Jointly Optimize Data Augmentation and Network Training: Adversarial Data Augmentation in Human Pose Estimation

Xi Peng*
Rutgers University
xipeng.cs@rutgers.edu

Zhiqiang Tang*
Rutgers University
zt53@cs.rutgers.edu

Fei Yang
Facebook
yangfei@fb.com

Rogerio S. Feris
IBM T.J. Watson Research Center
rsferis@us.ibm.com

Dimitris Metaxas
Rutgers University
dnm@cs.rutgers.edu

Abstract

Random data augmentation is a critical technique to avoid overfitting in training deep models. Yet, data augmentation and network training are often two isolated processes in most settings, yielding to a suboptimal training. Why not jointly optimize the two? We propose adversarial data augmentation to address this limitation. The key idea is to design a generator (e.g. an augmentation network) that competes against a discriminator (e.g. a target network) by generating hard examples online. The generator explores weaknesses of the discriminator, while the discriminator learns from hard augmentations to achieve better performance. A reward/penalty strategy is also proposed for efficient joint training. We investigate human pose estimation and carry out comprehensive ablation studies to validate our method. The results prove that our method can effectively improve state-of-the-art models without additional data effort.

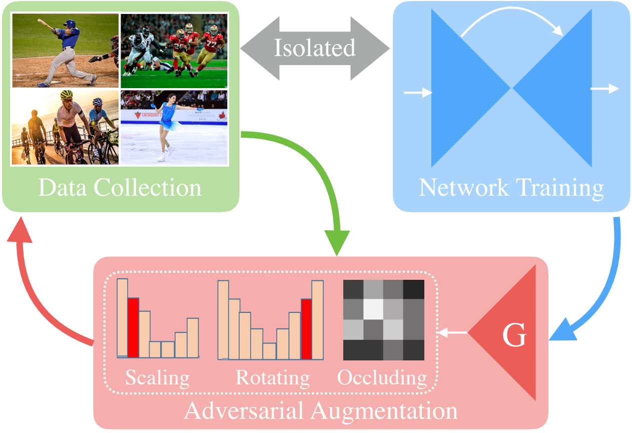

Figure 1: Data preparation and network training are usually isolated. We propose to bridge the two by generating adversarial augmentations online. The generations are conditioned to both training images and the status of the target network.

1. Introduction

Deep Neural Networks (DNNs) have led to significant improvements in many computer vision tasks [20, 10, 18, 9]. In most settings, a two-phase paradigm is usually organized to train deep models as shown in Figure 1: data collection and network training. Yet, this paradigm may not be efficient. First, data collection happens ahead of network training, impelling large-scale data collection to cover possible variations [20]. Second, natural images usually follow long-tail distributions [46, 33]. Effective examples may still be rare even a great number of images have been collected.

A common solution for this problem is to perform random data augmentation [21, 37]. Prior to being fed into the network, training images are heuristically disturbed by predefined transformations (*e.g.* scaling, rotating, occluding) to increase variations. This strategy is simple but the dilemma is still there: data agumentation and network training are isolated, leading to following limitations.

First, static distributions are usually used to sample augmented examples. They can hardly match the network status as the training continues. Intensive data efforts are still needed for large-scale collection and annotation [42, 26]. Second, the same augmentation strategy is usually applied to the entire training set without considering any individual difference. This may produce many ineffective augmentations that are either too hard or too easy to help the network training [32, 39]. Third, the long-tail issue is hardly addressed since random augmentations are usually sampled from Gaussian distributions. There is a small chance to sample rare but effective examples.

*Contributed equally. The project page is public available: https://sites.google.com/site/xipengcshomepage/cvpr2018

A natural question then arises: can data augmentation be conditional to the training status of deep models, so we can make less data effort to obtain more training effect? In other words, can the two be jointly optimized in a unified framework where the convergence of the both is guaranteed?

In this work, we answer the above question by proposing a new approach that leverages adversarial learning for joint optimization. Specifically, we investigate human pose estimation, aiming to improve the network training with bounded datasets. Note that our approach can be easily generalized to other vision tasks, such as face alignment [25] and instance segmentation [23, 13].

Given an off-the-shelf human pose estimation network, our goal is to obtain more training effect from a bounded dataset. Specifically, we propose an augmentation network that acts as a generator. It aims to create adversarial examples that intend to fail the pose network. The pose network, on the other hand, is modeled as a discriminator. It evaluates the quality of generations, and more importantly, tries to learn from the hard examples. The key idea is adversarial examples are created online, conditioned to both input images and the current status of the pose network (see Figure 1 for an illustration). In other words, the augmentation network explores weaknesses of the pose network which, at the same time, learns from hard augmentations for better performance.

Jointly optimizing two networks is not a trivial task. Our experiments indicate that a straightforward design, such as generating adversarial pixels [12, 30] or deformations [39, 18] directly, would yield problematic convergence behaviors (*e.g.* diverging and model collapse). Instead, we design the augmentation network to generate a set of distributions, from which adversarial augmentations (*i.e.* scaling, rotating, occluding) are sampled to create new data points. In addition, a novel reward and penalty policy is proposed to address the missing supervision issue during joint training. Moreover, instead of a raw image, the augmentation network is designed to take the byproduct, *i.e.* intermediate hierarchical features, of the pose network as the input. It can further improve the joint training by leveraging spatial constraints. To summarize, our key contributions are:

- To the best of our knowledge, we are the first to investigate the joint optimization of data augmentation and network training in human pose estimation.

- We propose an augmentation network to play a minimax game against the target network, by generating adversarial augmentations online.

- We take advantage of the wildly used U-net design and propose a reward and penalty mechanism for efficient joint training of the two networks.

- Strong performance on public benchmarks, e.g. MPII and LSP, and intensive ablation studies, validate our method substantially in various aspects.

2. Related Work

We provide a brief overview of works that are most relevant to ours in three categories:adversarial learning, hard example mining and human pose estimation.

Adversarial learning. Generative Adversarial Networks (GANs) [12, 45, 47] includes two networks: generator and discriminator which compete against each other to improve performances. Recent applications of GANs in the human pose estimation include [6] and [7]. They both treat the pose estimation network as the generator and use a discriminator to provide supervisions. However, in our framework, the pose estimation network is a discriminator. The key difference is that we generate adversarial hard examples to improve estimation performances while they add adversarial losses to do that. A-Fast-RCNN [39] uses GANs to generate deformed object region proposals for the object detector.

Hard example mining. The hard example mining usually alternates between optimizing models and updating training data. Once a model is optimized on the current training set, it is used to collect more hard data for further training [38]. It was used in training SVM models for object detection [38, 41]. Recently, Shrivastava *et al.* [32] adapted it into the neural network based object detector. While hard example mining focuses on selecting hard examples from the training set, our method actively generates hard examples to improve network training.

Human pose estimation. With recent advances in Deep Neural Networks (DNNs), image based human pose estimation has achieved significant progress in the past few years [4, 35, 15, 28, 22, 11, 17, 40, 3, 24]. DeepPose [37] is one of the first attempts of using DNNs for human pose estimation. Tompson *et al.* [36] used multiple branches of convolutional networks to fuse the features from an image pyramid, and used Markov Random Field for post-processing. Chen *et al.* [5] also tried to combine the neural networks with the graphical inference.

Recently, sequential human pose prediction becomes popular. Especially, Convolutional Pose Machine [40] brings obvious improvements by cascading multiple networks and adding intermediate supervisions. Better performance is achieved by the stacked hourglasses [24], which also does multi-stage pose estimation. More recent state-of-the-art mostly built on the stacked hourglasses. Each hourglass is able to make inference at multiple scales with the bottom-up and top-down processing. Chu *et al.* [8] added some layers into the stacked hourglasses for the attention modeling. Yang *et al.* [43] also enhanced its performance using the pyramid residual modules in it. In this paper, instead of designing a new network architecture for pose estimation, we introduce an adversarial dynamic data augmentation framework, that can be applied on existing state-of-the-art methods and achieve better performance.

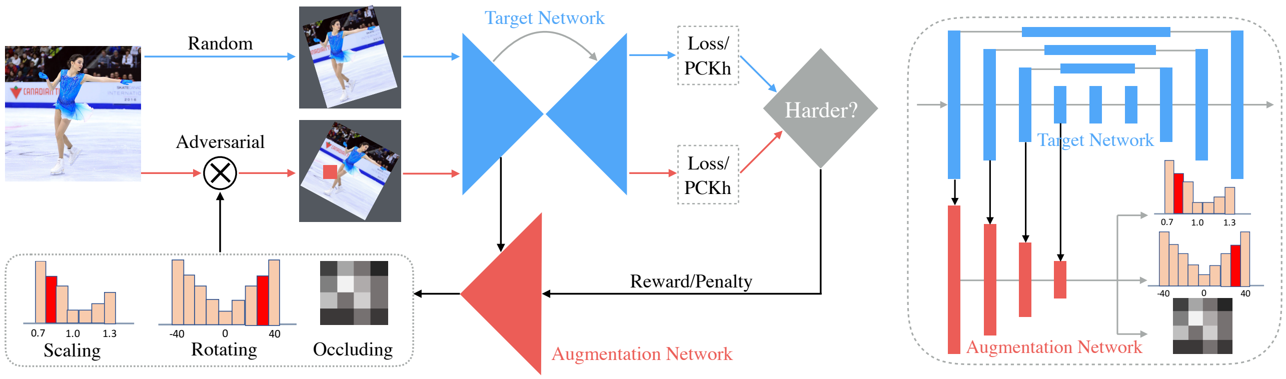

Figure 2: **Left**: Overview of our approach. We propose an augmentation network to help the training of the target network. The former creates hard augmentations; the latter learns from generations and produces reward/penalty for model update. **Right**: Illustration of the augmentation network. Instead of raw images, it takes hierarchical features of an U-net as inputs.

3. Adversarial Data Augmentation

Given a target network, *e.g.* a pre-designed human pose estimator [36], our goal is to improve its training without looking for more data. As we mentioned, the wildly used random data augmentations are suboptimal. They are usually "blindly" sampled from static distributions without considering the training status of the target network. This may produce many ineffective samples that are either too hard or too easy for the target network to learn from.

Instead, we propose to leverage adversarial learning to jointly optimize data augmentation and network training. The key idea is to learn an agent $G(\cdot|\theta_G)$ that generates "hard" augmentations that may increase the target network loss. The target network $D(\cdot|\theta_D)$, on the other hand, tries to learn from the adversarial augmentations and, at the same time, evaluates the quality of the generations. Please refer to Figure 2 for an overview of our approach.

Generation path. The augmentation network is designed as a generator. It outputs a set of distributions of augmentation operations. Mathematically, the augmentation network G outputs adversarial augmentation $\tau_a(\cdot)$ that may increase D's loss, compared with random augmentation $\tau_r(\cdot)$, by maximizing the expectation:

$$\max_{\theta_G} \mathop{\mathbb{E}}_{\mathbf{x}\sim\Omega} \mathop{\mathbb{E}}_{\substack{\tau_r\sim\Gamma \\ \tau_a\sim G(\mathbf{x},\theta_D)}} \mathcal{L}[D(\tau_a(\mathbf{x}),\mathbf{y})] - \mathcal{L}[D(\tau_r(\mathbf{x}),\mathbf{y})], \tag{1}$$

where Ω is the training image set and Γ is the random augmentation space. $\mathcal{L}(\cdot,\cdot)$ is a predefined loss function and $\mathbf{y}$ is the image annotation. We highlight $G(\mathbf{x},\theta_D)$ to specify that the generation of G is conditioned to both the input image $\mathbf{x}$ and the current status of the target network D.

Discrimination path. The target network is designed as a discriminator. It plays two roles: 1) D evaluates the generation quality as indicated in Equation (1); 2) D tries to learn from adversarial generations for better performance by minimizing the expectation:

$$\min_{\theta_D} \mathop{\mathbb{E}}_{\mathbf{x}\sim\Omega} \mathop{\mathbb{E}}_{\tau_a\sim G(\mathbf{x},\theta_D)} \mathcal{L}[D(\tau_a(\mathbf{x}),\mathbf{y})], \tag{2}$$

where adversarial augmentation τ_a can better reflect the weakness of D than random augmentation τ_r, resulting in more effective network training.

Joint training. The joint training of G and D is not a trivial task. Augmentation operations are usually not differentiable [39], which stops gradients to flow from D to G in backpropagation. To solve this issue, we propose a reward and penalty policy to efficiently create online ground truth of G. So G can be always updated to follow D's training status. The details will be explained soon in Section 4.3.

It is crucial that G generates distributions instead of direct operations [39] or adversarial pixels [30]. Our experiments indicate that, by sampling from distributions, the generation is more robust to outliners which may produce upside-down augmentations. Thus, there is less chance that D would get trapped in local optimums yielding training crashes. Besides, the reward and penalty can be directly applied on distributions to efficiently update G online.

Comparison with prior methods. There is a sharp difference between our method and recent adversarial human pose estimation [6, 7]. They usually follow a common design that connects a target network (generator) with an additional network (discriminator) to get auxiliary adversarial loss. In contrast, we propose to learn adversarial network (generator) to improve the target network (discriminator), by jointly optimizing data augmentation and network training.

Our method is also different from others that perform online hard example mining [38, 32]. We can create novel samples that might not exist in the dataset, whereas the latter is usually bound by the dataset. An exception is [39] which uses GANs to generate deformations for object detection. However, applying adversarial augmentation for human pose estimation is still an open question without investigation.

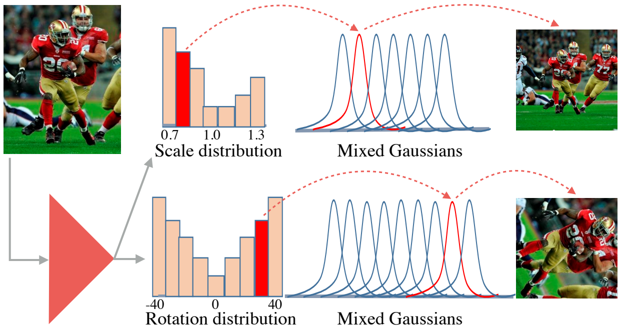

Figure 3: Adversarial scaling and rotating. Our generator predicts distributions of mixed Gaussian, from which scaling and rotating are then sampled to augment the training image.

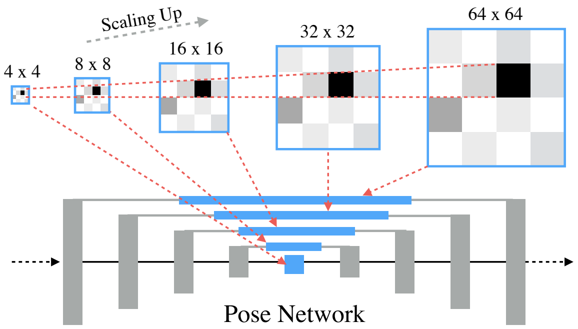

Figure 4: Adversarial Hierarchical Occluding. The occlusion mask is generated at the lowest resolution and then scaled up to apply on hierarchical bridge features of the pose network.

4. Adversarial Human Pose Estimation

Our task is to improve the training of a pre-designed pose network. We take the wildly used U-net design [24, 31] as an example. As illustrated in Figure 2 (right), the augmentation network follows an encoder architecture. It takes the bridged features of the U-net as inputs instead of raw images for efficient training. A set of distributions are then generated to sample three typical augmentations: scaling, rotating, and hierarchical occluding. Furthermore, a reward and penalty strategy is designed for joint training.

4.1. Adversarial Scaling and Rotating

The augmentation network generates hard training samples by scaling and rotating images. The pose network then learns from hard augmentations for better testing performance. In our experiments, we find that a direct generation would collapse the training. It is very easy to generate upside-down augmentations, which is the hardest in most cases. Instead, we divide the augmentation ranges into m and n bins (*e.g.* $m = 7$ for scaling and $n = 9$ for rotating). Each bin is corresponding to a small bounded Gaussian. The augmentation network will first predict distributions over scaling and rotating bins. Then, bounded Gaussians are activated by sampling from distributions. Please refer to Figure 3 for an illustration of the sampling process.

ASR pre-training. It is important to pre-train the augmentation network to obtain a sense of augmentation distributions before the joint training. For every training image, we can sample totally $m \times n$ augmentations, each of which is drawn from a pair of Gaussian. The augmentation is then fed forward into the target network to calculate the loss which represents how "difficult" the augmentation is. We accumulate $m \times n$ losses into the corresponding scaling and rotation bins. By normalizing the sum of bins to 1, we generate two vectors of probabilities: $P^s \in \mathbb{R}^m$ and $P^r \in \mathbb{R}^n$, which approximate the ground truth of scaling and rotation distributions, respectively.

Given the ground-truth distributions P^s and P^r, we propose a KL-divergence loss to pre-train the augmentation network for scaling and rotating:

$$\mathcal{L}_{SR} = \sum_{i=1}^{m} P_i^s \log \frac{P_i^s}{\tilde{P}_i^s} + \sum_{i=1}^{n} P_i^r \log \frac{P_i^r}{\tilde{P}_i^r}, \tag{3}$$

where $\tilde{P}^s \in \mathbb{R}^m$ and $\tilde{P}^r \in \mathbb{R}^n$ are the predicted distributions following the above generation procedure. m and n are the numbers of scale and rotation bins.

Discussions. Predicting distributions instead of direct augmentations has two advantages. First, it introduces uncertainties to avoid upside-down augmentations during the pre-training. Second, it helps to address the issue of missing ground truth during the joint training, which will be explained in Section 4.3. In our design, the scaling and rotating are directly applied on training images instead of deep features [39]. The reason is we want to preserve the location correspondence between image pixels and landmark coordinates. Otherwise, we might hurt the localization accuracy once the intermediate feature maps are disturbed.

4.2. Adversarial Hierarchical Occluding

In addition to scaling and rotating, the augmentation network also generates occluding to make the pose estimation task "harder". Human body has a strong structure where joint locations are highly dependent to each other. By occluding parts of the image, the pose network is encouraged to learn better references among visible and invisible joints.

Different from scaling and rotating, we find that it more effective to occlude deep features instead of image pixels. It does not have the location correspondence issue since joint positions are unchanged after the occluding. Specifically, the augmentation network generates a mask indicating which part of features to be occluded so that the pose network has more estimation errors. We only generate the mask at the lowest resolution of 4×4. The mask is then scaled up to 64×64 to apply on bridge features of the U-net. Figure 4 explains the proposed hierarchical occluding.

```
Algorithm 1: Training scheme of a mini batch
Input: Mini-batch $\mathbf{X}$, augmentation net $G$, pose net $D$.
Output: G, D.
Randomly and equally divide $\mathbf{X}$ into $\mathbf{X}_1$, $\mathbf{X}_2$ and $\mathbf{X}_3$;
Train D using $\mathbf{X}_1$;
Train D, G using $\mathbf{X}_2$ with ASR following Alg. 2;
Train D, G using $\mathbf{X}_3$ with AHO following Alg. 2;
```

AHO pre-training. Similar to scaling and rotating, the augmentation network predicts an occluding distribution instead of an instance occluding mask. The first task is to create the ground truth of the occluding distribution. The idea is to assign values into a grid of $w \times h$ (*e.g.* $w = h = 4$). The value indicates the importance of the features at the corresponding cell. To achieve this, we vote a joint to one of the $w \times h$ cells according to its coordinates. By counting all joints from all images and normalizing the sum of cells to 1, we generate a heat map $P^o \in \mathbb{R}^{w \times h}$, which approximates the ground truth of the occluding distribution.

Given the ground-truth distribution P^o, we propose a KL-divergence loss to pre-train the AHO task:

$$\mathcal{L}_{AHO} = \sum_{i=1}^{h} \sum_{j=1}^{w} P^o_{i,j} \log \frac{P^o_{i,j}}{\tilde{P}^o_{i,j}}, \tag{4}$$

where $\tilde{P}^o \in \mathbb{R}^{w \times h}$ is the heat map predicted by the augmentation network. To generate the occluding mask, we sample one or two cells according to $\tilde{P}^o$, which are labeled as 0 while the rests are labeled as 1.

Discussions. Intuitively, there are three ways to apply hierarchical occluding: (1) a single mask scales up from the lowest to the highest resolutions, (2) a single mask scales down from the highest to the lowest resolutions, and (3) independent masks are generated at different resolutions. We exclusively use the first design in our approach. Since it would occlude more than needed due to the large receptive field in the second case, and the occluded information may be compensated at other resolutions in the third case.

4.3. Joint Training of Two Networks

Once ASR and AHO are pre-trained, we can jointly optimize the augmentation network and the target network. As we mentioned in Sec. 3, this is a nontrivial task since the augmentation ground truth is missing. A naive approach could be repeating the pre-training process as described in Section 4.1 and Section 4.2 online. However, it would be extremely time-consuming since there are a large number of augmentation combinations.

Reward and penalty. Instead, we propose a reward and penalty policy to address this issue. The key idea is, the prediction of augmentation network should be updated according to the current status of the target network, while its quality should be evaluated by comparing with a reference.

```
Algorithm 2: Training scheme of one image.
Input: Image $\mathbf{x}$, augmentation net $G$, pose net $D$.
Output: G, D.
Forward $D$ to get bridge features $\mathbf{f}$;
Forward $G$ with $\mathbf{f}$ to get a distribution $P$;
Sample an adversarial augmentation $\tilde{\mathbf{x}}$ from $P$;
Forward $D$ with $\tilde{\mathbf{x}}$ to compute loss $\tilde{\mathcal{L}}$;
Random augment $\mathbf{x}$ to get $\hat{\mathbf{x}}$;
Forward $D$ with $\hat{\mathbf{x}}$ to compute loss $\hat{\mathcal{L}}$;
Compare $\tilde{\mathcal{L}}$ with $\hat{\mathcal{L}}$ to update $G$ using (3) and (4);
Update $D$;
```

To this end, we sample a pair of augmentations for each image: 1) an adversarial augmentation τ_a and 2) a random augmentation τ_r, as indicated in Equation (1). If the adversarial augmentation is harder than the random one, we reward the augmentation network by increasing the probability of the sampled bin (ASR) or cell (AHO). Otherwise, we penalize it by decreasing the probability accordingly.

Mathematically, let $\tilde{P} \in \mathbb{R}^k$ denotes the predicted distribution of the augmentation network. $P \in \mathbb{R}^k$ denotes the ground truth we are looking for. k is the number of bins (ASR) or cells (AHO) and i is the sampled one.

If the adversarial augmentation τ_a leads to higher pose network loss (more "difficult") comparing with the reference (a random augmentation τ_r), we update P by rewarding:

$$P_i = \tilde{P}_i + \alpha \tilde{P}_i; \quad P_j = \tilde{P}_j - \frac{\alpha \tilde{P}_i}{k-1}, \forall j \neq i. \tag{5}$$

Similarly, if τ_a leads to lower pose network loss (less "difficult") comparing with τ_r, we update P by penalizing:

$$P_i = \tilde{P}_i - \beta \tilde{P}_i; \quad P_j = \tilde{P}_j - \frac{\beta \tilde{P}_i}{k-1}, \forall j \neq i, \tag{6}$$

where $0 < \alpha, \beta \leq 1$ are hyperparameters that controls the amount of reward and penalty. The augmentation network keeps updating online, regardless of being rewarded or penalized, generating adversarial augmentations that intend to improve the pose network.

Discussions. The pose network can learns from the ordinary random augmentation to maintain its regular performance. More importantly, it can also learns from the adversarial augmentations to get better performance. If ASR and AHO are applied on the same image, the adversary may become too hard for the pose network to learn. Thus, we alternately use them on different images. Here we equally split every mini batch into three shares: one is used for the regular training and the other two are used for ASR and AHO, respectively. Please check Algorithm 1 for the details.

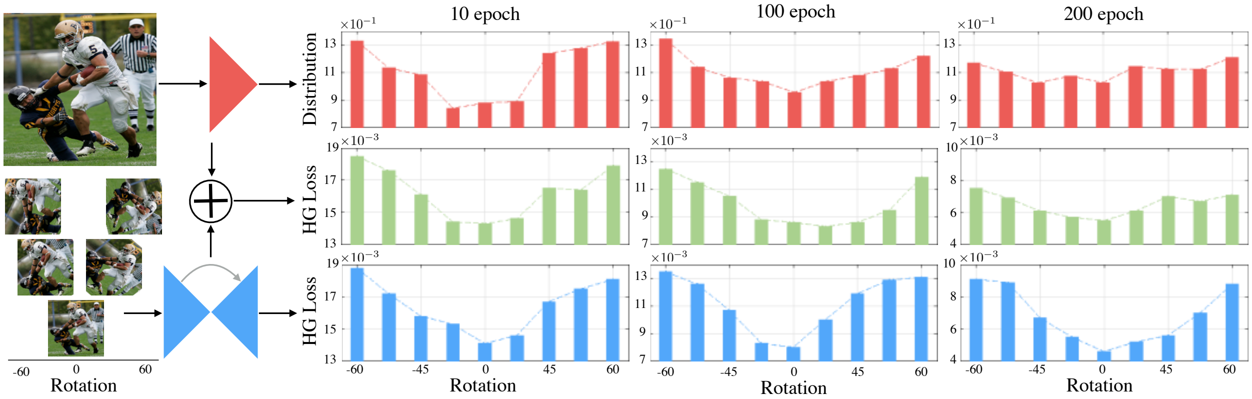

Figure 5: Changes of network states during training: augmentation network (**Top**), adversarial HG (**Middle**), and ordinary HG (**Bottom**). The first two rows show similar shapes which become flatter at last. The last row maintains one shape all the time.

5. Experiments

In this section, we first show the visualization of network training states to verify the motivation of doing adversarial dynamic augmentation. Then we quantitatively evaluate the effectiveness of different components in our method and further compare with state-of-the-art approaches.

5.1. Experiment Settings

In our experiments, we choose the popular hourglass [24] as our target network. Our augmentation network takes the top-down part of a hourglass and only uses one cell module in each resolution block. To evaluate the generalization capability of the proposed adversarial augmentation, we tested two types of modules: *Residual module* [14] and *Dense block* [16]. The dense block is recent approach, that all layers inside a dense block have direct connections. This makes the gradients flow smoother. It is worth mentioning that we are the first to employ dense blocks inside the hourglass architecture.

Network design. We test both residual hourglass and dense hourglass in our component evaluation experiments. For residual hourglass, each residual module has a bottleneck structure of BN-ReLU-Conv(1x1)-BN-ReLU-Conv(3x3)-BN-ReLU-Conv(1x1). The input/output dimension of each bottleneck is 256. The two 1×1 convolutions are used to halve and double the feature dimensions.

For dense hourglass, each module is a bottleneck structure of BN-ReLU-Conv(1x1)-BN-ReLU-Conv(3x3), with neck size 4, growth rate 32, and input dimension 128. The dimension increases by 32 after each dense layer. At the end of each dense block, we use BN-ReLU-Conv(1x1) to reduce the dimension to 128. When comparing with state-of-the-art methods, we use the standard 8 stacked residual hourglasses [24] as our baseline.

Datasets. We evaluate the proposed adversarial human pose estimation on two benchmark datasets: MPII Human Pose [1] and Leeds Sports Pose (LSP) [19]. MPII is collected from YouTube videos with a broad range of human activities. It has 25K images and 40K annotated persons, which are split into a training set of 29K and a test set of 11K. Following [36], 3K samples are chosen from the training set as validation set. Each person has 16 labeled joints.

The LSP dataset contains images from many sport scenes. Its extended version has 11K training samples and 1K testing samples. Each person in LSP has 14 labeled joints. Since there are usually multiple people in one image, we crop around each person and resize it to 256x256. Traditionally, random scaling (0.75-1.25), rotating (-/+30°) and flipping are used to augment the data.

Training. All the networks are implemented in PyTorch, and RMSProp [34] is used to optimize the networks. The adversarial training is divided into three stages. We first train hourglass for a few epochs with a learning rate 2.5×10^{-4}. Then we freeze the hourglass model and use it train the AHO and ASR networks with learning rate 2.5×10^{-4}. Once they are pre-trained, we lower the learning rates of AHO and ASR networks to 5×10^{-5} and jointly train the three networks. The learning rate of hourglass is decayed to 5×10^{-5} after the validation accuracy plateaus. In all experiments, the Percentage of Correct Keypoints (PCK) [44] is used to measure the accuracy.

5.2. Visualization of Training States

In this experiment, we use one residual hourglass and each resolution block has 3 residual modules. Generally, it is difficult to directly visualize the internal state of a neural network. However, in our case, we are interested in knowing how the hourglass handles human images with three variations: rotating, scaling and occluding. Since our method treats these three variations in a similar way, we take rotating as an example. More specifically, we visualize the losses of hourglass on images with different rotations.

Table 1: Comparison of the baseline and our methods on the MPII validation set, using PCKh@0.5 metric.

	Residual hourglass (size: **38M**)								Dense hourglass (size: **18M**)							
	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	Mean	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	Mean
Baseline	97.2	94.8	87.8	83.4	87.8	81.3	76.5	87.0	97.1	94.6	87.9	83.0	87.5	81.2	76.6	86.8
+ASR	97.3	**95.2**	88.2	84.2	88.2	81.8	77.3	87.5	**97.2**	95.0	88.3	83.5	87.7	81.8	77.4	87.3
+AHO	97.3	95.0	88.2	83.6	88.0	82.2	77.6	87.4	97.1	94.8	88.2	83.6	87.6	81.7	77.5	87.2
+ASR+AHO	**97.3**	95.1	**88.7**	**84.7**	**88.4**	**82.5**	**78.1**	**87.8**	**97.2**	**95.2**	**88.8**	**84.1**	**88.1**	**82.0**	**77.9**	**87.6**

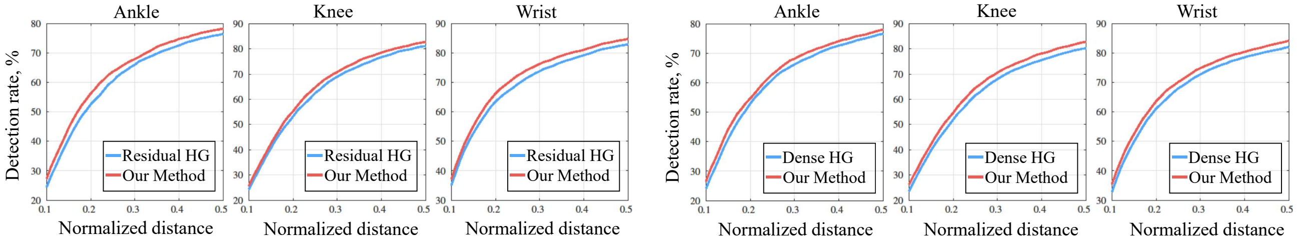

Figure 6: Comparison of one hourglass with and without our adversarial augmentation on MPII validation set. Our method shows consistent improvements over a range of normalized distances, on both residual modules (left) and dense blocks (right).

Ordinary hourglass state. When the hourglass is trained by ordinary rotating augmentation, as indicated in the last row in Figure 5, the shape of its losses over different rotations look like an inverted Gaussian through the training process. This is because ordinary rotated examples are sampled from a zero-centered Gaussian distribution.

Adversarial hourglass state. At first, its loss shape is also like an inverted Gaussian since it is pre-trained by the ordinary rotating augmentation. However, the shape becomes smoother as the training continues. At last, all the loss decreases and the shape becomes flatter, which means that it could better handle examples with different rotations. This benefits from the rotated hard examples dynamically generated from our augmentation network.

Adversarial augmentation state. The state of our augmentation network could be reflected by its predicted rotation distributions. By comparing the first two rows in Figure 5, we find that shapes of predicted distributions are similar to the shapes of losses from the adversarial hourglass. This means that the augmentation network could track the state of hourglass and further generate beneficial examples.

5.3. Component Evaluation

We first verify the effectiveness of our ASR and AHO in both residual hourglass and dense hourglass. In residual hourglass, we use 3 residual bottlenecks in each block. While in dense hourglass, we use 6 densely connected bottlenecks in one dense block. However, the size of dense hourlgass model is still below half of the residual hourlgass. In Table 1, we compare variants of our approach on PCKh@0.5. Figure 6 shows the improvement of our approach compared with baseline, on PCKh threshold from 0.1 to 0.5.

ASR only. Table 1 shows that ASR itself improves the accuracy of all the keypoints on both residual and dense hourglass, with average improvement of 0.5% and 0.5% respectively. This indicates that the generated scaled and rotated examples are beneficial for training the pose networks.

AHO only. Table 1 shows that AHO itself also improves accuracy on both residual and dense hourglass, with average improvement of 0.4% and 0.4% respectively. This demonstrates that hourglass model training also benefits from the occluded examples generated by the AHO.

ASR and AHO. Applying both ASR and AHO further improves the accuracy by 0.4%, compared with applying only one of them. This shows that ASR and AHO are complementary to each one. Figure 6 shows that ASR and AHO significantly boost the accuracy of the most challenging keypoints, e.g. ankle, knee and wrist, on a wide range of normalized distances.

Dense hourglass vs Residual hourglass. Table 1 also shows that residual hourglass and dense hourglass have equivalent accuracy, while the former one is twice bigger (38M vs 18M). This shows a big advantage of the dense blocks, which gives equivalent performance with much fewer parameters. The gradients in the dense hourglass propagate more effectively through the direct connections. Thus, it could learn better with much fewer parameters.

5.4. Comparing with State-of-the-art Methods

Quantitative comparison. To compare with other approaches on the human pose estimation, we train our model on top of the 8 stacked hourglasses in [24]. The bridge features from the first hourglass are used as input of both ASR and AHO networks. The same hierarchical dropout masks are applied to all the hourglasses. Table 2 gives the compari-

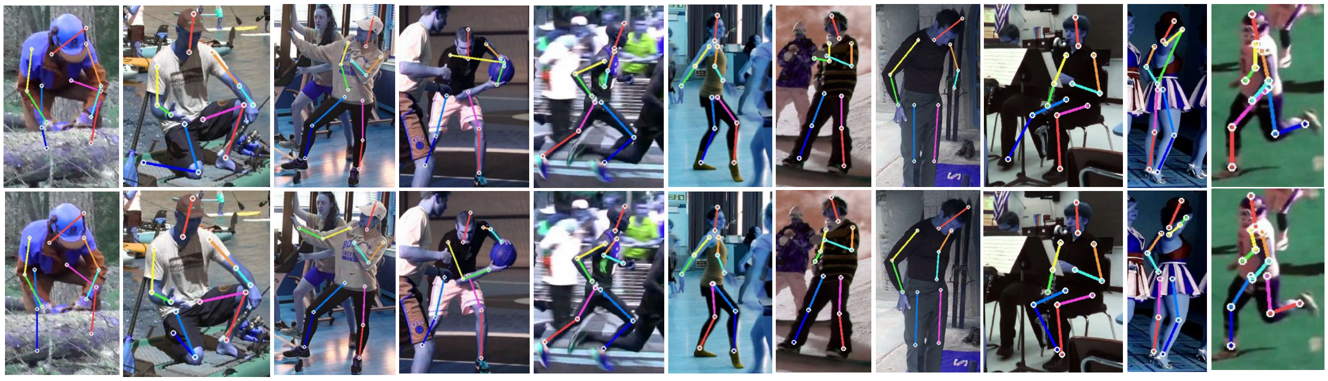

Figure 7: Qualitative comparison between stacked HGs (**top**, totally 8 modules are stacked) and our method (**bottom**). Clear improvements can be spotted on challenging joints (*e.g.* ankle, elbow, wrist), as well as left-right confusion.

son on MPII dataset on PCKh@0.5. Our approach outperforms the baseline [24] by 0.6% and achieves state-of-the-art performance. Table 3 shows the PCKh@0.2 comparison on LSP dataset. Our method achieves state-of-the-art accuracy 94.5%, improving the baseline by 1.5%.

Qualitative Comparison. Figure 7 gives some qualitative comparisons produced by the 8 stacked hourlgasses [24] trained with the ordinary and adversarial augmentations. The figures show where the adversarial augmentation improves the baseline. The pose network becomes more robust to occlusions after adding the adversarial augmentation. Interestingly, the adversarial hourglass could handle the left-right confusions more effectively.

Table 2: PCKh@0.5 on the MPII test set. Our adversarial data augmentation improves baseline stacked HGs(8) [24].

Method	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	Mean
Pishchulin *et al.*[27]	74.3	49.0	40.8	34.1	36.5	34.4	35.2	44.1
Tompson *et al.*[36]	95.8	90.3	80.5	74.3	77.6	69.7	62.8	79.6
Carreira *et al.*[4]	95.7	91.7	81.7	72.4	82.8	73.2	66.4	81.3
Tompson *et al.*[35]	96.1	91.9	83.9	77.8	80.9	72.3	64.8	82.0
Hu *et al.*[15]	95.0	91.6	83.0	76.6	81.9	74.5	69.5	82.4
Pishchulin *et al.*[28]	94.1	90.2	83.4	77.3	82.6	75.7	68.6	82.4
Lifshitz *et al.*[22]	97.8	93.3	85.7	80.4	85.3	76.6	70.2	85.0
Gkioxary *et al.*[11]	96.2	93.1	86.7	82.1	85.2	81.4	74.1	86.1
Rafi *et al.*[29]	97.2	93.9	86.4	81.3	86.8	80.6	73.4	86.3
Belagiannis *et al.*[2]	97.7	95.0	88.2	83.0	87.9	82.6	78.4	88.1
Insafutdinov *et al.*[17]	96.8	95.2	89.3	84.4	88.4	83.4	78.0	88.5
Wei *et al.*[40]	97.8	95.0	88.7	84.0	88.4	82.8	79.4	88.5
Bulat *et al.*[3]	97.9	95.1	89.9	85.3	89.4	85.7	81.7	89.7
Chu *et al.*[8]	**98.5**	96.3	91.9	88.1	90.6	**88.0**	**85.0**	**91.5**
Stacked HGs(8) [24]	98.2	96.3	91.2	87.1	90.1	87.4	83.6	90.9
Ours: +ASR+AHO	98.1	**96.6**	**92.5**	**88.4**	**90.7**	87.7	83.5	**91.5**

6. Conclusion

In the paper, we have proposed the adversarial data augmentation modeled as a reversed generative adversarial learning problem. An effective training scheme with the reward penalty policy is given to jointly optimize the target network and the augmentation network. We apply the adversarial data augmentation to human pose estimation and design the adversarial scaling and rotation as well as the adversarial hierarchical occluding to boost existing pose estimators. Experiments on the benchmarks show clear improvements over the baselines and state-of-the-art performances.

Table 3: PCK@0.2 on the LSP dataset. Clear improvements are observed over the baseline stacked HGs(8) [24].

Method	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	Mean
Belagiannis *et al.*[2]	95.2	89.0	81.5	77.0	83.7	87.0	82.8	85.2
Lifshitz *et al.*[22]	96.8	89.0	82.7	79.1	90.9	86.0	82.5	86.7
Pishchulin *et al.*[28]	97.0	91.0	83.8	78.1	91.0	86.7	82.0	87.1
Insafutdinov *et al.*[17]	97.4	92.7	87.5	84.4	91.5	89.9	87.2	90.1
Wei *et al.*[40]	97.8	92.5	87.0	83.9	91.5	90.8	89.9	90.5
Bulat *et al.*[3]	97.2	92.1	88.1	85.2	92.2	91.4	88.7	90.7
Chu *et al.*[8]	98.1	93.7	89.3	86.9	93.4	94.0	92.5	92.6
Stacked HGs(8) [24]	98.2	94.0	91.2	87.2	93.5	94.5	92.6	93.0
Ours: ASR+AHO	**98.6**	**95.3**	**92.8**	**90.0**	**94.8**	**95.3**	**94.5**	**94.5**

7. Acknowledgment

This work is partly supported by the Air Force Office of Scientific Research (AFOSR) under the Dynamic Data-Driven Application Systems program and NSF CISE.

References

[1] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014.

[2] V. Belagiannis and A. Zisserman. Recurrent human pose estimation. In *FG*, 2017.

[3] A. Bulat and G. Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In *ECCV*, 2016.

[4] J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik. Human pose estimation with iterative error feedback. In *CVPR*, 2016.

[5] X. Chen and A. L. Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. In *NIPS*, 2014.

[6] Y. Chen, C. Shen, X.-S. Wei, L. Liu, and J. Yang. Adversarial posenet: A structure-aware convolutional network for human pose estimation. In *ICCV*, 2017.

[7] C.-J. Chou, J.-T. Chien, and H.-T. Chen. Self adversarial training for human pose estimation. *arXiv*, 2017.

[8] X. Chu, W. Yang, W. Ouyang, C. Ma, A. L. Yuille, and X. Wang. Multi-context attention for human pose estimation. *arXiv*, 2017.

[9] M. Elhoseiny, Y. Zhu, H. Zhang, and A. Elgammal. Link the head to the beak: Zero shot learning from noisy text description at part precision. In *CVPR*, 2017.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[11] G. Gkioxari, A. Toshev, and N. Jaitly. Chained predictions using convolutional neural networks. In *ECCV*, 2016.

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

[13] K. He, G. Gkioxari, P. Dollr, and R. Girshick. Mask R-CNN. In *ICCV*, 2017.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[15] P. Hu and D. Ramanan. Bottom-up and top-down reasoning with hierarchical rectified gaussians. In *CVPR*, 2016.

[16] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. *arXiv*, 2016.

[17] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *ECCV*, 2016.

[18] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NIPS*, 2015.

[19] S. Johnson and M. Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *BMVC*, 2010.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *IEEE*, 1998.

[22] I. Lifshitz, E. Fetaya, and S. Ullman. Human pose estimation using deep consensus voting. In *ECCV*, 2016.

[23] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[24] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.

[25] X. Peng, R. S. Feris, X. Wang, and D. N. Metaxas. A recurrent encoder-decoder network for sequential face alignment. In *ECCV*, 2016.

[26] X. Peng, J. Huang, Q. Hu, S. Zhang, A. Elgammal, and D. Metaxas. From circle to 3-sphere: Head pose estimation by instance parameterization. *CVIU*, 2015.

[27] L. Pishchulin, M. Andriluka, P. Gehler, and B. Schiele. Strong appearance and expressive spatial models for human pose estimation. In *ICCV*, 2013.

[28] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. V. Gehler, and B. Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *CVPR*, 2016.

[29] U. Rafi, B. Leibe, J. Gall, and I. Kostrikov. An efficient convolutional network for human pose estimation. In *BMVC*, 2016.

[30] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *arXiv*, 2016.

[31] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, Lecture Notes in Computer Science, 2015.

[32] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016.

[33] Z. Tang, Y. Zhang, Z. Li, and H. Lu. Face clustering in videos with proportion prior. In *IJCAI*, 2015.

[34] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. In *NNML*, 2012.

[35] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In *CVPR*, 2015.

[36] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, 2014.

[37] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *CVPR*, 2014.

[38] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *IJCV*, 2013.

[39] X. Wang, A. Shrivastava, and A. Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. *arXiv*, 2017.

[40] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016.

[41] L. Wu, K. J. Wu, A. Sim, M. Churchill, J. Y. Choi, A. Stathopoulos, C.-S. Chang, and S. Klasky. Towards real-time detection and tracking of spatio-temporal features: Blob-filaments in fusion plasma. *TBD*, 2016.

[42] L. Wu, I. E. Yen, J. Chen, and R. Yan. Revisiting random binning features: Fast convergence and strong parallelizability. In *KDD*, 2016.

[43] W. Yang, S. Li, W. Ouyang, H. Li, and X. Wang. Learning feature pyramids for human pose estimation. *arXiv*, 2017.

[44] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011.

[45] H. Zhang, V. Sindagi, and V. M. Patel. Image de-raining using a conditional generative adversarial network. *arXiv*, 2017.

[46] X. Zhu, D. Anguelov, and D. Ramanan. Capturing long-tail distributions of object subcategories. In *CVPR*, 2014.

[47] Y. Zhu, M. Elhoseiny, B. Liu, X. Peng, and A. Elgammal. A generative adversarial approach for zero-shot learning from noisy texts. In *CVPR*, 2018.