DEEP VARIATIONAL INFORMATION BOTTLENECK

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, Kevin Murphy
Google Research
{alemi,iansf,jvdillon,kpmurphy}@google.com

ABSTRACT

We present a variational approximation to the information bottleneck of Tishby et al. (1999). This variational approach allows us to parameterize the model using a neural network and leverage the reparameterization trick for efficient training. We call this method "Deep Variational Information Bottleneck", or Deep VIB. We show that models trained with the VIB objective outperform those that are trained with other forms of regularization, in terms of generalization performance and robustness to adversarial attack.

1 INTRODUCTION

We adopt an information theoretic view of deep networks. We regard the internal representation of some intermediate layer as a stochastic encoding Z of the input source X, defined by a parametric encoder $p(\mathbf{z}|\mathbf{x};\boldsymbol{\theta})$. Our goal is to learn an encoding that is maximally informative about our target Y, measured by the mutual information between our encoding and the target $I(Z,Y;\boldsymbol{\theta})$, where

$$I(Z,Y;\boldsymbol{\theta}) = \int dx\; dy\; p(z,y|\boldsymbol{\theta}) \log \frac{p(z,y|\boldsymbol{\theta})}{p(z|\boldsymbol{\theta})p(y|\boldsymbol{\theta})}.^{1} \tag{1}$$

Given the data processing inequality, and the invariance of the mutual information to reparameterizations, if this was our only objective we could always ensure a maximally informative representation by taking the identity encoding of our data ($Z = X$), but this is not a useful representation of our data. Instead we would like to find the best representation we can obtain subject to a constraint on its complexity. A natural and useful constraint to apply is on the mutual information between our encoding and the original data, $I(X,Z) \leq I_c$, where I_c is the information constraint. This suggests the objective:

$$\max_{\boldsymbol{\theta}} I(Z,Y;\boldsymbol{\theta}) \text{ s.t. } I(X,Z;\boldsymbol{\theta}) \leq I_c\,. \tag{2}$$

Equivalently, with the introduction of a Lagrange multiplier β, we can maximize the objective function

$$R_{IB}(\theta) = I(Z,Y;\boldsymbol{\theta}) - \beta I(Z,X;\boldsymbol{\theta}). \tag{3}$$

Here our goal is to learn an encoding Z that is maximally expressive about Y while being maximally compressive about X, where $\beta \geq 0$ controls the tradeoff.[2] This approach is known as the information bottleneck (IB), and was first proposed in Tishby et al. (1999). Intuitively, the first term in R_{IB} encourages Z to be predictive of Y; the second term encourages Z to "forget" X. Essentially it forces Z to act like a minimal sufficient statistic of X for predicting Y.

The IB principle is appealing, since it defines what we mean by a good representation, in terms of the fundamental tradeoff between having a concise representation and one with good predictive power (Tishby & Zaslavsky, 2015a). The main drawback of the IB principle is that computing mutual information is, in general, computationally challenging. There are two notable exceptions: the first is when X, Y and Z are all discrete, as in Tishby et al. (1999); this can be used to cluster discrete

[1] Note that in the present discussion, Y is the ground truth label which is independent of our parameters so $p(y|\boldsymbol{\theta}) = p(y)$.

[2] Note that, in our notation, large β results in a highly compressed representation. In some works, the IB principle is formulated as the minimization of $I(Z,X) - \beta I(Z,Y)$, in which case large β corresponds to high mutual information between Z and Y, and hence low compression.

data, such as words. The second case is when X, Y and Z are all jointly Gaussian (Chechik et al., 2005). However, this is a very limiting assumption.

In this paper, we propose to use variational inference to construct a lower bound on the IB objective in Equation 3. We call the resulting method VIB (variational information bottleneck). By using the reparameterization trick (Kingma & Welling, 2014), we can use Monte Carlo sampling to get an unbiased estimate of the gradient, and hence we can optimize the objective using stochastic gradient descent. This allows us to use deep neural networks to parameterize our distributions, and thus to handle high dimensional, continuous data, such as images, avoiding the previous restrictions to the discrete or Gaussian cases.

We also show, by a series of experiments, that stochastic neural networks, fit using our VIB method, are robust to overfitting, since VIB finds a representation Z which ignores as many details of the input X as possible. In addition, they are more robust to adversarial inputs than deterministic models which are fit using (penalized) maximum likelihood estimation. Intuitively this is because each input image gets mapped to a distribution rather than a unique Z, so it is more difficult to pass small, idiosyncratic perturbations through the latent bottleneck.

2 Related work

The idea of using information theoretic objectives for deep neural networks was pointed out in Tishby & Zaslavsky (2015b). However, they did not include any experimental results, since their approach for optimizing the IB objective relied on the iterative Blahut Arimoto algorithm, which is infeasible to apply to deep neural networks.

Variational inference is a natural way to approximate the problem. Variational bounds on mutual information have previously been explored in Agakov (2004), though not in conjunction with the information bottleneck objective. Mohamed & Rezende (2015) also explore variational bounds on mutual information, and apply them to deep neural networks, but in the context of reinforcement learning. We very recently discovered Chalk et al. (2016), who independently developed the same variational lower bound on the IB objective as us. However, they apply it to sparse coding problems, and use the kernel trick to achieve nonlinear mappings, whereas we apply it to deep neural networks, which are computationally more efficient. In addition, we are able to handle large datasets by using stochastic gradient descent, whereas they use batch variational EM.

In the supervised learning literature, the most closely related paper is the recently proposed confidence penalty (entropy regularization) method of (Pereyra et al., 2016). In this work, they fit a deterministic network by optimizing an objective that combines the usual cross entropy loss with an extra term which penalizes models for having low entropy predictive distributions. In more detail, their cost function has the form

$$J_{CP} = \frac{1}{N}\sum_{n=1}^{N}\left[H(p(y|y_n), p(y|x_n)) - \beta H(p(y|x_n))\right] \tag{4}$$

where $H(p,q) = -\sum_y p(y)\log q(y)$ is the cross entropy, $H(p) = H(p,p)$ is the entropy, $p(y|y_n) = \delta_{y_n}(y)$ is a one-hot encoding of the label y_n, and N is the number of training examples. (Note that setting $\beta = 0$ corresponds to the usual maximum likelihood estimate.) In (Pereyra et al., 2016) they show that CP performs better than the simpler technique of label smoothing, in which we replace the zeros in the one-hot encoding of the labels by $\epsilon > 0$, and then renormalize so that the distribution still sums to one. We will compare our VIB method to both confidence penalty and label smoothing in Section 4.1.

In the unsupervised learning literature, the most closely related paper is the one on variational autoencoders Kingma & Welling (2014). In fact, their method is a special case of an unsupervised version of the VIB, but with the β parameter fixed at 1.0, as we explain in Appendix A. (The VAE objective, but with different values of β, was also explored in Higgins et al. (2016), but from a different perspective.)

3 METHOD

Following standard practice in the IB literature, we assume that the joint distribution $p(X, Y, Z)$ factors as follows:

$$p(X, Y, Z) = p(Z|X, Y)p(Y|X)p(X) = p(Z|X)p(Y|X)p(X) \tag{5}$$

i.e., we assume $p(Z|X, Y) = p(Z|X)$, corresponding to the Markov chain $Y \leftrightarrow X \leftrightarrow Z$. This restriction means that our representation Z cannot depend directly on the labels Y. (This opens the door to unsupervised representation learning, which we will discuss in Appendix A.)

Recall that the IB objective has the form $I(Z, Y) - \beta I(Z, X)$. We will examine each of these expressions in turn. Let us start with $I(Z, Y)$. Writing it out in full, this becomes

$$I(Z, Y) = \int dy\, dz\, p(y, z) \log \frac{p(y, z)}{p(y)p(z)} = \int dy\, dz\, p(y, z) \log \frac{p(y|z)}{p(y)}\,. \tag{6}$$

where $p(y|z)$ is defined by our encoder and Markov Chain as follows:

$$p(y|z) = \int dx\, p(y, x|z) = \int dx\, p(y|x)p(x|z) = \int dx\, \frac{p(y|x)p(z|x)p(x)}{p(z)}. \tag{7}$$

Since this is intractable in our case, let $q(y|z)$ be a variational approximation to $p(y|z)$. This is our decoder, which we will take to be another neural network with its own set of parameters. Using the fact that the Kullback Leibler divergence is always positive, we have

$$\mathrm{KL}[p(Y|Z), q(Y|Z)] \geq 0 \implies \int dy\, p(y|z) \log p(y|z) \geq \int dy\, p(y|z) \log q(y|z)\,, \tag{8}$$

and hence

$$I(Z, Y) \geq \int dy\, dz\, p(y, z) \log \frac{q(z|y)}{p(y)} \tag{9}$$

$$= \int dy\, dz\, p(y, z) \log q(z|y) - \int dy\, p(y) \log p(y) \tag{10}$$

$$= \int dy\, dz\, p(y, z) \log q(z|y) + H(Y)\,. \tag{11}$$

Notice that the entropy of our labels $H(Y)$ is something outside of our control and so can be ignored for the purposes of optimization.

Focusing on the first term in Equation 11, we can rewrite $p(y, z)$ as $p(y, z) = \int dx\, p(x, y, z) = \int dx\, p(x)p(y|x)p(z|x)$ (leveraging our Markov assumption), which gives us a lower bound on the first term of our objective:

$$I(Z, Y) \geq \int dx\, dy\, dz\, p(x)p(y|x)p(z|x) \log q(y|z)\,. \tag{12}$$

Now we need to tackle the second term in our objective, $\beta I(Z, X)$. We have

$$I(Z, X) = \int dz\, dx\, p(x, z) \log \frac{p(z|x)}{p(z)} = \int dz\, dx\, p(x, z) \log p(z|x) - \int dz\, p(z) \log p(z)\,. \tag{13}$$

In general, computing the marginal distribution of our code, $p(z) = \int dx\, p(z|x)p(x)$, might be difficult. So let $r(z)$ be a variational approximation to this marginal. Since $\mathrm{KL}[p(Z), r(Z)] \geq 0 \implies \int dz\, p(z) \log p(z) \geq \int dz\, p(z) \log r(z)$, we have the following upper bound:

$$I(Z, X) \leq \int dx\, dz\, p(x)p(z|x) \log \frac{p(z|x)}{r(z)}\,. \tag{14}$$

Combining both of these bounds we have that

$$I(Z, Y) - \beta I(Z, X) \geq \int dx\, dy\, dz\, p(x)p(y|x)p(z|x) \log q(y|z)$$
$$- \beta \int dx\, dz\, p(x)p(z|x) \log \frac{p(z|x)}{r(z)} = L\,. \tag{15}$$

We now discuss how to compute the lower bound L in practice. We can approximate $p(x, y)$ using the empirical data distribution $p(x, y) = \frac{1}{N} \sum_{n=1}^{N} \delta_{x_n}(x)\delta_{y_n}(y)$, and hence we can write

$$L = \frac{1}{N} \sum_{n=1}^{N} \left[\int dz\, p(z|x_n) \log q(y_n|z) - \beta\, p(z|x_n) \log \frac{p(z|x_n)}{r(z)} \right]. \tag{16}$$

Suppose we use an encoder of the form $p(z|x) = \mathcal{N}(z|f_e^{\mu}(x), f_e^{\Sigma}(x))$, where f_e is an MLP which outputs both the K dimensional mean of z as well as the $K \times K$ covariance matrix. Then we can use the reparameterization trick (Kingma & Welling, 2014) to write $p(z|x)dz = p(\epsilon)d\epsilon$, where $z = f(x, \epsilon)$ is a deterministic function of x and the Gaussian random variable ϵ. This formulation has the important advantage that the noise term is independent of the parameters of the model, so it is easy to take gradients.

Putting this altogether, we get the following objective function, which we will try to minimize:

$$J_{IB} = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{\epsilon \sim p(\epsilon)} \left[-\log q(y_n|f(x_n, \epsilon)) \right] + \beta\, \mathrm{KL}\left[p(Z|x_n), r(Z) \right]. \tag{17}$$

Assuming our choice of $p(z|x)$ and $r(z)$ allows computation of an analytic Kullback-Leibler divergence, we can directly back propagate through a single sample of our stochastic code and ensure that our gradient is an unbiased estimate of the true expected gradient. Even if our choice of encoding distribution and variational prior do not admit an analytic KL, we could similarly reparameterize through a sample of the divergence as in Blundell et al. (2015).

4 Experimental results

In this section, we present various experimental results, comparing the behavior of standard deterministic networks to stochastic neural networks fit by optimizing the VIB objective. For simplicity, we restrict attention to the well-known MNIST dataset, which consists of 60,000 28x28 images of hand-drawn digits, from 10 classes.

In all the experiments, the encoder has the form $p(z|x) = \mathcal{N}(z|f_e^{\mu}(x), f_e^{\Sigma}(x))$. The f_e MLP has two hidden layers of size 1024, and uses ReLu activations. The decoder is just a logistic regression model of the form $q(y|z) = \mathcal{S}(y|f_d(z))$, where $f_d(z) = Wz + b$ returns the logits over the $C = 10$ classes, and $\mathcal{S}(a) = [\exp(a_c)/\sum_{c'=1}^{C} \exp(a_{c'})]$ is the softmax function. Finally, we treat $r(z)$ as a fixed K dimensional spherical Gaussian, $r(z) = \mathcal{N}(z|0, I)$.

In the special case that $\beta = 0$, we obtain the following objective function:

$$J_{IB0} = -\frac{1}{N} \sum_{n=1}^{N} E_{z \sim \mathcal{N}(f_e^{\mu}(x_n), f_e^{\Sigma}(x_n))} \left[\log \mathcal{S}(y_n|f_d(z)) \right] \tag{18}$$

When $\beta \to 0$, we observe the VIB optimization process tends to make $f_e^{\Sigma}(x) \to 0$, so the network becomes nearly deterministic. However, in our experiments we also explicitly fit a deterministic model that has the same form as the stochastic model, except that we just use $z = f_e^{\mu}(x)$ as the hidden encoding, and drop the Gaussian layer.

4.1 Behavior on MNIST

In this section, we compare deterministic and stochastic models on an unmodified version of the MNIST dataset.

4.1.1 Higher dimensional embedding

To demonstrate that our VIB method can achieve competitive classification results, we compared against a deterministic MLP trained with various forms of regularization. We use a $K = 256$ dimensional bottleneck and a diagonal Gaussian for $p(z|x)$. The networks were trained using Tensorflow for 200 epochs using the Adam optimizer (Kingma & Ba, 2015) with a learning rate of 0.001.

Model	error
Baseline	1.38%
Dropout	1.34%
Dropout (Pereyra et al., 2016)	1.40%
Confidence Penalty	1.36%
Confidence Penalty (Pereyra et al., 2016)	1.17%
Label Smoothing	1.40%
Label Smoothing (Pereyra et al., 2016)	1.23%
VIB ($\beta = 10^{-3}$)	**1.13%**

Table 1: Test set misclassification rate on MNIST using $K = 256$. We compare our method (VIB) to an equivalent deterministic model using various forms of regularization. The discrepancy between our results for confidence penalty and label smoothing and the numbers reported in (Pereyra et al., 2016) are due to slightly different hyperparameters.

The results are shown in Table 1. We see that we can slightly outperform other forms of regularization that have been proposed in the literature. Of course, the performance varies depending on β. Figure 1(a) plots the train and test error vs β, for the case where we use a single Monte Carlo sample of z when predicting, and also for the case where we average over 12 posterior samples (i.e., we use $p(y|x) = \frac{1}{S}\sum_{s=1}^{S} q(y|z^s)$ for $z^s \sim p(z|x)$, where $S = 12$).

We see several interesting properties in Figure 1(a). First, we notice that the error rate shoots up once β rises above the critical value of $\beta \sim 10^{-2}$. This corresponds to a setting where the mutual information between X and Z is less than $\log_2(10)$ bits, so the model can no longer represent the fact that there are 10 different classes. Second, we notice that, for small values of β, the test error is higher than the training error, which indicates that we are overfitting. This is because the network learns to be more deterministic, forcing $\sigma \approx 0$, thus reducing the benefits of regularization. Third, we notice that for intermediate values of β, Monte Carlo averaging helps.

In Figure 1(c), we plot the IB curve, i.e., we plot $I(Z, Y)$ vs $I(Z, X)$ as we vary β. As we allow more information from the input through to the bottleneck (by lowering β), we increase the mutual information between our embedding and the label on the training set, but not necessarily on the test set, as is evident from the plot.

In Figure 1(d) we plot the second term in our objective, the upper bound on the mutual information between the images X and our stochastic encoding Z, which in our case is simply the relative entropy between our encoding and the fixed isotropic unit Gaussian prior. Notice that the y-axis is a logarithmic one. This demonstrates that our best results (when β is between 10^{-3} and 10^{-2}) occur where the mutual information between the stochastic encoding and the images is on the order of 10 to 100 bits.

4.1.2 TWO DIMENSIONAL EMBEDDING

To better understand the behavior of our method, we refit our model to MNIST using a $K = 2$ dimensional bottleneck, but using a full covariance Gaussian. (The neural net predicts the mean and the Cholesky decomposition of the covariance matrix.) Figure 1(b) shows that, not surprisingly, the classification performance is worse, but the overall trends are the same as in the $K = 256$ dimensional case. The IB curve (not shown) also has a similar shape to before, except now the gap between training and testing is even larger.

Figure 2 provides a visualization of what the network is doing. We plot the posteriors $p(z|x)$ as a 2d Gaussian ellipse (representing the 95% confidence region) for 1000 images from the test set. Colors correspond to the true class labels. In the background of each plot is the entropy of the variational classifier $q(y|z)$ evaluated at that point.

We see several interesting properties. First, as β decreases (so we pass less information through), the posterior covariances become larger, and the classes start to overlap. Second, once β passes a critical value, the encoding "collapses", and essentially all the class information is lost. Third, there is a fair amount of posterior uncertainty in the predictive distribution $q(y|z)$ in the areas between the class embeddings. Fourth, for intermediate values of β (say 10^{-1} in Figure 2(b)), predictive

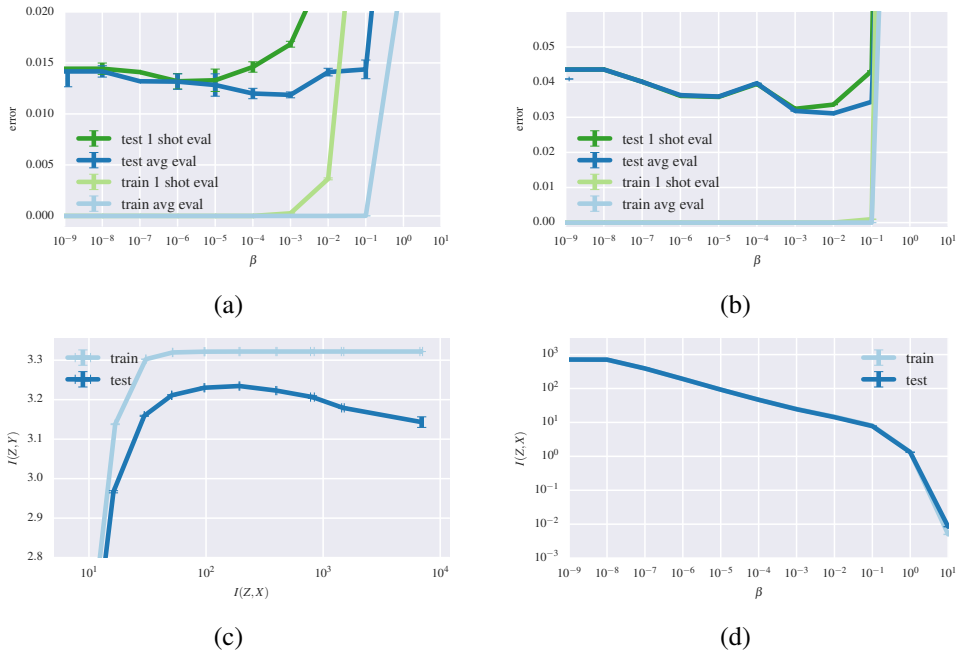
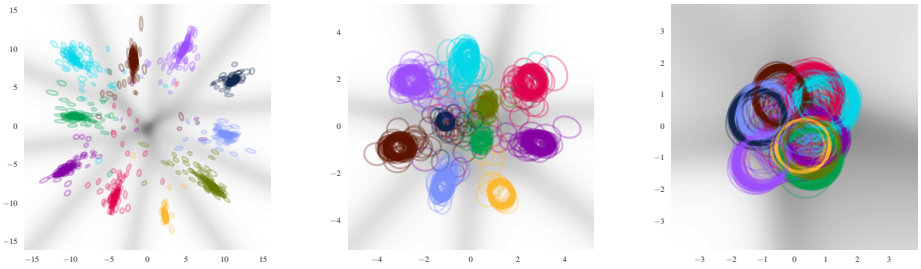


Figure 1: Results of VIB model on MNIST. (a) Error rate vs β for $K = 256$ on train and test set. "1 shot eval" means a single posterior sample of z, "avg eval" means 12 Monte Carlo samples. The spike in the error rate at $\beta \sim 10^{-2}$ corresponds to a model that is too highly regularized. (b) Same as (a), but for $K = 2$. Performance is much worse, since we pass through a very narrow bottleneck. (c) $I(Z, Y)$ vs $I(Z, X)$ as we vary β for $K = 256$. We see that increasing $I(Z, X)$ helps training set performance, but can result in overfitting. (d) $I(Z, X)$ vs β for $K = 256$. We see that for a good value of β, such as 10^{-2}, we only need to store about 10 bits of information about the input.



(a) $\beta = 10^{-3}$, $\text{err}_{\text{mc}} = 3.18\%$, $\text{err}_1 = 3.24\%$ (b) $\beta = 10^{-1}$, $\text{err}_{\text{mc}} = 3.44\%$, $\text{err}_1 = 4.32\%$ (c) $\beta = 10^{0}$, $\text{err}_{\text{mc}} = 33.82\%$, $\text{err}_1 = 62.81\%$.

Figure 2: Visualizing embeddings of 1000 test images in two dimensions. We plot the 95% confidence interval of the Gaussian posterior $p(z|x) = \mathcal{N}(\mu, \Sigma)$ as an ellipse. The images are colored according to their true class label. The background greyscale image denotes the entropy of the variational classifier evaluated at each two dimensional location. As β becomes smaller, and we forget more about the input, the embeddings start to overlap to such a degree that the classes become indistinguishable. We also report the test error using a single sample, err_1, and using 12 Monte Carlo samples, err_{mc}. For "good" values of β, a single sample suffices.

performance is still good, even though there is a lot of uncertainty about where any individual image will map to. This means it would be difficult for an outside agent to infer which particular instance the model is representing, a property which we will explore more in the following sections.

4.2 BEHAVIOR ON ADVERSARIAL IMAGES

Szegedy et al. (2013) was the first work to show that deep neural networks (and other kinds of classifiers) can be easily "fooled" into making mistakes by changing their inputs by imperceptibly small amounts. In this section, we will show how training with the VIB objective makes models significantly more robust to such adversarial inputs.

4.2.1 TYPES OF ADVERSARIES

Since the initial work by Szegedy et al. (2013) and Goodfellow et al. (2014), many different adversaries have been proposed. Most attacks fall into three broad categories: optimization-based attacks (Szegedy et al., 2013; Carlini & Wagner, 2016; Moosavi-Dezfooli et al., 2016; Papernot et al., 2015; Robinson & Graham, 2015; Sabour et al., 2016), which directly run an optimizer such as L-BFGS or ADAM (Kingma & Ba, 2015) on image pixels to find a minimal perturbation that changes the model's classification; single-step gradient-based attacks (Goodfellow et al., 2014; Kurakin et al., 2016; Huang et al., 2015), which choose a gradient direction of the image pixels at some loss, and then take a single step in that direction; and iterative gradient-based attacks (Kurakin et al., 2016), which take multiple small steps along the gradient direction of the image pixels at some loss, recomputing the gradient direction at each step.[3]

Many adversaries can be formalized as either untargeted or targeted variants. An untargeted adversary can be defined as $A(X, M) \rightarrow X'$, where $A(.)$ is the adversarial function, X is the input image, X' is the perturbed output, and M is the target model. A is considered successful if $M(X) \neq M(X')$. Recently, Moosavi-Dezfooli et al. (2016) showed how to create a "universal" adversarial perturbation δ that can be added to any image X in order to make $M(X + \delta) \neq M(X)$.

A targeted adversary can be defined as $A(X, M, l) \rightarrow X'$, where l is an additional target label, and A is only considered successful if $M(X') = l$.[4] Targeted attacks usually require larger magnitude perturbations, since the adversary cannot just "nudge" the input across the nearest decision boundary, but instead must force it into a desired decision region.

In this work, we focus on the L_2 attack method proposed in Carlini & Wagner (2016), which has been shown to attack more models with smaller perturbations than any other method published to date. We consider both targeted attacks and untargeted attacks.[5]

4.2.2 ADVERSARIAL ROBUSTNESS

There are multiple definitions of adversarial robustness in the literature. The most basic, which we shall use, is accuracy on adversarially perturbed versions of the test set.

It is also important to have a measure of the magnitude of the adversarial perturbation. Since adversaries are defined relative to human perception, the ideal measure would explicitly correspond to how easily a human observer would notice the perturbation. In lieu of such a measure, it is common to compute the size of the perturbation using L_0, L_1, L_2, and L_∞ norms (Szegedy et al., 2013; Goodfellow et al., 2014; Carlini & Wagner, 2016; Sabour et al., 2016). In particular, the L_0 norm

[3] There are also other adversaries that don't fall as cleanly into those categories, such as "fooling images" from Nguyen et al. (2014), which remove the human perceptual constraint, generating regular geometric patterns or noise patterns that networks confidently classify as natural images; and the idea of generating adversaries by stochastic search for images near the decision boundary of multiple networks from Baluja et al. (2015).

[4] Sabour et al. (2016) proposes a variant of the targeted attack, $A(X_S, M, X_T, k) \rightarrow X'_S$, where X_S is the source image, X_T is a target image, and k is a target layer in the model M. A produces X'_S by minimizing the difference in activations of M at layer k between X_T and X'_S. The end result of this attack for a classification network is still that $M(X'_S)$ yields a target label implicitly specified by X_T in a successful attack.

[5] Carlini & Wagner (2016) shared their code with us, which allowed us to perform the attack with exactly the same parameters they used for their paper, including the maximum number of iterations and maximum C value (see their paper for details).

measures the number of perturbed pixels, the L_2 norm measures the Euclidean distance between X and X', and the L_∞ norm measures the largest single change to any pixel.

4.2.3 EXPERIMENTAL SETUP

We used the same model architectures as in Section 4.1, using a $K = 256$ bottleneck. The architectures included a deterministic (base) model trained by MLE; a deterministic model trained with dropout (the dropout rate was chosen on the validation set); and a stochastic model trained with VIB for various values of β.

For the VIB models, we use 12 posterior samples of Z to compute the predictive distribution $p(y|x)$. This helps ensure that the adversaries can get a consistent gradient when constructing the perturbation, and that they can get a consistent evaluation when checking if the perturbation was successful (i.e., it reduces the chance that the adversary "gets lucky" in its perturbation due to an untypical sample). We also ran the VIB models in "mean mode", where the σs are forced to be 0. This had no noticeable impact on the results, so all reported results are for normal stochastic evaluation.

4.2.4 RESULTS AND DISCUSSION

We selected the first 10 zeros in the MNIST test set, and use the L2 optimization adversary of Carlini & Wagner (2016) to try to perturb those zeros into ones.[6] Some sample results are shown in Figure 3. We see that the deterministic models are easily fooled by making small perturbations, but for the VIB models with reasonably large β, the adversary often fails to find an attack (indicated by the green borders) within the permitted number of iterations. Furthermore, when an attack is succesful, it needs to be much larger for the VIB models. To quantify this, Figure 4(a) plots the magnitude of the perturbation (relative to that of the deterministic model) needed for a successful attack as a function of β. As β increases, the L_0 norm of the perturbation decreases, but both L_2 and L_∞ norms increase, indicating that the adversary is being forced to put larger modifications into fewer pixels while searching for an adversarial perturbation.

Figure 4(b) plots the accuracy on adversarially perturbed versions of the first 1000 images of the MNIST test set as a function of β. Each point in the plot corresponds to 3 separate executions of three different models trained with the same value of β. All models tested achieve over 98.4% accuracy on the unperturbed MNIST test set, so there is no appreciable measurement distortion due to underlying model accuracy.

We try both untargeted and targeted attacks. For targeting, we generate a random target label different from the source label in order to avoid biasing the results with unevenly explored source/target pairs. We see that for a reasonably broad range of β values, the VIB models have significantly better accuracy on the perturbed test set than the deterministic models, which have an accuracy of 0% (the attack of Carlini & Wagner (2016) is very effective on traditional model architectures).

Figure 4(b) also reveals a surprising level of adversarial robustness even when $\beta \to 0$. This can be explained by the theoretical framework of Fawzi et al. (2016). Their work proves that quadratic classifiers (e.g., $x^\mathsf{T} Ax$, symmetric A) have a greater capacity for adversarial robustness than linear classifiers. As we show in Appendix B, our Gaussian/softmax encoder/decoder is approximately quadratic for all $\beta < \infty$.

5 FUTURE DIRECTIONS

There are many possible directions for future work, including: testing on real images; using richer parametric marginal approximations, rather than assuming $r(z) = \mathcal{N}(0, I)$; exploring the connections to differential privacy (see e.g., Wang et al. (2016); Cuff & Yu (2016)); and investigating open universe classification problems (see e.g., Bendale & Boult (2015)). In addition, we would like to explore applications to sequence prediction, where X denotes the past of the sequence and Y the future, while Z is the current representation of the network. This form of the information bottleneck is known as predictive information (Bialek et al., 2001; Palmer et al., 2015).

[6] We chose this pair of labels since intuitively zeros and ones are the digits that are least similar in terms of human perception, so if the adversary can change a zero into a one without much human-noticeable perturbation, it is unlikely that the model has learned a representation similar to what humans learn.

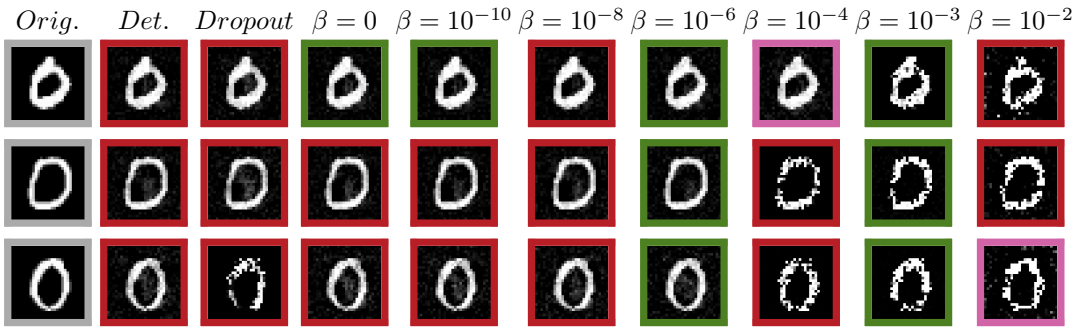
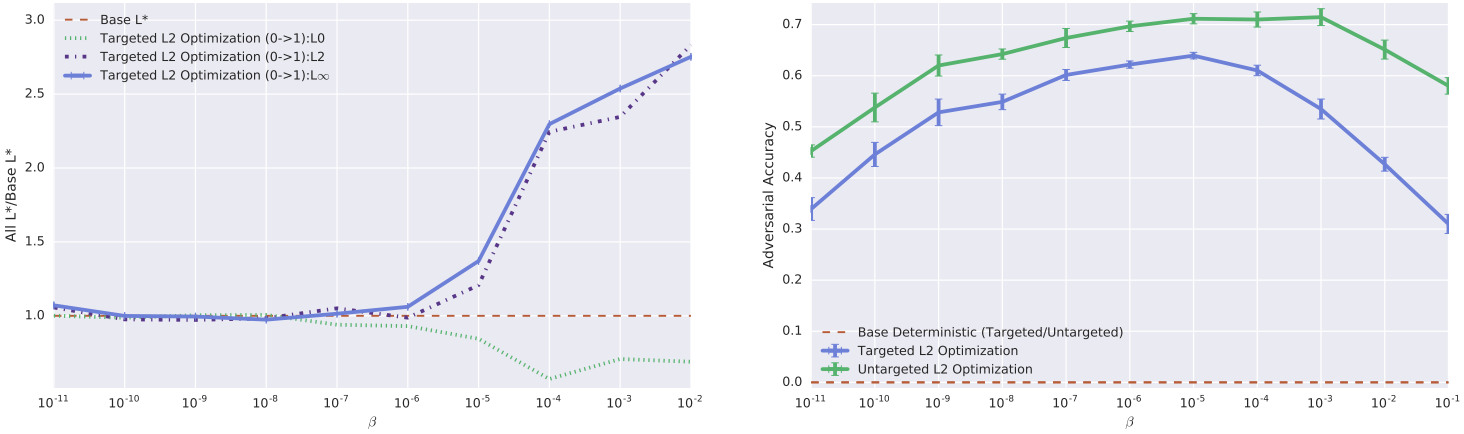


Figure 3: The adversary is trying to force each digit to be classified as class 1. Successful attacks have a red background. Unsuccessful attacks have a green background. In the case that the label is changed to an incorrect label different from the target label, the background is purple. The first column is the original image. The second column is our deterministic baseline model. The third column is our dropout model. The remaining columns are VIB models for different β.



(a) (b)

Figure 4: (a) Relative magnitude of the perturbation, measured using L_0, L_2 and L_∞ norms, for the images in Figure 3 as a function of β. (We normalize all values by the corresponding norm of the perturbation against the base model.) As β increases, L_0 decreases, but both L_2 and L_∞ increase, indicating that the adversary is being forced to put larger modifications into fewer pixels while searching for an adversarial perturbation. (b) Classification accuracy on L_2 adversarially perturbed images (of all classes) as a function of β. The blue line is for targeted attacks, and the green line is for untargeted attacks (which are easier to resist). In this case, $\beta = 10^{-11}$ has performance indistinguishable from $\beta = 0$. The deterministic model has a classification accuracy of 0% in both the targeted and untargeted attack scenarios, indicated by the horizontal red dashed line at the bottom of the plot.

REFERENCES

David Barber Felix Agakov. The IM algorithm: a variational approach to information maximization. In *NIPS*, volume 16, 2004.

Shumeet Baluja, Michele Covell, and Rahul Sukthankar. The virtues of peer pressure: A simple method for discovering high-value mistakes. In *Intl. Conf. Computer Analysis of Images and Patterns*, 2015.

Abhijit Bendale and Terrance Boult. Towards open world recognition. In *CVPR*, 2015.

William Bialek, Ilya Nemenman, and Naftali Tishby. Predictability, complexity, and learning. *Neural computation*, 13(11):2409–2463, 2001.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *ICML*, 2015.

Ryan P. Browne and Paul D. McNicholas. Multivariate sharp quadratic bounds via Σ-strong convexity and the fenchel connection. *Electronic Journal of Statistics*, 9, 2015.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. *Arxiv*, 2016.

Matthew Chalk, Olivier Marre, and Gasper Tkacik. Relevant sparse codes with variational information bottleneck. In *NIPS*, 2016.

G. Chechik, A Globersonand N. Tishby, and Y. Weiss. Information bottleneck for gaussian variables. *J. of Machine Learning Research*, 6:165188, 2005.

Paul Cuff and Lanqing Yu. Differential privacy as a mutual information constraint. In *ACM Conference on Computer and Communications Security (CCS)*, 2016.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *NIPS*, 2016.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Irina Higgins, Loic Matthey, Xavier Glorot, Arka Pal, Benigno Uria, Charles Blundell, Shakir Mohamed, and Alexander Lerchner. Early visual concept learning with unsupervised deep learning. *arXiv preprint 1606.05579*, 2016.

Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. Learning with a strong adversary. *CoRR*, abs/1511.03034, 2015.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *ICLR*, 2014.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *NIPS*, pp. 2125–2133, 2015.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *Arxiv*, 2016.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, 2016.

Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *CoRR*, abs/1412.1897, 2014.

Stephanie E Palmer, Olivier Marre, Michael J Berry, and William Bialek. Predictive information in a sensory population. *PNAS*, 112(22):6908–6913, 2015.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *Proceedings of the 1st IEEE European Symposium on Security and Privacy*, 2015.

G. Pereyra, G. Tuker, L. Kaiser, and G. Hinton. Regularizing neural networks by penalizing confident output predictions, 2016. Submitted.

Leigh Robinson and Benjamin Graham. Confusing deep convolution networks by relabelling. *arXiv preprint 1510.06925*, 2015.

Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. In *ICLR*, 2016.

Noam Slonim, Gurinder Singh Atwal, Gašper Tkačik, and William Bialek. Information-based clustering. *PNAS*, 102(51):18297–18302, 2005.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.

N Tishby and N Zaslavsky. Deep learning and the information bottleneck principle. In *IEEE Information Theory Workshop*, pp. 1–5, April 2015a.

N. Tishby, F.C. Pereira, and W. Biale. The information bottleneck method. In *The 37th annual Allerton Conf. on Communication, Control, and Computing*, pp. 368–377, 1999.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *Information Theory Workshop (ITW), 2015 IEEE*, pp. 1–5. IEEE, 2015b.

Weina Wang, Lei Ying, and Junshan Zhang. On the relation between identifiability, differential privacy and Mutual-Information privacy. *IEEE Trans. Inf. Theory*, 62:5018–5029, 2016.

A CONNECTION TO VARIATIONAL AUTOENCODERS

We can also consider unsupervised versions of the information bottleneck objective. Consider the objective:

$$\max I(Z, X) - \beta I(Z, i), \tag{19}$$

similar to the information theoretic objective for clustering introduced in Slonim et al. (2005).

Here the aim is to take our data X and maximize the mutual information contained in some encoding Z, while restricting how much information we allow our representation to contain about the identity of each data element in our sample (i). We will form a bound much like we did in the main text. For the first term, we form a variational decoder $q(x|z)$ and take a bound:

$$\begin{aligned} I(Z, X) &= \int dx\, dz\, p(x, z) \log \frac{p(x|z)}{p(x)} && (20)\\ &= H(x) + \int dz\, p(x) \int dx\, p(x|z) \log p(x|z) && (21)\\ &\geq \int dz\, p(x) \int dx\, p(x|z) \log q(x|z) && (22)\\ &= \int dx\, p(x) \int dz\, p(x|z) \log q(x|z). && (23) \end{aligned}$$

Here we have dropped the entropy in our data $H(X)$ because it is out of our control and we have used the nonnegativity of the Kullbach-Leibler divergence to replace our intractable $p(x|z)$ with a variational decoder $q(x|z)$.

Turning our attention to the second term, note that:

$$p(z|i) = \int dx\, p(z|x) p(x|i) = \int dx\, p(z|x) \delta(x - x_i) = p(z|x_i), \tag{24}$$

and that we will take $p(i) = \frac{1}{N}$.

So that we can bound our second term from above

$$\begin{aligned} I(Z, i) &= \sum_i \int dz\, p(z|i) p(i) \log \frac{p(z|i)}{p(z)} && (25)\\ &= \frac{1}{N} \sum_i \int dz\, p(z|x_i) \log \frac{p(z|x_i)}{p(z)} && (26)\\ &\leq \frac{1}{N} \sum_i \int dz\, p(z|x_i) \log \frac{p(z|x_i)}{r(z)}, && (27) \end{aligned}$$

Where we have replaced the intractable marginal $p(z)$ with a variational marginal $r(z)$.

Putting these two bounds together we have that our unsupervised information bottleneck objective takes the form

$$I(Z, X) - \beta I(Z, i) \leq \int dx\, p(x) \int dz\, p(z|x) \log q(x|z) - \beta \frac{1}{N} \sum_i \mathrm{KL}[p(Z|x_i), r(Z)]. \tag{28}$$

And this takes the form of a variational autoencoder (Kingma & Welling, 2014), except with the second KL divergence term having an arbitrary weight β.

This precise setup, albeit with a different motivation was recently explored in Higgins et al. (2016), where they demonstrated that by changing the weight of the variational autoencoders regularization term, there were able to achieve latent representations that were more capable when it came ot zero-shot learning and understanding "objectness". In that work, they motivated their choice to change the relative weightings of the terms in the objective by appealing to notions in neuroscience. Here we demonstrate that appealing to the information bottleneck objective gives a principled motivation

and could open the door to better understanding the optimal choice of β and more tools for accessing the importance and tradeoff of both terms.

Beyond the connection to existing variational autoencoder techniques, we note that the unsupervised information bottleneck objective suggests new directions to explore, including targetting the exact marginal $p(z)$ in the regularization term, as well as the opportunity to explore tighter bounds on the first $I(Z, X)$ term that may not require explicit variational reconstruction.

B QUADRATIC BOUNDS FOR STOCHASTIC LOGISTIC REGRESSION DECODER

Consider the special case when the bottleneck Z is a multivariate Normal, i.e., $z|x \sim N(\mu_x, \Sigma_x)$ where Σ_x is a $K \times K$ positive definite matrix. The parameters μ_x, Σ_x can be constructed from a deep neural network, e.g.,

$$\begin{aligned}\mu_x &= \gamma_{1:K}(x)\\ \mathrm{chol}(\Sigma_x) &= \mathrm{diag}(\log(1+\exp(\gamma_{K+1:2K}))) + \mathrm{subtril}(\gamma_{2K+1:K(K+3)/2}),\end{aligned}$$

where $\gamma(x) \in \mathbb{R}^{K(K+3)/2}$ is the network output of input x.

Suppose that the prediction is a categorical distribution computed as $\mathcal{S}(Wz)$ where W is a $C \times K$ weight matrix and $\log \mathcal{S}(x) = x - \mathsf{lse}(x)$ is the log-soft-max function with $\mathsf{lse}(x) = \log \sum_{k=1}^{K} \exp(x_k)$ being the log-sum-exp function.

This setup (which is identical to our experiments) induces a classifier which is bounded by a quadratic function, which is interesting because the theoretical framework Fawzi et al. (2016) proves that quadratic classifiers have greater capacity for adversarial robustness than linear functions.

We now derive an approximate bound using second order Taylor series expansion (TSE). The bound can be made proper via Browne & McNicholas (2015). However, using the TSE is sufficient to sketch the derivation.

Jensen's inequality implies that the negative log-likelihood soft-max is upper bounded by:

$$\begin{aligned}-\log \mathsf{E}\left[\mathcal{S}(WZ)|\mu_x, \Sigma_x\right] &\leq -\mathsf{E}\left[\log \mathcal{S}(WZ)|\mu_x, \Sigma_x\right]\\ &= -W\mu_x + \mathsf{E}\left[\mathsf{lse}(WZ)|\mu_x, \Sigma_x\right]\\ &= -W\mu_x + \mathsf{E}\left[\mathsf{lse}(Z)|W\mu_x, W\Sigma_x\right].\end{aligned}$$

The second order Taylor series expansion (TSE) of lse is given by,

$$\mathsf{lse}(x+\delta) \approx \mathsf{lse}(x) + \delta^{\mathsf{T}}\,\mathcal{S}(x) + \tfrac{1}{2}\delta^{\mathsf{T}}\left[\mathrm{diag}(\mathcal{S}(x)) - \mathcal{S}(x)\mathcal{S}(x)^{\mathsf{T}}\right]\delta.$$

Taking the expectation of the TSE at the mean yields,

$$\begin{aligned}\mathsf{E}_{N(0,W\Sigma_x W^{\mathsf{T}})}[\,\mathsf{lse}(W\mu_x+\delta)] &\approx \mathsf{lse}(W\mu_x) + \mathsf{E}_{N(0,W\Sigma_x W^{\mathsf{T}})}[\delta^{\mathsf{T}}]\mathcal{S}(W\mu_x) +\\ &\quad + \tfrac{1}{2}\,\mathsf{E}_{N(0,W\Sigma_x W^{\mathsf{T}})}[\delta^{\mathsf{T}}\left[\mathrm{diag}(\mathcal{S}(W\mu_x)) - \mathcal{S}(W\mu_x)\mathcal{S}(W\mu_x)^{\mathsf{T}}\right]\delta]\\ &= \mathsf{lse}(W\mu_x) + \tfrac{1}{2}\,\mathrm{tr}(W\Sigma_x W^{\mathsf{T}}\left[\mathrm{diag}(\mathcal{S}(W\mu_x)) - \mathcal{S}(W\mu_x)\mathcal{S}(W\mu_x)^{\mathsf{T}}\right])\\ &= \mathsf{lse}(W\mu_x) + \tfrac{1}{2}\,\mathrm{tr}(W\Sigma_x W^{\mathsf{T}}\,\mathrm{diag}(\mathcal{S}(W\mu_x))) - \tfrac{1}{2}\mathcal{S}(W\mu_x)^{\mathsf{T}}\,W\Sigma_x W^{\mathsf{T}}\,\mathcal{S}(W\mu_x)\\ &= \mathsf{lse}(W\mu_x) + \tfrac{1}{2}\sqrt{\mathcal{S}(W\mu_x)}^{\mathsf{T}}\,W\Sigma_x W^{\mathsf{T}}\,\sqrt{\mathcal{S}(W\mu_x)} - \tfrac{1}{2}\mathcal{S}(W\mu_x)^{\mathsf{T}}\,W\Sigma_x W^{\mathsf{T}}\,\mathcal{S}(W\mu_x)\end{aligned}$$

The second-moment was calculated by noting,

$$\mathsf{E}[X^{\mathsf{T}}\,BX] = \mathsf{E}\,\mathrm{tr}(XX^{\mathsf{T}}\,B) = \mathrm{tr}(\mathsf{E}[XX^{\mathsf{T}}]B) = \mathrm{tr}(\Sigma B).$$

Putting this altogether, we conclude,

$$\mathsf{E}\left[\mathcal{S}(WZ)|\mu_x, \Sigma_x\right] \gtrapprox \mathcal{S}(W\mu_x)\exp\left(-\tfrac{1}{2}\sqrt{\mathcal{S}(W\mu_x)}^{\mathsf{T}}\,W\Sigma_x W^{\mathsf{T}}\,\sqrt{\mathcal{S}(W\mu_x)} + \tfrac{1}{2}\mathcal{S}(W\mu_x)^{\mathsf{T}}\,W\Sigma_x W^{\mathsf{T}}\,\mathcal{S}(W\mu_x)\right).$$

As indicated, rather than approximate the lse via TSE, we can make a sharp, quadratic upper bound via Browne & McNicholas (2015). However this merely changes the $\mathcal{S}(W\mu_x)$ scaling in the exponential; the result is still log-quadratic.