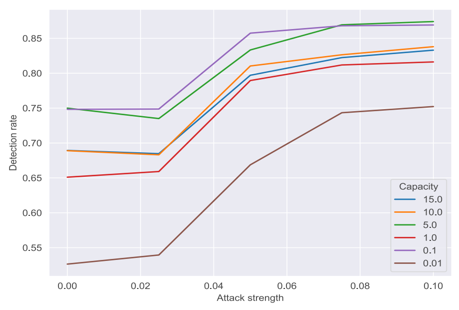
In this paper we proposed an analysis of β-VAC in the presence of adversarial perturbations. We have shown that the model does not provide increased robustness to adversarial examples however it is able to detect them effectively thanks to the reconstruction error of the decoder network. Sparse regularization of the classifier does not help in reducing the effects of adversarial perturbations on the classification. We have shown that the decoder, when fed with an adversarial example, tend to reconstruct an image that belongs to the class mistakenly selected by the classifier. As a future work, we want to investigate deeper this aspect that may be exploited to perform conditional image generation and style transfer tasks. We also plan to extend these results to more complex vision datasets.



(a) Reconstruction error

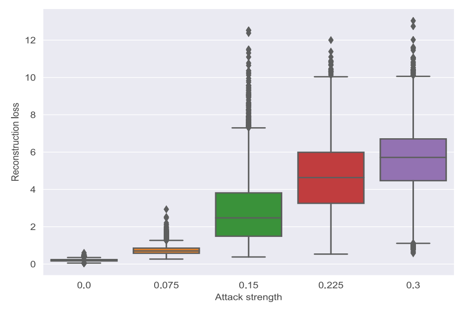


(b) Model accuracy

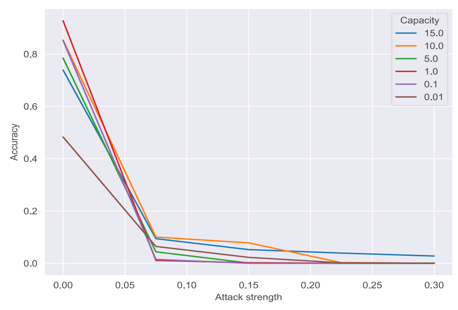


(c) Detection rate

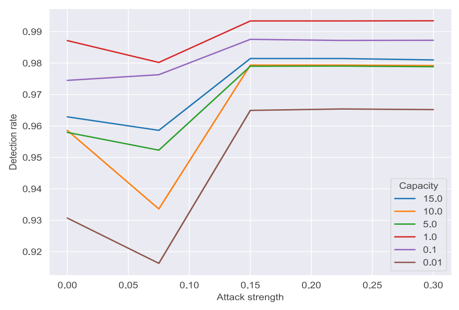
Fig. 3. Results for the FMNIST dataset.



(a) Reconstruction error



(b) Model accuracy



(c) Detection rate

Fig. 4. Results for the MNIST dataset.

Table 1. $l1$ regularization effect on the robustness for the MNIST dataset.

	$\epsilon = 0.0$	$\epsilon = 0.075$	$\epsilon = 0.150$	$\epsilon = 0.225$	$\epsilon = 0.3$
1e-06	0.9395	0.1360	0.0988	0.0918	0.0471
5-07	0.8461	0.0111	0.0004	0.0002	0.0001
1-07	0.8452	0.0498	0.0218	0.0215	0.0254
5e-08	0.8320	0.1194	0.0646	0.0527	0.0377

Table 2. $l1$ regularization effect on the robustness for the FMNIST dataset.

	$\epsilon = 0.0$	$\epsilon = 0.025$	$\epsilon = 0.050$	$\epsilon = 0.075$	$\epsilon = 0.1$
1e-03	0.624	0.185	0.083	0.027	0.009
1e-04	0.657	0.081	0.041	0.026	0.020
1e-05	0.722	0.260	0.140	0.089	0.063
1e-06	0.764	0.216	0.084	0.048	0.033

Table 3. ℓ_1 regularization effect on the detection accuracy for the MNIST dataset.

ℓ_1 strength	$\epsilon = 0.0$	$\epsilon = 0.075$	$\epsilon = 0.150$	$\epsilon = 0.225$	$\epsilon = 0.3$
1e-06	0.976	0.976	0.988	0.988	0.988
5e-07	0.967	0.976	0.983	0.983	0.983
1e-07	0.968	0.969	0.984	0.984	0.984
5e-08	0.945	0.953	0.972	0.972	0.972

REFERENCES

Belkin, M., Ma, S., and Mandal, S. (2018a). To understand deep learning we need to understand kernel learning. *arXiv:1802.01396*.

Belkin, M., Rakhlin, A., and Tsybakov, A.B. (2018b). Does data interpolation contradict statistical optimality? *arXiv:1806.09471*.

Bishop, C.M. (2006). *Pattern recognition and machine learning*. springer.

Botev, Z.I., Grotowski, J.F., Kroese, D.P., et al. (2010). Kernel density estimation via diffusion. *The annals of Statistics*, 38(5), 2916–2957.

Burgess, C.P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., and Lerchner, A. (2018). Understanding disentangling in β-vae. *arXiv:1804.03599*.

Feinman, R., Curtin, R.R., Shintre, S., and Gardner, A.B. (2017). Detecting adversarial samples from artifacts. *arXiv:1703.00410*.

Table 4. ℓ_1 regularization effect on the detection accuracy for the FMNIST dataset.

ℓ_1 strength	$\epsilon = 0.0$	$\epsilon = 0.025$	$\epsilon = 0.050$	$\epsilon = 0.075$	$\epsilon = 0.1$
1e-03	0.628	0.637	0.741	0.782	0.800
1e-04	0.738	0.716	0.812	0.843	0.852
1e-05	0.699	0.686	0.810	0.838	0.841
1e-06	0.713	0.720	0.830	0.846	0.850

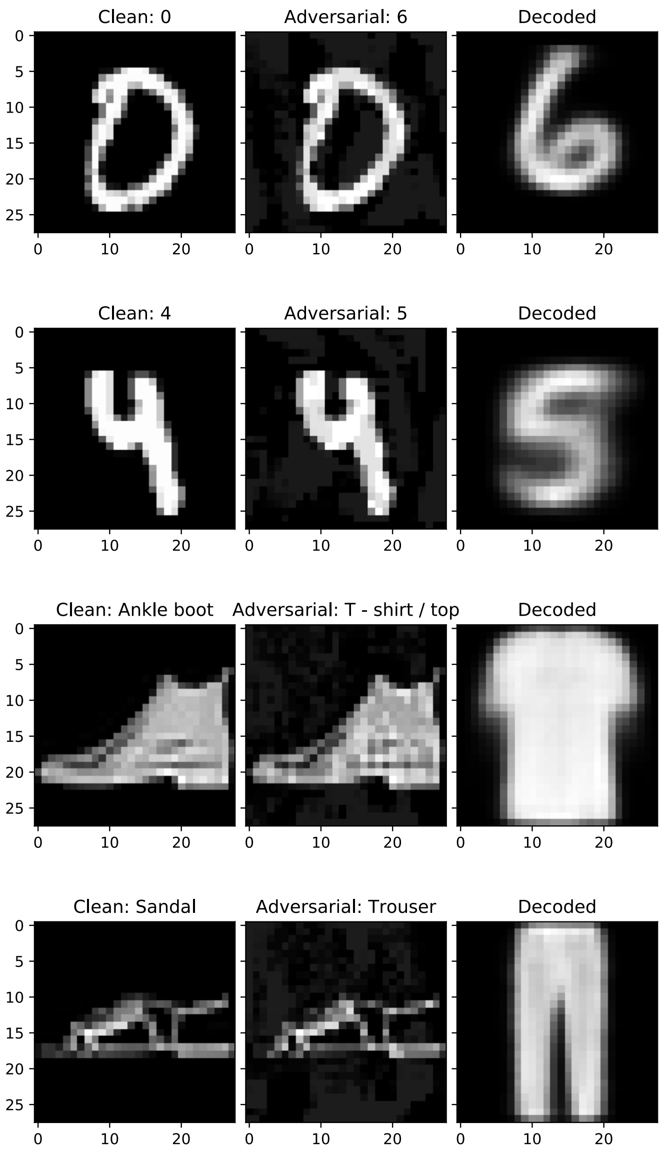


Fig. 5. Example of the effect of adversarial perturbations on the decoder network. It is noticeable how the reconstruction belongs to the class mistakenly chosen by the classifier.

Finlay, C., Calder, J., Abbasi, B., and Oberman, A. (2018). Lipschitz regularized deep neural networks generalize and are adversarially robust. *arXiv:1808.09540*.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICLR*, 1050–1059.

Gong, Z., Wang, W., and Ku, W.S. (2017). Adversarial and clean data are not twins. *arXiv:1704.04960*.

Goodfellow, I.J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv:1412.6572*.

Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar), 723–773.

Grosse, K., Manoharan, P., Papernot, N., Backes, M., and McDaniel, P. (2017). On the (statistical) detection of adversarial examples. *arXiv:1702.06280*.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2017). beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2(5), 6.

Kingma, D.P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv:1312.6114*.

Li, Y., Bradshaw, J., and Sharma, Y. (2018). Are generative classifiers more robust to adversarial attacks? *arXiv:1802.06552*.

Liu, Y., Chen, X., Liu, C., and Song, D. (2016). Delving into transferable adversarial examples and black-box attacks. *arXiv:1611.02770*.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv:1706.06083*.

Moosavi-Dezfooli, S.M., Fawzi, A., Uesato, J., and Frossard, P. (2019). Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9078–9086.

Nair, V. and Hinton, G.E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, 807–814.

Ross, A.S. and Doshi-Velez, F. (2018). Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Thirty-second AAAI conf. on artificial intelligence.*

Shaham, U., Yamada, Y., and Negahban, S. (2018). Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307, 195–204.

Springenberg, J.T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv:1412.6806*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv:1312.6199*.

Terzi, M., Susto, G.A., and Chaudhari, P. (2020). Directional adversarial training for cost sensitive deep learning classification applications. *Engineering Applications of Artificial Intelligence*, 91, 103550.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. (2018). Robustness may be at odds with accuracy. *arXiv:1805.12152*.