Matching Adversarial Networks

Gellért Máttyus and Raquel Urtasun
Uber Advanced Technologies Group and University of Toronto
gmattyus@uber.com, urtasun@uber.com

Abstract

Generative Adversarial Nets (GANs) and Conditonal GANs (CGANs) show that using a trained network as loss function (discriminator) enables to synthesize highly structured outputs (e.g. natural images). However, applying a discriminator network as a universal loss function for common supervised tasks (e.g. semantic segmentation, line detection, depth estimation) is considerably less successful. We argue that the main difficulty of applying CGANs to supervised tasks is that the generator training consists of optimizing a loss function that does not depend directly on the ground truth labels. To overcome this, we propose to replace the discriminator with a matching network taking into account both the ground truth outputs as well as the generated examples. As a consequence, the generator loss function also depends on the targets of the training examples, thus facilitating learning. We demonstrate on three computer vision tasks that this approach can significantly outperform CGANs achieving comparable or superior results to task-specific solutions and results in stable training. Importantly, this is a general approach that does not require the use of task-specific loss functions.

1. Introduction

GANs [10] have become extremely popular due to their ability to generate sharp, realistic images [2]. GANs train deep generative models using a minimax game. The idea is to learn a generator by fooling a discriminator which tries to distinguish between real and generated examples. CGANs [22] are an extension to model conditional distributions by making the generator and the discriminator a function of the input. This is a very interesting idea showing good results on image generation tasks. However, CGANs do not work well on common supervised tasks (e.g. semantic segmentation, instance segmentation, line detection), since the generator is optimized by minimizing a loss function that does not depend on the training examples. Practitioners try to tackle this issue by defining and adding a task dependent loss function to the objective. Unfortunately, it is very difficult to balance the two loss functions resulting in unstable and often poor training.

In this paper we propose to replace the discriminator with a siamese network. The inputs to the siamese network are the ground truth, the generated output or perturbations (random transformations) of these. The discriminator then attempts to predict whether or not the input pair contains the generated output (fake) or just the ground truth and its perturbations (real). As a consequence, the generator loss depends on the training targets, which results in better, faster and more robust learning. Applying random perturbations makes the task of the discriminator more difficult, while, as we show in the technical section, the generator target still remains the ground truth. We call our approach Matching Adversarial Network (MatAN) which can be used as a discriminative model for supervised tasks.

Our experimental evaluation shows that this approach can achieve very good performance in the tasks of semantic segmentation, road network centerline extraction from aerial images and instance segmentation. In particular, we significantly outperform CGANs and achieve comparable or superior results to supervised approaches that exploit task-specific solutions. The training of MatAN was stable (not resulting in degenerative output) in all our experiments, even with different generator and discriminator architectures. This is an important advantage over CGANs which are sensitive to the applied network architectures.

2. Related Work

Generative Adversarial Networks (GANs) [10] can create very realistic images; however, the stability of the training is an issue. To overcome this, many methods attempt to improve the training performance of GANs. Radford et al. [26] proposed to use specific network architectures, including transposed convolution and leaky ReLu, which result in better image generation. [28] introduces several practices for more stable training, e.g. minibatch discrimination. [1] analyzes the reasons for the instability of GAN training. [2] proposed the Wasserstein GAN, which achieves better training by using a discriminator (also called critic) which estimates the earth mover's distance between the probabil-

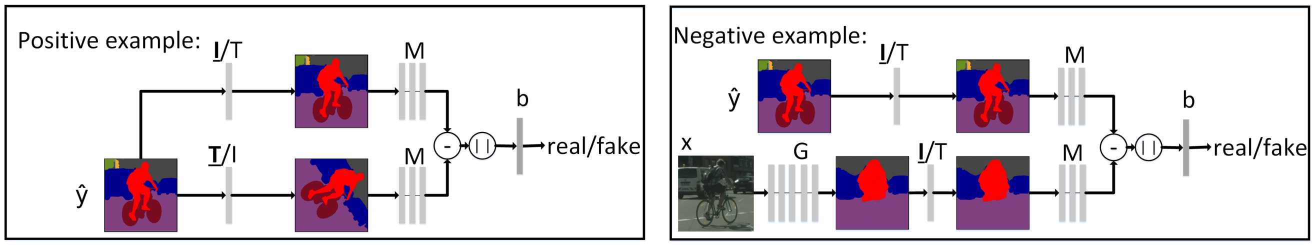


Figure 1: Our MatAN discriminator is a siamese network: (left) positive examples, (right) negative ones. The input to the siamese network is passed through a perturbation T or through the identity transformation I. The configurations of T and I result in different training behavior. The drawing shows the case when the perturbation is only applied to one branch of the positive samples.

ity density function of the generated and the ground truth data. This approach has the restriction that the discriminator must be a 1-Lipschitz function, which is achieved by weight trimming. This disadvantage is addressed in [11] by penalizing the norm of the gradients of the discriminator. Other works improving and analyzing GANs include [25, 24, 21, 3, 16, 14]. In [7] the generator outputs an image pair of the same person and the siamese discriminator net predicts if the image pair is real or generated. In contrast, our siamese discriminator can take as input a real-generated pair and functions as a supervised loss.

The idea of using a discriminator as loss function has been also used in the context of supervised problems, by making both the generator and the discriminator a function of the input. This is typically refereed as a conditional GAN (CGAN) [22]. It has been applied to many problems, including text to image generation [27, 34, 8], image description [6, 19], super-resolution [18], shadow detection [23], style transfer [36], semi-supervised learning [35, 30], general image-to-image translation [15, 33, 9], learning from simulated images [29]. While CGANs are successful in tasks including image generation, they perform poorly in tasks with well defined metrics, such as semantic segmentation. We are not aware of any previous work producing comparable results to state-of-the-art by using only adversarial networks in the loss function.

3. Matching Adversarial Networks

We start our discussion with a short overview of GANs and conditional GANs. We then formulate a new discriminator consisting of a siamese network. This allow us to make better use of the training examples, resulting in much better performance and more stable training.

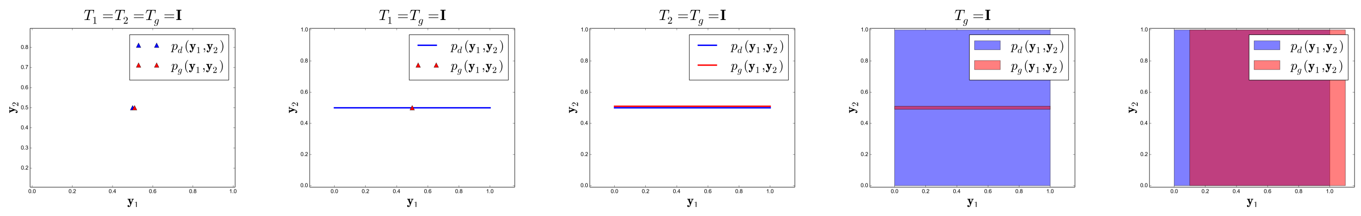
3.1. Overview of GANs and CGANs

Generative Adversarial Networks (GANs) [10] train deep generative models using a minimax game. To generate samples, the generator maps a random noise vector $\mathbf{z}$ into a high dimensional output $\mathbf{y}$ (e.g., an image) via a neural network $\mathbf{y} = G(\mathbf{z}, \Theta_G)$. The generator G is trained to fool a discriminator, $D(\mathbf{y}, \Theta_D)$, which tries to discriminate between fake (i.e., generated) and real samples. The GAN minimax game can thus be written as

$$\min_{\Theta_G} \max_{\Theta_D} \mathcal{L}_{\text{GAN}}(\hat{\mathbf{y}}, \mathbf{z}, \Theta_D, \Theta_G) = \mathbb{E}_{\hat{\mathbf{y}} \sim p_y} \log(D(\hat{\mathbf{y}}, \Theta_D)) + \mathbb{E}_{z \sim p(z)} \log(1 - D(G(\mathbf{z}, \Theta_G), \Theta_D) \quad (1)$$

where the first term sums over the positive examples for the discriminator (i.e., training examples), while the second term sums over the negative examples which are generated by the generator by sampling from the noise prior. Learning in GANs is an iterative process which alternates between optimizing the loss $\mathcal{L}_{\text{GAN}}(\hat{\mathbf{y}}, \mathbf{z}, \Theta_D, \Theta_G)$ w.r.t. to the discriminator parameters Θ_D and the generator parameters Θ_G respectively. The discriminator estimates the ratio of the data distribution $p_d(\mathbf{y})$ and the generated distribution $p_g(\mathbf{y})$: $D_G^*(\mathbf{y}) = \frac{p_d(\mathbf{y})}{p_d(\mathbf{y}) + p_g(\mathbf{y})}$. As shown in [10], the global minimum of the training criterion (equilibrium) is where the two probability distributions are identical: $p_g = p_d, D_G^*(\mathbf{y}) = 1/2$. Note that the gradients w.r.t. to Θ_G do not depend on $\hat{\mathbf{y}}$ directly, but only implicitly through the current estimate of Θ_D. As a consequence, the generator can produce any samples from the data distribution instead of learning input-output relations in a supervised fashion.

To overcome this, GANs can be easily extended to conditional GANs (CGANs) by introducing dependency of the generator and the discriminator on the input $\mathbf{x}$. So the discriminator for the positive samples becomes $D(\mathbf{x}, \hat{\mathbf{y}}, \Theta_D)$, while for the negative ones it is $D(\mathbf{x}, G(\mathbf{x}, \Theta_G, \mathbf{z}), \Theta_D)$. Since $D(\mathbf{x}, G(\mathbf{x}, \mathbf{z}, \Theta_G), \Theta_D)$ does not depend on the training targets, practitioners [15] add an additional discriminative loss function to the objective, e.g. a pixel-wise ℓ_1 norm. Unfortunately, it is very difficult to balance the influence of the adversarial and task losses, and a simple linear combination does not work well in practice. Adding an adversarial loss to a task-specific one does not necessarily improve the performance [15].



(a): α (b): β (c): γ (d): ϵ (e): ζ

Figure 2: The joint probability distributions of the siamese inputs in case of a 1D toy problem. α is a trivial case with equilibrium in $G(\mathbf{x}) = \hat{\mathbf{y}}$, but this did not work in practice. In β the equilibrium $p_d = p_g$ is not achievable, but the network will converge towards $G(\mathbf{x}) = \hat{\mathbf{y}}$. γ can achieve equilibrium if $G(\mathbf{x}) = \hat{\mathbf{y}}$. δ (when $T_1 = \mathbf{I}$) is the transposed of γ which can achieve equilibrium if $G(\mathbf{x}) \in T_2(\hat{\mathbf{y}})$. ϵ cannot achieve equilibrium, nor is it converging. ζ can achieve equilibrium if $G(\mathbf{x}) \in T_2(\hat{\mathbf{y}})$ (like δ).

3.2. Matching Adversarial Networks (MatANs)

We propose to use a siamese architecture for our discriminator, allowing us to exploit the training points explicitly in our loss function. The branches of our siamese network $\mathbf{y}_1, \mathbf{y}_2$ takes as input either perturbations (random transformations) of the ground truth $\mathbf{y}_i = T_i(\hat{\mathbf{y}})$ or the generated output $\mathbf{y}_2 = T_g(G(\mathbf{x}))$. Depending on the configuration of the perturbations, which we denote as t, the perturbation can be set to identity transformation $T_i() = \mathbf{I}()$. We refer the reader to Fig. 1 for an illustration of our siamese discriminator architecture. We refer the reader to Fig. 3 for an example of the perturbations we employ for a semantic segmentation task. Each branch of the siamese network undergoes a complex multi-layer non-linear transformation with parameters Θ_M mapping the input $\mathbf{y}_i$ to a feature space $m(\mathbf{y}, \Theta_M)$. Then $\mathbf{d}$ is calculated as an element-wise absolute value (i.e., *abs*) applied to the difference of the two feature vectors $m()$ coming from the two branches.

$$\mathbf{d}(\mathbf{y}_1, \mathbf{y}_2, \Theta_M) = abs(m(\mathbf{y}_1, \Theta_M) - m(\mathbf{y}_2, \Theta_M)) \quad (2)$$

Based on the negative mean of the $\mathbf{d}$ vector, the discriminator predicts whether a sample pair is fake or real. This is a linear transformation followed by a sigmoid function

$$D(\mathbf{y}_1, \mathbf{y}_2, b, \Theta_M) = \sigma(-\sum_{i}^{K} d_i(\mathbf{y}_1, \mathbf{y}_2, \Theta_M)/K + b) \quad (3)$$

where b is a trained bias and K is the number of features. This formulation ensures that the magnitude of $\mathbf{d}$ should be small for positive examples and large for the negative (i.e., generated) pairs. Similarly to GANs, we train our network as a minimax game with the objective:

$$\min_{\Theta_G} \max_{\Theta_M, b} \mathcal{L}_{\text{MatAN}}(\hat{\mathbf{y}}, \mathbf{x}, \Theta_M, \Theta_G) = \\ \mathbb{E}_{\mathbf{y}_1, \mathbf{y}_2 \sim p_{data}(\mathbf{x}, \mathbf{y}, t)} \log D(T_1(\hat{\mathbf{y}}), T_2(\hat{\mathbf{y}}), \Theta_M, b) + \\ \mathbb{E}_{\mathbf{y}_1, \mathbf{x} \sim p_{data}(\mathbf{x}, \mathbf{y}, t)} \log(1 - D(T_1(\hat{\mathbf{y}}), T_g(G(\mathbf{x}, \Theta_G)), \Theta_M, b)) \quad (4)$$

We do not require a noise since we do deterministic predictions. We optimize by alternating between updating the discriminator and the generator parameters and apply the modified generator loss:

$$\mathcal{L}_{\text{MatAN},G} = -\log D(T_1(\hat{\mathbf{y}}), T_g(G(\mathbf{x}, \Theta_G)), |\Theta_M, b) \quad (5)$$

This formulation enables the generator to match the ground truth labels, while the discriminator must learn to differentiate between the pairs including the generated output and the pairs not. The perturbations serve the purpose of making the matching of the ground truth (positive samples to the discriminator) non trivial, which would be the case if the input of the siamese branches would be identical, resulting always in $\mathbf{d} = \mathbf{0}$. We show in our analysis that in certain configurations the generator learns the ground truth, despite applying random perturbations on it. To investigate the effect of the perturbations, we analyze the joint probability distribution of the branch inputs. This is the extension of the standard GAN to two variable joint distributions. We apply a simplified model assuming one training sample and a perturbation which transforms the training sample to a uniform distribution. In case of multiple samples the distribution of the GT would consist of multiple points.

The optimal matching discriminator We set the first input of the siamese network always $\mathbf{y}_1 = T_1(\hat{\mathbf{y}})$. The second input is $\mathbf{y}_2 = T_2(\hat{\mathbf{y}})$ for the positive samples and $\mathbf{y}_2 = T_g(G(\mathbf{x}))$ for the negatives. T_1, T_2, T_g might be the identity transformation, depending on the $T_i()$ configuration. For a given t perturbation configuration the discriminator loss function can be written as:

$$\mathcal{L}_{\text{MatAN},D} = \mathbb{E}_{\mathbf{y}_1, \mathbf{y}_2 \sim p_d(\mathbf{y}_1, \mathbf{y}_2)} \log(D(\mathbf{y}_1, \mathbf{y}_2) + \\ \mathbb{E}_{\mathbf{y}_1, \mathbf{y}_2 \sim p_g(\mathbf{y}_1, \mathbf{y}_2)} \log(1 - D(\mathbf{y}_1, \mathbf{y}_2)) \quad (6)$$

where $p_d()$ is the joint distribution of $T_1(\hat{\mathbf{y}})$, and $T_2(\hat{\mathbf{y}})$ and $p_g()$ is the joint distribution of $T_1(\hat{\mathbf{y}})$ and $T_g(G(\mathbf{x}))$. Following [10], the optimal value of the discriminator for a

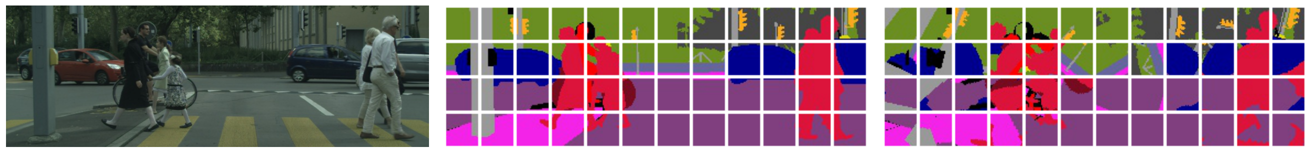

(a): Input (b) GT (c) Perturbations

Figure 3: Perturbations: (a) Cityscapes input image, (b) the corresponding ground truth divided in patches, (c) rotation perturbations applied independently patch wise on the ground truth.

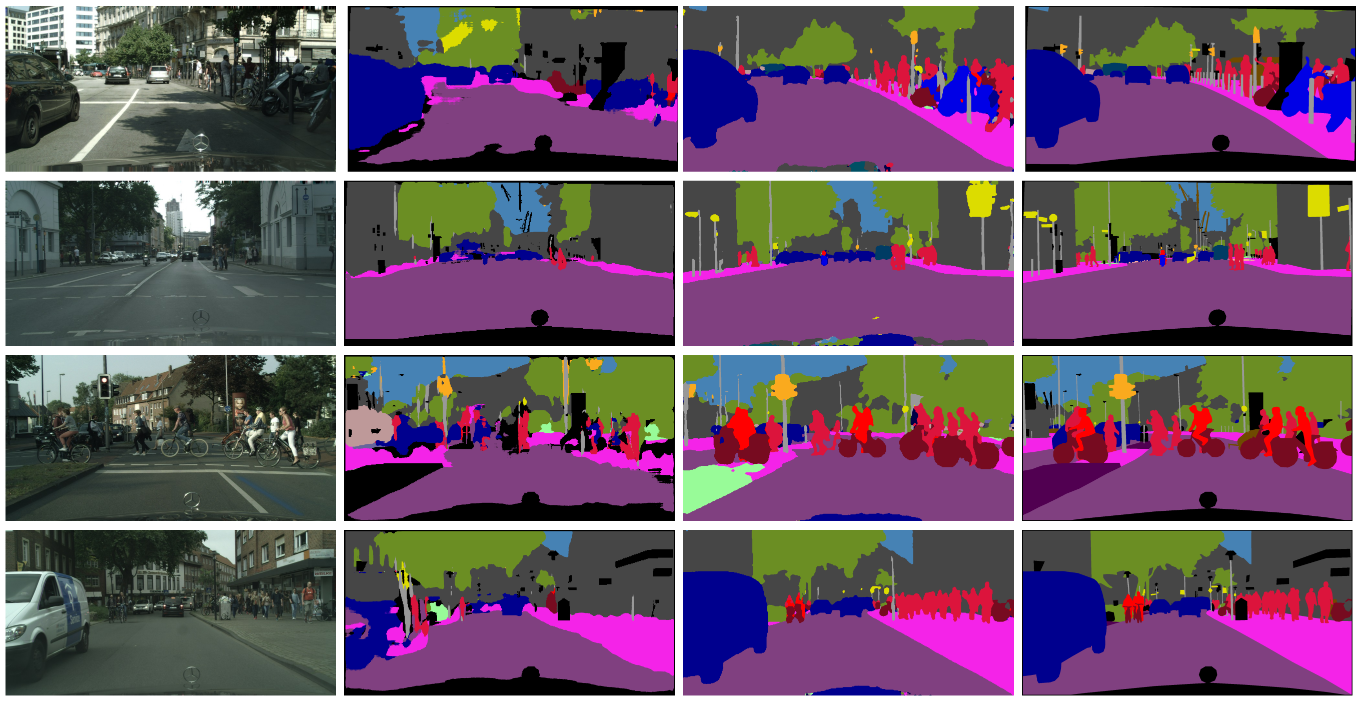

(a): Input (b) Pix2Pix CGAN [15] (c) MatAN β MS (ours) (d) GT

Figure 4: Segmentation results on Cityscapes: Pix2Pix (b) captures the larger objects with homogeneous texture, but it hallucinates objects in the image. In contrast, our method (c) can produce results very similar to the ground truth.

fixed G is:

$$D^*(\mathbf{y}_1, \mathbf{y}_2) = \frac{p_d(\mathbf{y}_1, \mathbf{y}_2)}{p_d(\mathbf{y}_1, \mathbf{y}_2) + p_g(\mathbf{y}_1, \mathbf{y}_2)} \tag{7}$$

Similarly to [10], the equilibrium of the adversarial training would be when $D = 1/2$, $p_d = p_g$ the GT and the generated data distributions match. The equilibrium conditions depend on which non-identity perturbations are applied. See Fig. 2 for illustrations. The equilibrium for each:

α: $T_1() = T_2() = T_g() = \mathbf{I}()$: Equilibrium can be achieved if $\hat{\mathbf{y}} = G(\mathbf{x})$, however in practice this did not work since $\mathbf{d}(\hat{\mathbf{y}}, \hat{\mathbf{y}}) = \mathbf{0}$ regardless of $m()$.

β: $T_1() = T_g() = \mathbf{I}()$: Only $T_2(\hat{\mathbf{y}})$ perturbation is applied. Here $p_g(\mathbf{y}_1, \mathbf{y}_2)$ is approximately a Dirac-delta, thus $p_g(\hat{\mathbf{y}}, G(\mathbf{x})) \gg p_d(\hat{\mathbf{y}}, T_2(\hat{\mathbf{y}}))$ always, which implies that the equilibrium of $D = 1/2$ is not achievable. However, since d is the output of a Siamese network $d(G(\mathbf{x}), \hat{\mathbf{y}}) = 0$, if $G(\mathbf{x}) = \hat{\mathbf{y}}$. Since D is a monotonically decreasing function of $d(G(\mathbf{x}), \hat{\mathbf{y}})$ and $d \geq 0$, the maximum is at $G(\mathbf{x}) = \hat{\mathbf{y}}$. So the discriminator values for the generator after discriminator training will be: $D(\hat{\mathbf{y}}, \hat{\mathbf{y}}) > D^*(\hat{\mathbf{y}}, T(\hat{\mathbf{y}})) > D^*(\hat{\mathbf{y}}, \mathbf{y}), \mathbf{y} \notin T(\hat{\mathbf{y}})$, and so the generator loss has its minimum in $\hat{\mathbf{y}}$.

γ: $T_2() = T_g() = \mathbf{I}()$: Only $T_1(\hat{\mathbf{y}})$ is applied. Equilibrium can be achieved if $G(\mathbf{x}) = \hat{\mathbf{y}}$, since in this case the two joint distributions p_d, p_g match.

δ: $T_1() = \mathbf{I}()$: $T_2(\hat{\mathbf{y}})$ and $T_g()$ are applied. Equilibrium can be achieved if $G(\mathbf{x}) \in T_2(\hat{\mathbf{y}})$, since in this case the two joint distributions p_d, p_g match.

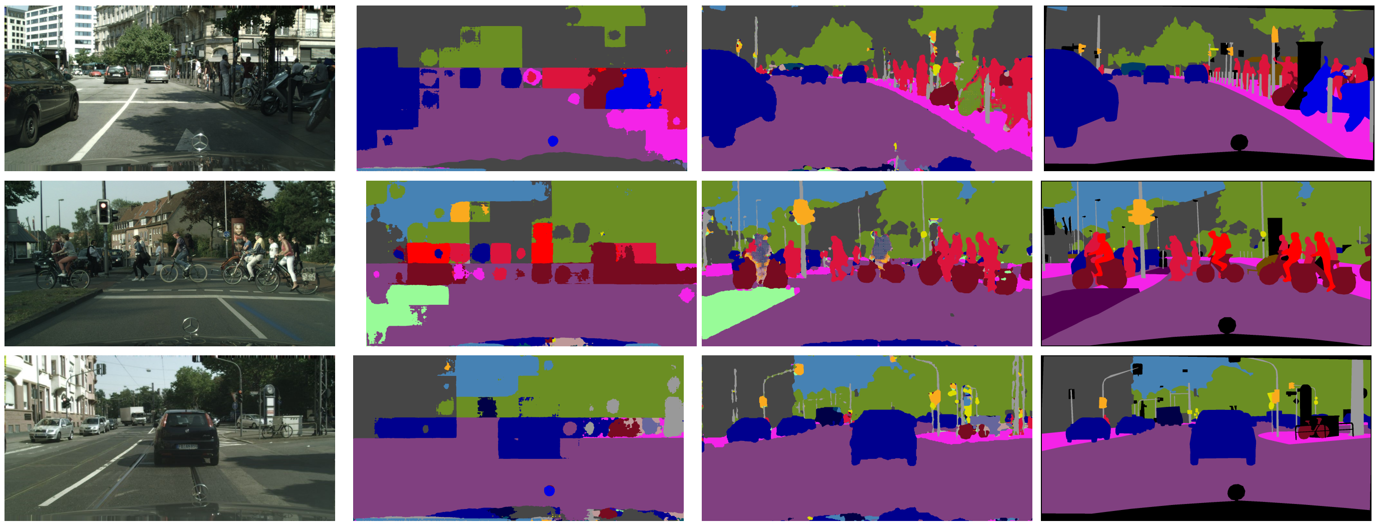


Figure 5: Segmentation results on Cityscapes when trained with perturbed GT. Note that with MatAN the generator learns continuous poles, however these were not observed as training samples. When using cross entropy loss, the net only learns blobs.

ϵ: Only $T_g() = \mathbf{I}()$. Since $p_g(T_1(\hat{\mathbf{y}}), G(\mathbf{x})) \gg p_d(T_1(\hat{\mathbf{y}}), T_2(\hat{\mathbf{y}}))$, there is no equilibrium.

ζ: All perturbations are applied. Equilibrium is achievable if $G(\mathbf{x}) \in T_2(\hat{\mathbf{y}})$, the generator produces any of the perturbations.

We show an ablation study for these configurations on the Cityscapes semantic segmentation task in Table 1. Note, that no additional discriminative loss function is necessary.

Patch-wise discriminator: We divide the output image into small overlapping patches and use each patch as an independent training example. As perturbation we apply random rotations in the range of $[0, 360^o]$ with random flips resulting in an uniform angle distribution. We implement the rotation over a larger patch than the target to avoid boundary effects. Note that, as shown in Fig. 3, the perturbations are applied independently to each patch and thus the discriminator cannot be applied in a convolutional manner.

4. Experiments

We demonstrate the effectiveness of MatAN on three computer vision tasks with structured outputs: semantic segmentation, road centerline detection and instance segmentation. We provide an ablation study on Cityscapes [5] and further experiments on the aerial images of the TorontoCity dataset [31].

ResNet Gen.	mIoU	Pix. Acc
Original Ground Truth:		
Cross Ent.	66.9	94.7
MatAN α NoPer.	6.0	58.1
MatAN β NoAbs.	21.3	77.5
MatAN β	63.3	94.1
MatAN MS β	**66.8**	**94.5**
MatAN γ Match2Per.	63.5	93.3
MatAN δ PertGen.	60.2	93.8
MatAN β MS + Cross Ent.	65.1	94.2
Perturbed Ground Truth:		
Pert. GT	44.8	78.0
Pert. Cross Entropy	42.7	85.1
MatAN ϵ GT Perturb	25.9	82.8
MatAN ζ All Perturb	**58.1**	**93.8**

Table 1: Mean IoU and pixelwise accuracy results from 3 fold cross-validation on the Cityscapes validation set with the ResNet generator. All values are in %. The greek letters denote the perturbation configuration. The multi-scale MatAN achieves almost the same performance as cross-entropy and is 200% higher than CGAN [15]. When we applied the perturbations to the GT, the MatAN could achieve considerably higher results than this noisy GT or cross-entropy.

4.1. Implementation Details

Network architecture: We use the same generator architecture in all experiments, a ResNet-50 [12] based encoder

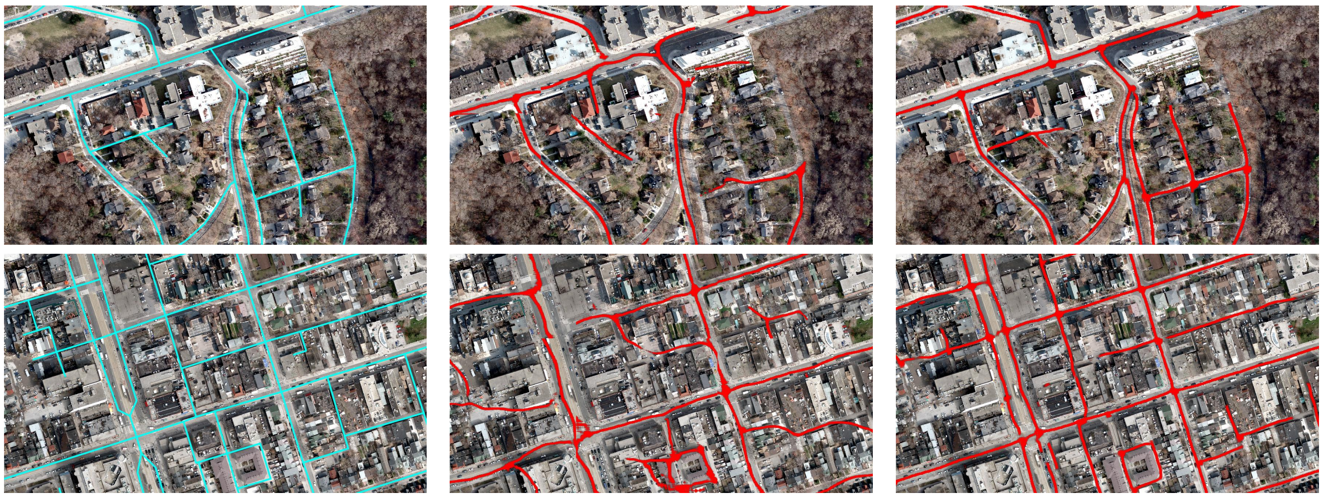

(a): GT (b) CGAN [15] (c) MatAN (ours)

Figure 6: Road centerline extraction: The output of the network is shown in red. Our MatAN can capture the topology even for parallel roads.

Method	Seg.	Validation set				Test set			
		F1	Precision	Recall	CRR	F1	Precision	Recall	CRR
OSM (human) *	-	-	-	-	-	89.7	**93.7**	86.0	85.4
DeepRoadMapper [20] *	✓	-	-	-	-	84.0	84.5	83.4	77.8
Seg3+thinning	✓	91.7	**96.0**	87.8	87.8	**91.0**	93.6	88.4	**88.0**
HED [32] *	-	-	-	-	-	42.4	27.3	**94.9**	91.2
Seg2+thinning	-	89.7	94.9	85.1	82.5	88.4	92.7	84.5	78.0
CGAN	-	75.7	76.4	74.9	75.1	77.0	67.65	89.7	81.8
CGAN + L1	-	78.5	95.1	66.8	68.9	68.6	93.3	54.3	55.0
MatAN (ours)	-	**92.5**	95.7	**89.5**	**88.1**	90.4	91.4	89.5	87.1

Table 2: Road topology recovery metrics in %. The Seg. column indicates if the method uses extra semantic segmentation labeling (background, road, building). * indicates that the results are from [20, 32].

and a decoder containing transposed convolutions for upsampling as well as identity ResNet blocks [13] as non-linearity. The output of the net has always half the size of the input. As default we use 32×32 input size for the discriminator in all experiments with 50% overlap of the patches. For CityScapes we report results also with a multi-scale discriminator. In the discriminator we apply ResNets without batch norm.

Learning details: We use the Adam optimizer [17] with 10^{-4} learning rate, weight decay of $2 \cdot 10^{-4}$ and batch size of 4. We use dropout with 0.9 keep probability in the generator and to the feature vector $\mathbf{d}$ of the discriminator. We train the networks until convergence, which generally requires on the order of 10K iterations. Each iteration (generator and discriminator update) takes 4 seconds on an NVIDIA Tesla P100 GPU. We normalize the output to $[-1, 1]$ by a $\tanh$ if the output image has a single channel (e.g. road centerline) or by a rescaled softmax in case of segmentation.

4.2. Semantic segmentation on Cityscapes

Pixel-wise cross-entropy is well aligned with the pixel-wise intersection over union (IoU) metric and is used as the task loss for state-of-the-art semantic segmentation networks. As we show, our MatAN loss can achieve almost the same performance. We do an ablation study where the generator architecture is fixed (i.e. the ResNet based encoder-decoder), but the discriminator function is changed. We downsample the input image to 1024×512. We randomly split the official validation set to half-half and use one half for early stopping of the training and compute the results on the other half. We repeat this three times and report the mean performance.

We test the perturbation configurations listed in Fig. 2 and provide the results in Table 1. When there is no per-

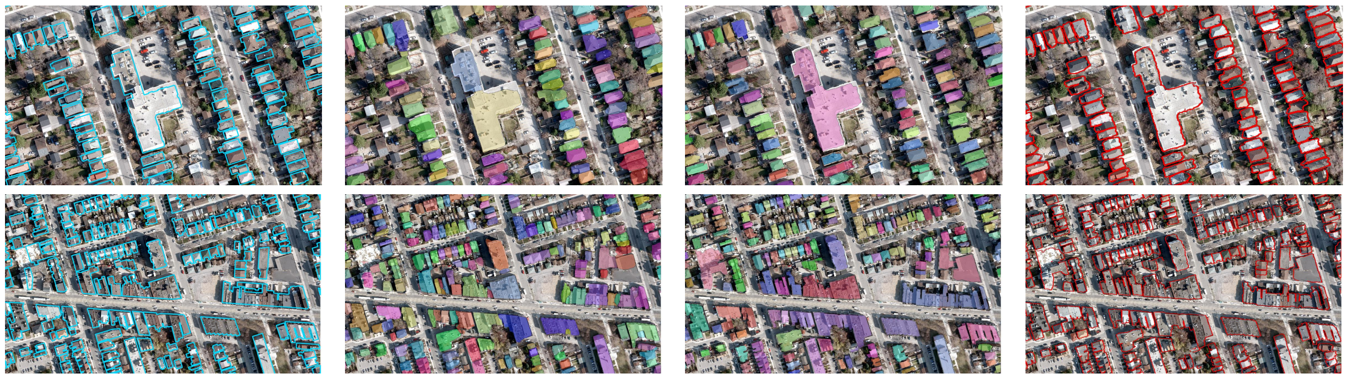

(a): GT (b) DWT [4] (c) MatAN (ours) (d) MatAN contours

Figure 7: (a) Ground truth building polygons overlay over the original image. (b)-(c) final extracted instances, each with a different color. (d) shows the prediction of the deepnet for the building contours which is used to predict the instances. The GT of this task has a small systemic error due to image parallax. In contrast to DWT, MatAN does not overfit on this noise.

turbation (MatAN α NoPer.), the network does not learn. In our experiments the configurations β and γ (the generated output is matched to the GT or the perturbations of the GT) perform similarly. We tested these with a single scale discriminator input size of 32×32. It might be surprising at first that MatAN γ is able to learn, but the joint PDF explains this. Both β and γ can only achieve equilibrium if the GT is generated as output and not a perturbation. Therefore a single discriminator (not patch-wise) would also enable to learn the ground truth. We also apply the discriminator in a multi-scale way by extracting patches on $16, 32$ and 64 scales and resize them to 16. This achieves very close results to cross-entropy and is listed as MatAN β MS. Matan β NoAbs shows that the use of an ℓ_1 distance in (2) for $\mathbf{d}$ is critical. Removing it results in a large performance decrease.

Combining our adversarial loss with the cross entropy loss performs slightly worse than using each loss separately. This shows that fusing loss functions is not trivial.

In configuration δ PertGen. the generated output is perturbed, therefore equilibrium can be achieved in any of the perturbations of the ground truth. Here overlap was not applied for the discriminator patches. The results show that the network still learns the original ground truth (instead of a perturbed one). This can be explained by the patchwise discriminator. An output satisfying all the the discriminator is probably very close to the original ground truth. A deterministic network will rather output a straight line/boundary on an image edge than randomly rotated versions where the cut has to align with the patch boundary.

Applying perturbations to both branches of the true samples can be considered as a noisy ground truth, e.g. two labelers provide different output for similar image regions. The perturbations simulate this with a known distribution of the noise. The entry Pert. GT shows the mIoU of a perturbed GT compared to the original one. When Cross Entropy is trained with this noisy labels (Pert. Cross Entropy), the network looses the fine details and performs around the same as the provided GT, see Fig. 5. The results with perturbed GT are in Table 1. If the generated output is not perturbed, equilibrium cannot be achieved, resulting in low performance (MatAN ϵ GT Perturb). If the output is also perturbed (MatAN ζ ALL Perturb), equilibrium is possible in any of the perturbed GT. Since perturbations are rotations applied patch-wise, a consistent solution for the entire image will be similar to the GT. Note that in this case the generator is trained to infer a consistent solution. For example, a continuous pole was predicted (as seen in Fig 5 (c)), yet it almost never occurs in the perturbed training images.

U-Net Gen. [15]	mIoU	Pix. Acc
Cross Ent.	50.9	91.8
Pix2Pix CGAN [15]	21.5	73.1
Pix2Pix CGAN [15] *	22.0	74.0
Pix2Pix CGAN+L1 [15] *	29.0	83.0
CycGAN [36] *	16.0	58.0
MatAN β MS	**48.9**	**91.4**
MatAN β Pix2Pix arch. MS	**48.4**	**91.5**

Table 3: Mean IoU and pixelwise accuracy results from 3 fold cross-validation on the Cityscapes validation set with the U-Net generator of [15]. All values are in %. * marks the results reported from other papers on the validation set. Our MatAN achieves much higher performance.

Additionally, we show a comparison to the Pix2Pix CGAN [15] by replacing our generator with the U-net architecture of Pix2Pix. To show that the performance increase is not simply caused by the ResNet blocks, we change our discriminator design to match the Pix2Pix discriminator. As shown in Table 3, this achieves lower mIoU values, but still doubles the performance of Pix2Pix. Moreover, this is close to the performance achieved by training the generator using the cross-entropy loss. This indicates that the stability of the learned loss function is not sensitive to the choice of generator architecture, and that the decrease in performance relative to ResNet-based models is due to the reduced capability of the U-net architecture. We implemented the Pix2Pix CGAN method [15] and applied it without the additional task loss. This is far behind MatAN. This can only learn the large objects which appear with relatively homogeneous texture, e.g. road, sky, vegetation, building. As also reported in [15], this method "hallucinates" objects into the image. We see this as a sign that the input-output relation is not captured properly with CGANs using no task loss. Additionally, we report the numbers from the original Pix2Pix [15] paper, showing that even by adding L1 CGAN is outperformed by MatAN. CycleGAN [36] scores even lower than Pix2Pix.

4.3. Road centerline extraction

Roads are represented by their centerlines as vectors in the map. We perform experiments on the aerial images of the TorontoCity dataset [31] resized to 20 cm/pixel. We encode the problem as a one channel image generation with $[-1, 1]$ values and rasterize the vector data according to this as 6 pixel wide lines to serve as training samples. At intersections we added circles to avoid the need for generating very sharp edges which is difficult for neural networks. We use the metrics expressing the quality of the road topology introduced in [20].

We compare our method (the β config) to the HED [32] deepnet based edge detector and DeepRoadMapper [20] which first extracts the road centerlines from the segmentation mask of the roads and then reasons about graph connectivity. Additionally, we use semantic segmentation followed by thinning as a baseline with the same generator as in MatAN. In particular, we tested two variants *Seg3+thinning* which exploits extra 3 class labeling (background, road, buildings) for semantic segmentation, and *Seg2+thinning* which exploits two labels instead (background, road). For comparison we also use OpenStreetMap as a human baseline. As last we compare to CGANs [15] using the adversarial loss and also the adversarial loss combined with L1. We have trained our generator architecture with the CGAN loss but it was not generating reasonable outputs even after 15k iterations. This shows that CGANs are sensitive to the network architecture.

As shown in Table 2, the two best results are achieved by our method and *Seg3+thinning*, which exploits additional labels (i.e., semantic segmentation). Without this extra labeling the segmentation based method (Seg2) and [20] falls behind. The Pix2Pix CGAN approach [15] generates road like objects but they are not aligned with the input image resulting in poor results. OSM achieves similar numbers to the best automatic methods, which shows that mapping roads is not an easy task especially since it can be ambiguous what counts as road. We refer the reader to Fig. 6 for qualitative results.

Method	mAP	Pr. @50%	R. @ 50%	WCov.
ResNet *	22.4	44.6	18.0	38.1
FCN *	16.0	35.1	20.3	38.9
DWT [4] *	**43.4**	75.1	**76.8**	**64.4**
MatAN (ours)	42.2	**82.6**	75.9	64.1

Table 4: Instance segmentation results on the TorotonCity validation set with the metrics given in [31]. All the values are in %. WCov. stands for weighted coverage, mAP for mean precision, R. for recall and Pr. for precision. We refer the reader to [31] for more details about the metrics. * marks implementations of others.

4.4. Instance segmentation

We predict the building instances in the TorontoCity dataset [31] on aerial images resized to 20 cm/pixel. We randomly crop, rotate and flip images with size 768×768 pixels and use a batch size of 4. We jointly generate the 3 class semantic segmentation and the instance contours as a binary image ($[-1, 1]$). We train the net as a single MatAN (β config) showing that it can be used as a single loss for a multi task network. We obtain the instances from the connected components as the result of subtracting the skeleton of the contour image from the semantic segmentation. We compare our results with baseline methods reported in [31], as well as the Deep Watershed Transform (DWT) [4] which also predicts the instance boundaries. As shown in Table 4, our method outperforms DWT by 7% in Precision @ 50%, while been similar in all other metrics. We refer the reader to Fig. 7 for visual results.

Limitations Our method is a discriminative model and is not intended to train conditional generative models, e.g. image generation [15], where an input can be mapped to multiple outputs.

5. Conclusion and future work

We have presented an Adversarial Network, which we call Matching Adversarial Network (MatAN). The discriminator is replaced by a siamese network taking also random perturbations of the ground truth as input. As we have shown in our experiments, this significantly outperforms CGANs, achieves similar or even superior results to task specific loss functions and results in stable training. As future work we plan to investigate more applications, different perturbations and the effect of noisy ground truth.

References

[1] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *CoRR*, abs/1701.04862, 2017. 1

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017. 1

[3] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang. Generalization and equilibrium in generative adversarial nets (gans). *CoRR*, abs/1703.00573, 2017. 2

[4] M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. In *CVPR*, 2017. 7, 8

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 5

[6] B. Dai, D. Lin, U. Raquel, and F. Sanja. Towards diverse and natural image descriptions via a conditional gan. In *ICCV*, 2017. 2

[7] C. Donahue, A. Balsubramani, J. McAuley, and Z. C. Lipton. Semantically decomposing the latent spaces of generative adversarial networks. In *ICLR*, 2018. 2

[8] H. Dong, S. Yu, C. Wu, and Y. Guo. Semantic image synthesis via adversarial learning. In *ICCV*, 2017. 2

[9] H.-Y. Fish Tung, A. W. Harley, W. Seto, and K. Fragkiadaki. Adversarial inverse graphics networks: Learning 2d-to-3d lifting and image-to-image translation from unpaired supervision. In *ICCV*, 2017. 2

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*. 2014. 1, 2, 3, 4

[11] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved Training of Wasserstein GANs. *ArXiv e-prints*, Mar. 2017. 2

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 5

[13] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. *CoRR*, abs/1603.05027, 2016. 6

[14] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie. Stacked generative adversarial networks. In *CVPR*, 2017. 2

[15] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 2, 4, 5, 6, 7, 8

[16] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. *ArXiv e-prints*, Oct. 2017. 2

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 6

[18] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 2

[19] X. Liang, Z. Hu, H. Zhang, C. Gan, and E. P. Xing. Recurrent topic-transition gan for visual paragraph generation. In *ICCV*, 2017. 2

[20] G. Mattyus, W. Luo, and R. Urtasun. Deeproadmapper: Extracting road topology from aerial images. In *ICCV*, 2017. 6, 8

[21] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. *CoRR*, abs/1611.02163, 2016. 2

[22] M. Mirza and S. Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. 1, 2

[23] V. Nguyen, T. F. Yago Vicente, M. Zhao, M. Hoai, and D. Samaras. Shadow detection with conditional generative adversarial networks. In *ICCV*, 2017. 2

[24] S. Nowozin, B. Cseke, and R. Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *NIPS*. 2016. 2

[25] B. Poole, A. A. Alemi, J. Sohl-Dickstein, and A. Angelova. Improved generator objectives for gans. *CoRR*, abs/1612.02780, 2016. 2

[26] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015. 1

[27] S. E. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *CoRR*, abs/1605.05396, 2016. 2

[28] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016. 1

[29] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017. 2

[30] N. Souly, C. Spampinato, and M. Shah. Semi supervised semantic segmentation using generative adversarial network. In *ICCV*, 2017. 2

[31] S. Wang, M. Bai, G. Mattyus, H. Chu, W. Luo, B. Yang, J. Liang, J. Cheverie, S. Fidler, and R. Urtasun. Torontocity: Seeing the world with a million eyes. In *ICCV*, 2017. 5, 8

[32] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, 2015. 6, 8

[33] Z. Yi, H. Zhang, P. Tan, and M. Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *ICCV*, 2017. 2

[34] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017. 2

[35] Z. Zheng, L. Zheng, and Y. Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *ICCV*, 2017. 2

[36] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 2, 7, 8