Semi-Supervised Training of Structured Output Neural Networks with an Adversarial Loss

Mateusz Koziński Loïc Simon Frédéric Jurie

Groupe de recherche en Informatique, Image, Automatique et Instrumentation de Caen Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France

mateusz.kozinski@unicaen.fr loic.simon@ensicaen.fr frederic.jurie@unicaen.fr

Abstract

We propose a method for semi-supervised training of structured-output neural networks. Inspired by the framework of Generative Adversarial Networks (GAN), we train a discriminator network to capture the notion of 'quality' of network output. To this end, we leverage the qualitative difference between outputs obtained on the labelled training data and unannotated data. We then use the discriminator as a source of error signal for unlabelled data. This effectively boosts the performance of a network on a held out test set. Initial experiments in image segmentation demonstrate that the proposed framework enables labelling two times less data than in a fully supervised scenario, while achieving the same network performance.

1 Introduction

The unprecedented power that neural networks offer when applied to vision problems comes at a cost of large volumes of annotated training data required from training. When the annotations are produced manually the process can be laborious and costly, especially for structured output problems like image segmentation.

In this paper we propose an approach to semi-supervised training of structured output neural networks. The proposed approach allows to capitalize on large sets of unlabelled data. We show that the performance of a network trained in a fully supervised regime on a certain amount of labelled data can be matched by using a significantly smaller amount of labelled data, together with a sufficiently large volume of unlabelled data. In consequence, significant labelling effort can be saved.

In technical terms, we generate a useful error signal for data for which no ground truth labels are available, by means of adversarial training. During training, both the labelled training data and the unlabelled data is forwarded through the network. The network produces qualitatively better output on the labelled images than on the unlabelled images. Much like in training a Generative Adversarial Network (GAN), we train a discriminator network to capture this difference. The negative gradient of the discriminator with respect to its unlabelled input is used as the error signal for the unlabelled data.

Our technical contribution consists in an adversarial learning approach for semi-supervised training of structured output neural networks. A particular advantage of our method is that it can be applied to any structured output problem, independently of the architecture of the applied predictor. Contrary to pre-training, the proposed method can be applied to a complete network, not just to its feature-extracting part.

2 Related work

Our work is related to previous efforts to use unannotated data for training neural networks, including autoencoders, self-supervised learning and the use of GANs.

A considerable research effort has been devoted to autoencoders [7, 11] - neural networks that encode an image into a latent representation, from which they attempt to reconstruct the original image. Different regularization techniques are applied to impose useful properties on the hidden representation. The encoder of a trained autoencoder is considered a useful 'feature extractor'. In a pre-training scenario [21, 16] the encoder of a trained autoencoder is incorporated as a feature extractor into another network, which is then fine-tuned for a particular task on labelled data. In a semi-supervised scenario [23, 22] parameters are shared between an encoder of an autoencoder and a feature extractor of a supervised network, and both are trained simultaneously. Plain autoencoders attempt to encode all the visual information in the latent representation. It has been hypothesised that much of the information is irrelevant for particular vision tasks, and autoencoders that transfer some of it between the input and the output [16, 21, 22], instead of encoding everything in the latent representation, produce more useful representations. A recent example of such architecture is the ladder network [20, 17], where the critical information content that should be encoded in the latent representation is learnt in a semi-supervised setting. One drawback of autoencoders is that they constrain the ar-

chitecture of the supervised network to be the same as that of the encoder. While the convolutional autoencoders [11] with pooling layers [16, 21, 22] match the architectures of contemporary image classification networks [19] well, they can only be matched to an initial part of a structured output network, for example one used for image segmentation [12, 1]. In consequence, the other part of a network does not benefit from the unlabelled data. This is consistent with the intuition that the 'final' part of such network, that upsamples feature maps, represents a correlation between the output variables. Such correlation cannot be learnt by an autoencoder that is never exposed to any ground truth annotations.

A number of 'self-supervised' methods of training feature extractors [2, 14, 13] emerged recently. They consist in 'deconstructing' unlabelled images by removing some information, and training a neural network to reproduce the original image. The 'deconstruction' methods include masking image regions, or dividing an image into tiles and shuffling the resulting tiles. The corresponding reconstruction tasks include inpainting the masked regions based on the context [14] and guessing a relative position of two or more tiles [2, 13]. The reconstruction requires extracting high-level information from the image, which makes the obtained 'feature extractors' useful for other vision tasks. However, from the perspective of structured-output tasks, the 'self-supervised' methods suffer from the same drawbacks as autoencoders: they constrain the architecture of the trained network and are not suitable for capturing dependencies between output variables.

Our unsupervised objective is inspired by the Generative Adversarial Networks (GANs) [6]. In GAN, a generator network is trained to transform a random vector originating from a simple sampling distribution to a sample from a complicated target distribution. The flagship application is to train a generator to yield realistically looking images from random vectors. The interesting property of GANs is that all that is required for training the generator is a collection of vectors originating from the target distribution. The error signal is backpropagated to the generator from a discriminator network that attempts to differentiate between vectors originating from the true target distribution and the ones generated by the trained network. The generator and the discriminator are trained in an alternating manner. Theoretically, GAN training has been shown to be an instance of a saddle point problem. GANs are difficult to train in practice, and some work has been devoted to improving their behaviour during training [18], identifying architectures that work well in GANs [15], and generalizing the discriminator from a binary classifier to an energy function [24].

A number of attempts at using GANs for unsupervised learning has been made recently [18, 15]. In the simplest case, the initial layers of a discriminator are used as a feature extractor [15]. In the task of image classification, the generated images can constitute a new class of input images [18], augmenting the total number of training images.

GANs can also be used for mapping directly between two domains of interest. In this case discrimination is performed between pairs of input and output. A recent work [8] showcased learning such a mapping between artistic depictions and photographs, images taken during day and night time, or segmentations and the corresponding images. The discriminator differentiates pairs of input and the corresponding ground truth output from pairs of input and output generated by the network. This new, learned cost function is shown to give more visually plausible results than the standard L2 reconstruction loss. However, it performed worse than the baseline loss on mappings with less output ambiguity, like the mapping from images to segmentation maps. The same type of loss has been demonstrated to boost segmentation results when combined with a standard cost function [10]. The methods are fully supervised - the adversarial criterion is evaluated for the labelled training data. In contrast, we use a discriminator specifically to generate a training signal for unlabelled data.

Another body of research [3, 4] proves that mappings between the latent space and the data space can be learnt with just samples from both domains, and that corresponding input-output pairs are not necessarily needed. Two networks are trained simultaneously: a generator $g()$, producing samples from a latent representation z, and an encoder $f()$, inferring the latent representation from data x coming from the target distribution. Discrimination is performed between two types of pairs: a pair of generated data and the corresponding latent vector $(g(z), z)$, and a pair of data originating from the target distribution, and the inferred latent representation $(x, f(x))$. At optimality $g()$ and $f()$ are proven to be inverses. However, the methods have only been shown to work on low-dimensional inputs and latent representations.

The discrimination-based approach can also be applied to domain adaptation [5, 9]. The discriminator is learning to differentiate between features obtained for samples from two different domains, like synthetic and real images, or images representing different seasons of year. The discriminator is a source of an error signal, that makes the network produce similar features on data coming from both domains. The method can be used for object detection and image classification [5], and for semantic segmentation [9]. While the network architecture used in domain adaptation is similar to ours, the concept behind the methods is substantially different. The goal of domain adaptation is to make the network insensitive to certain shift in the input data. In contrast, our goal is to regularize the network with use of the unlabelled data.

3 Method description

We address the problem of training a structured output network f_w, parametrised with a weight vector w, to produce predictions y on input data x. The x and y can take any form as long as they can be input and output by a neural

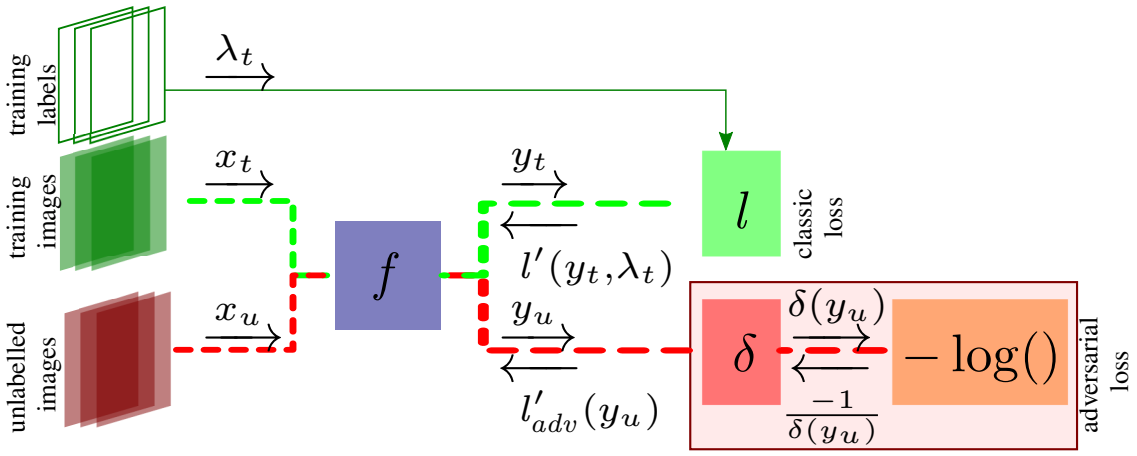

Figure 1: The flow of data and error signals when training a structured output network f with the proposed method, presented in algorithm 1. The discriminator update is not shown in the drawing. The green line denotes the flow of labelled training data and the corresponding gradients. The red line denotes the flow of unlabelled data and the corresponding gradients. By $l'(y, \lambda)$ we denote the partial derivative of the loss with respect to the prediction, $l'(y, \lambda) = \nabla_y l(y, \lambda)$.

network. We are targeting a scenario in which, in addition to a set of training examples x_t, $t \in \mathcal{T}$, with annotations λ_t, a volume of unlabelled examples x_u, for $u \in \mathcal{U}$, is available. To handle the unlabelled data, our approach combines a classic supervised loss $l(y_t, \lambda_t)$, measuring the consistency of y_t and λ_t, with a novel and unsupervised one $l_{adv}(y_u)$. The new loss term is described in section 3.1. We define a global cost consisting of two components

$$\begin{aligned} C_{tot}(w) =& C(w) + \alpha C_{adv}(w) \\ =& \mathbb{E}[l(f(x_t), \lambda_t)] + \alpha \mathbb{E}[l_{adv}(f(x_u)]\,, \end{aligned} \tag{1}$$

where α is a constant. Training consists in determining the optimal network parameter by solving

$$w^* = \arg\min_w C_{tot}(w). \tag{2}$$

We describe the training algorithm in section 3.2

3.1 Adversarial loss

Defining a loss function, that measures network performance based only on its outputs seems infeasible. On the other hand, given two image segmentations, output by a neural network at two sufficiently distant training epochs, a human can spot the perceptual difference corresponding to increasing accuracy. This suggests that there might exist a measure of 'output quality', applicable at least for a certain range of accuracy. In this section, we attempt to construct such a function.

In a classical setting, training a network on the labelled training data (x_t, λ_t), $t \in \mathcal{T}$ results in a qualitative difference between outputs $y_t = f_w(x_t)$ and outputs produced for the unseen data $y_u = f_w(x_u)$, $u \in \mathcal{U}$. Ideally, x_t and x_u are identically distributed, so one might think that the same holds for $f_w(x_t)$ and $f_w(x_u)$. In practice, the dimensionality of x is typically large and the training set is not necessarily representative of the variability present in some unseen data. This biases f_w towards better performance on the training examples.[1] We leverage this qualitative difference to define the unsupervised cost C_{adv} as a regularisation term that tends to close this gap.

Inspired by the GAN framework, we propose to train a discriminator network δ_v, parametrised by v, to capture the qualitative difference between $f_w(x_t)$ for $t \in \mathcal{T}$, and $f_w(x_u)$, for $u \in \mathcal{U}$. We use a binary discriminator. We interpret the scalar output $\delta_v(y)$ as the likelihood that y has been obtained from an element of the labelled training set $y = f_w(x_t)$, $t \in \mathcal{T}$, and we interpret $1 - \delta_v(y)$ as the likelihood of y originating from the unlabelled set $y = f_w(x_u)$, $u \in \mathcal{U}$. The optimal parameter configuration of the discriminator $v^* = \arg\min_v CE_{disc}(v)$ is defined in terms of the cross-entropy $CE_{disc}(v) = (-\mathbb{E}[\log(\delta_v(f(x_t)))] - \mathbb{E}[\log(1 - \delta_v(f(x_u)))])$.

When the discriminator is trained to optimum, its negative logarithm can be used as a local 'quality measure' for image segmentations. This is because $\delta_{v^*}(y)$ is the likelihood that y has been generated on the training set, and the outputs on the training set are qualitatively better. We therefore define the unsupervised cost as

$$C_{adv}(w) = \mathbb{E}[-\log(\delta_{v^*}(f_w(x_u))]\,. \tag{3}$$

Minimising (3) with respect to w drives f_w towards reducing the gap between performance on labelled training and unlabelled data. It is however important to do so in a one-way manner, so as to avoid making the performance of f_w degrade on the labelled examples. Clearly, we want f_w to perform as well on unlabelled examples as it does on labelled ones but not the other way around. Therefore, we

[1] Note that we are not referring to the phenomenon of overfitting, where a mismatch of model complexity to the size of the training set can cause a situation where decreasing the training objective results in increasing the error on a held out test data, but simply to the higher performance of the network on the training set.

apply the adversarial component of the cost function only to the unlabelled data.

3.2 Algorithm

The minimization can be performed with a gradient-based optimization routine, for example SGD. The gradient of the objective consists of two components and its estimate on a training batch T and unlabelled batch U can be denoted as

$$\nabla_w C^{TU}_{tot}(w) = \nabla_w C^T(w) + \alpha \nabla_w C^U_{adv}(w). \quad (4)$$

The gradient can be computed by backpropagation. The flow of data and gradients forward and back through the networks is depicted in Figure 3. In practice, we train the network using algorithm 1. The $\mathrm{update}(w, g)$ procedure accepts the network weights w and a gradient of the cost function g and performs an update on w. While we used SGD with momentum, any update rule used for training neural networks is applicable. Instead of training the discriminator to optimality at each iteration, we perform k updates of the discriminator for a single update of the network f_w itself. There is no guarantee of convergence of the algorithm. However, our experiments demonstrate its practical utility.

Algorithm 1 Training a structured output network with adversarial cost for unlabelled data

$v, w \leftarrow \mathrm{randInit}()$
while not converged **do**
  for $IterNum = 1$ to k **do**
    $T \leftarrow \mathrm{pickBatch}(\mathcal{T}, batchSize)$
    $U \leftarrow \mathrm{pickBatch}(\mathcal{U}, batchSize)$
    $g \leftarrow -\sum_{t \in T} \nabla_v \log(\delta_v(f_w(x_t))) - \sum_{u \in U} \nabla_v \log(1 - \delta_v(f_w(x_u)))$
    $v \leftarrow \mathrm{update}(v, g)$
  end for
  $T \leftarrow \mathrm{pickBatch}(\mathcal{T}, batchSize)$
  $g_T \leftarrow \sum_{t \in T} \nabla_w l(f_w(x_t))$
  $U \leftarrow \mathrm{pickBatch}(\mathcal{U}, batchSize)$
  $g_U \leftarrow -\sum_{u \in U} \nabla_w \log(\delta_v(f_w(x_u)))$
  $w \leftarrow \mathrm{update}(w, g_T + \alpha g_U)$
end while

4 Experimental evaluation

The goal of the experiments is to compare semi-supervised training with the adversarial loss to fully supervised training. In particular, we are interested in the trade-off between the labelling effort and the performance of the trained network. The question we are asking is: given a collection of training input data and having labelled a part of it, what is the benefit, in terms of network performance, of labelling a certain portion of it with respect to using it in an unsupervised manner. Knowing the answer helps to take the decision whether the expense of manpower required to label a certain portion of data is worth the increase in performance that it can bring.

We run experiments according to the following pattern. We run the baseline method on the whole training set, and on the training set consisting of $\frac{1}{2}$, $\frac{1}{4}$ and $\frac{1}{8}$ images of the original training set. Then, we apply the proposed method, with $\frac{1}{2}$, $\frac{1}{4}$ and $\frac{1}{8}$ of the training set used for the supervised subtask, and the remaining part of the training set used in an unsupervised manner.

We use the CamVid dataset, in the version used by Badrinarayanan, Kendall and Cipolla [1]. It consists of images captured by a forward-looking vehicle-mounted camera, of the size of 360×480 pixels. The datasets contains of 367 training, 101 validationframe and 233 test images. The set of labels consists of 11 classes. The compact size of the dataset lets us run a number of experiments in reasonable time.

We use the segnet-basic network [1]. It has an encoder-decoder architecture, where the encoder consists of four blocks of architecture `64c(7)-BN-ReLU-MP(2)`, where `Nc(K)` denotes a layer of `N` convolutional filters of size `K×K`, applied with output stride of one pixel in both directions, `BN` denotes the batch normalization, `ReLU` denotes the Rectified Linear Unit and `MP(K)` denotes the max-pooling operation performed in windows of size `K×K` with the output stride of `K` pixels. Each encoder block effectively subsamples the feature map by a factor of 2 in both dimensions. The decoder consists of four blocks of `MU(2)-64c(7)-BN`, where `MU(K)` denotes a max-unpooling layer, with the output stride of `K×K`, where the unpooling indeces are transfered from the symmetric max-pooling layer in the encoder. Each block of the decoder effectively upsamples the feature map by a factor of 2 in both dimensions. The output is produced by a 1x1 convolutional layer with 11 filters and a stride of 1 in both directions. We refer the reader to the original work [1] for a more detailed explanation of the encoder-decoder architecture with coupled pooling and unpooling layers.

The discriminator consists of three blocks of `64c(3,2)-BN-LReLU`, followed by a global average pooling layer and the final linear layer with a single output variable. By `64c(3,2)` we denote a convolutional layer consisting of 64 filters of size `3×3` and an output stride of `2×2`, and `LReLU` denotes a rectified linear unit with the slope of 0.2 for the 'deactivated' piece. When performing the experiments we found out that an important aspect of the discriminator architecture is the global pooling layer. Its effect is similar to per-pixel discriminator ground truth used in [8] and consists in preventing the discriminator from overfitting by memorizing the contents of labelled training and unlabelled images. The discriminator is binary and trained using a cross-entropy loss.

We use the basic SGD algorithm for updating both the segmentation network and the discriminator. We use momentum of 0.9 and weight decay of 0.001. For both the baseline and the proposed approach we train the network for 10^4 iterations with a learning rate of 0.1, then for $4 \cdot 10^3$ iterations with a learning rate of 0.05, another $4 \cdot 10^3$ with a learning

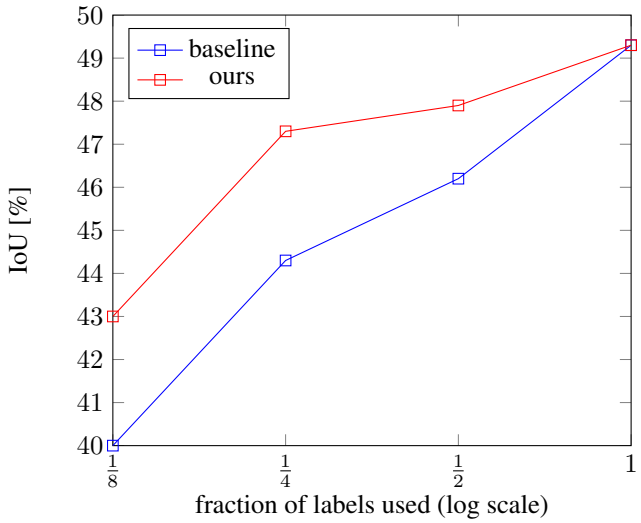

Figure 2: The IoU attained by segnet-basic on the CamVid dataset with respect to the number of annotations used for training. Note that the proposed method almost matches the baseline with two times as many annotations.

rate of 0.025 and finally for $2 \cdot 10^3$ iterations with a learning rate of 0.0125. We jitter the training images by shifting them by a random number of between 0 and 32 pixels in both directions. We perform the accuracy tests on the network defined by the weight vector after the last update in this procedure, instead of cherry-picking the model using a cross validation on the validation set. We found out that this strategy gave better results for both the baseline and the proposed algorithm.

When using our method, we set $k = 1$, that is, we update the discriminator once per every update of the trained network. We use batches of 16 training images for the baseline, and batches of 8 training and 8 unlabelled images for the semi-supervised setting.

We present numerical results in table 1 and in figure 2. The baseline attains an accuracy of 49.3% Intersection-over-Union (IoU), which exceeds the performance of 47.7% reported in the original paper [1]. We suspect the increase comes from the differences in the training protocol, including jitter and a decreasing learning rate. Our method consistently outperforms the baseline. Besides for every ratio of supervision, the performance of our network is nearly as good as the baseline using twice as many labelled examples. For a ratio of $\frac{1}{4}$, our approach even improves on the baseline with $\frac{1}{2}$ annotations used.

5 Conclusion

In this work, we propose a novel kind of regularization technique that can be applied in a semi-supervised context. Our approach attempts to learn a good regularization cost that is guided by an adversarial scheme. The rational behind this choice is that, as a standard learning procedure goes, a trained network tends to perform better on training than on test data. We can therefore train a classifier to discriminate between labelled and unlabelled output distributions. Then, the likelihood estimate produced by the discriminator can be used as a signal to improve the behaviour of the main network on the unlabelled population.

We have leveraged the aforementioned principle to derive a generic framework that can be applied to a large number of typical machine learning problems where a structured output is generated. In order not to hinder the exposition, we have focused our experiments on a single study case, namely semantic segmentation. Nonetheless, our approach can be adapted seamlessly to different tasks, such as depth or normal inference. In the considered scenario, we have evaluated our method on a standard benchmark and demonstrated that our regularization achieves substantial improvements over a baseline using only labelled data. We have also studied the evolution of the performance in response to a varying ratio of supervision. One interesting observation of this study is that one can match the performance of a supervised baseline with our semi-supervised approach using half the amount of ground truth annotations.

References

[1] BADRINARAYANAN, V., KENDALL, A., AND CIPOLLA, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561* (2015).

[2] DOERSCH, C., GUPTA, A., AND EFROS, A. A. Unsupervised visual representation learning by context prediction. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015* (2015), pp. 1422–1430.

[3] DONAHUE, J., KRÄHENBÜHL, P., AND DARRELL, T. Adversarial feature learning. *CoRR abs/1605.09782* (2016).

[4] DUMOULIN, V., BELGHAZI, I., POOLE, B., LAMB, A., ARJOVSKY, M., MASTROPIETRO, O., AND COURVILLE, A. Adversarially learned inference. *CoRR abs/1606.00704* (2016).

[5] GANIN, Y., USTINOVA, E., AJAKAN, H., GERMAIN, P., LAROCHELLE, H., LAVIOLETTE, F., MARCHAND, M., AND LEMPITSKY, V. S. Domain-adversarial training of neural networks. *CoRR abs/1505.07818* (2015).

[6] GOODFELLOW, I., POUGET-ABADIE, J., MIRZA, M., XU, B., WARDE-FARLEY, D., OZAIR, S., COURVILLE, A., AND BENGIO, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680.

Table 1: Performance of segnet-basic on CamVid w.r.t. the fraction of used annotations. The baseline is a fully supervised SGD with momentum. The proposed method uses the reported fraction of annotated data and the remaining part of the training set without annotations. The performance measures: intersection over union (IoU), average per-class recall (C) and global precision (G).

subs. factor:	1			$\frac{1}{2}$			$\frac{1}{4}$			$\frac{1}{8}$		
	IoU	C	G	IoU	C	G	IoU	C	G	IoU	C	G
baseline	49.3	64.6	83.5	46.2	58.0	82.6	44.3	56.7	81.0	40.0	50.9	79.1
ours	—	—	—	47.9	60.0	83.4	47.3	58.8	82.3	43.0	53.3	81.9

[7] HINTON, G. E., AND SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. *Science 313*, 5786 (July 2006), 504–507.

[8] ISOLA, P., ZHU, J.-Y., ZHOU, T., AND EFROS, A. A. Image-to-image translation with conditional adversarial networks. *arxiv* (2016).

[9] JUDY HOFFMAN, DEQUAN WANG, F. Y., AND DARRELL, T. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *CoRR abs/1612.02649* (2016).

[10] LUC, P., COUPRIE, C., CHINTALA, S., AND VERBEEK, J. Semantic segmentation using adversarial networks. *CoRR abs/1611.08408* (2016).

[11] MASCI, J., MEIER, U., CIREŞAN, D., AND SCHMIDHUBER, J. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks* (2011), Springer, pp. 52–59.

[12] NOH, H., HONG, S., AND HAN, B. Learning deconvolution network for semantic segmentation. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015* (2015), pp. 1520–1528.

[13] NOROOZI, M., AND FAVARO, P. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI* (2016), pp. 69–84.

[14] PATHAK, D., KRÄHENBÜHL, P., DONAHUE, J., DARRELL, T., AND EFROS, A. A. Context encoders: Feature learning by inpainting. *CoRR abs/1604.07379* (2016).

[15] RADFORD, A., METZ, L., AND CHINTALA, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR abs/1511.06434* (2015).

[16] RANZATO, M., HUANG, F., BOUREAU, Y., AND LECUN, Y. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. Computer Vision and Pattern Recognition Conference (CVPR'07)* (2007), IEEE Press.

[17] RASMUS, A., VALPOLA, H., HONKALA, M., BERGLUND, M., AND RAIKO, T. Semi-Supervised Learning with Ladder Networks, Nov. 2015.

[18] SALIMANS, T., GOODFELLOW, I. J., ZAREMBA, W., CHEUNG, V., RADFORD, A., CHEN, X., AND CHEN, X. Improved techniques for training gans. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain* (2016), pp. 2226–2234.

[19] SIMONYAN, K., AND ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *CoRR abs/1409.1556* (2014).

[20] VALPOLA, H. From neural {PCA} to deep unsupervised learning. In *Advances in Independent Component Analysis and Learning Machines*, E. Bingham, S. Kaski, J. Laaksonen, and J. Lampinen, Eds. Academic Press, 2015, pp. 143 – 171.

[21] ZEILER, M. D., TAYLOR, G. W., AND FERGUS, R. Adaptive deconvolutional networks for mid and high level feature learning. In *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011* (2011), pp. 2018–2025.

[22] ZHANG, Y., LEE, K., AND LEE, H. Augmenting supervised neural networks with unsupervised objectives for large-scale image classification. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016* (2016), pp. 612–621.

[23] ZHAO, J., MATHIEU, M., GOROSHIN, R., AND LECUN, Y. Stacked what-where auto-encoders. *CoRR abs/1506.02351* (2015).

[24] ZHAO, J. J., MATHIEU, M., AND LECUN, Y. Energy-based generative adversarial network. *CoRR abs/1609.03126* (2016).