ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9413822

SYNTHESIZE & LEARN: JOINTLY OPTIMIZING GENERATIVE AND CLASSIFIER NETWORKS FOR IMPROVED DROWSINESS DETECTION

Sandipan Banerjee[1], *Ajjen Joshi*[1], *Ahmed Ghoneim*[1], *Survi Kyal*[1], *and Taniya Mishra*[*2]

[1]Affectiva, USA, [2]SureStart, USA

ABSTRACT

Driving in a state of drowsiness is a major cause of road accidents, resulting in tremendous damage to life and property. Developing robust, automatic, real-time systems that can infer drowsiness states of drivers has the potential of making life-saving impact. However, real-world drowsy driving datasets are unbalanced, due to the sparsity of drowsy driving events. We focus on the problem of alleviating the class imbalance problem by using generative adversarial networks (GAN) to synthesize examples of sparse classes directly in the feature-space. Our GAN-based framework simultaneously generates realistic examples of sparse classes while using the generated samples to improve the performance of a separate drowsiness classifier. We validate this approach in a real-world drowsiness dataset, where we demonstrate a classifier trained using this approach outperforms a stand-alone classifier trained without any GAN-based augmentations.

Index Terms— Drowsiness Detection, Facial Expressions, GAN, Feature Synthesis, Joint Optimization

1. INTRODUCTION

The success of deep learning has been fueled by the availability of large amounts of labeled training data. However, for certain applications, collecting and curating real-world datasets that are of the requisite size and diversity is challenging. Furthermore, due to the sparsity of tail events, maintaining balance of samples across classes of interest can be difficult. In this paper, we focus on alleviating the class imbalance problem by using GANs [1] to synthesize examples of sparse classes in the feature-space.

Consider the problem of driver drowsiness detection. Drowsy driving is dangerous, causing accidents that kill or injure thousands of people every year [2]. Equipping vehicles with systems that can accurately detect driver drowsiness can be potentially life-saving. Robust drowsiness detection methods are dependent on training models on data that are representative of naturalistic drowsy driving behavior. Most technologies deployed in the automotive industry to detect driver fatigue rely on vehicular behavior, such as distance from lane markers on the road or steering behavior [3]. Arguably, a richer source of signal depicting drowsy behavior is the face of the driver. Such a drowsiness detection system was proposed in [4], where a CNN was trained on a dataset of overnight shiftworkers representing real-world drowsy driving behavior. Instead of image pixels, the CNN was trained on 18 facial features (eye closure, yawn, pose, etc.) generated by the Affdex SDK [5], accumulated over a set of frames. We train our models on this severely unbalanced dataset, demonstrating model improvement by augmenting sparse classes using a novel GAN-based joint training scheme.

The problem of unbalanced training classes is also very common in computer vision datasets, especially for face images, as shown in [6]. To augment the sparsely populated classes, researchers have synthesized artificial data that mimic features of the real samples from these classes. For example, the same facial texture can be reposed with 3D models to add more variation in facial pose and shape [6, 7]. GAN-based models have also been successfully deployed to generate hires synthetic face images [8] or edit visual attributes like age [9], lighting and pose [10], gender and expressions [11].

Unlike these works, we do not focus on hallucinating actual pixels of drowsy face images. Instead, we directly generate the n-dimensional feature vectors describing the drowsiness state of a video sample, similar to [4]. To directly synthesize such features from random noise, we define a novel objective function for GAN training that keeps track of the correlation between synthetic and actual samples to preserve naturalness and add diversity, while pushing different drowsy classes apart in feature space. Furthermore, our GAN framework jointly optimizes the final drowsiness classifier, along with the generator and discriminator. That is, signal from the classifier is utilized as an additional objective to improve the quality of the generated samples during training.

We demonstrate the effectiveness of our framework in mitigating bias in the challenging drowsiness dataset from [4], which suffers from severe class imbalance. The GAN based joint training not only boosts classifier performance for the targeted drowsy classes but improves inter-class separation between samples from all the existing classes.

2. AUGMENTING SPARSE TRAINING CLASSES USING GENERATIVE ADVERSARIAL NETWORKS

The dataset introduced in [4] contains real-world data annotated with 3 drowsiness levels: 'Alert', 'Slightly drowsy'

*Work done while at Affectiva

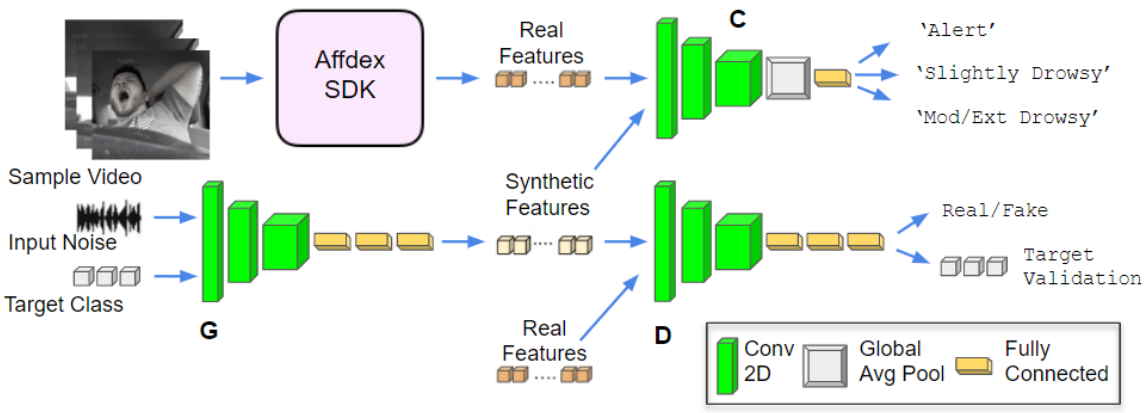

Fig. 1: Our GAN framework: **G** synthesizes features for a target drowsiness class from random noise, while **D** is trained with both synthetic and real drowsy features to predict the naturalness of generated samples and validate their drowsiness class association. The drowsiness classifier model **C** is jointly trained with both real and synthetic features and pushes **G** to generate more authentic samples and in turn improve the classifier performance.

and 'Moderately/Extremely drowsy'. The training dataset is highly unbalanced with plenty of 'Alert' samples, while the 'Slightly drowsy' and 'Moderately/Extremely drowsy' classes are quite sparse. Such an unbalanced training set typically results in a biased classifier model. To alleviate this issue, we propose a novel GAN [1] based augmentation model. A GAN is typically composed of an upsampling generator network for synthesizing artificial samples of a particular domain from random noise and a downsampling discriminator that learns to detect synthetic samples. These antagonistic objectives help the generator learn useful features of the target domain and refine synthesis quality.

However, majority of the research in this domain involves using GANs to generate synthetic images [12, 8], where synthesis quality is either visually inspected or evaluated using pre-tuned inception [13] based metrics like inception score[14] and FID[15]. This is inapplicable in our case as we generate 1800-dimensional feature vectors, in the same way as described in [4], that are neither human readable nor consistent with the expected input of the inception model. As a potential solution, we train our base classifier network (for drowsiness classification) jointly with the GAN, with its training set directly augmented with the generated samples from a particular training iteration. Once training finishes, the classifier is detached and deployed for drowsiness prediction.

2.1. Proposed GAN Architecture

Our framework consists of a generator (G), a discriminator (D) and a classification (C) networks that are trained jointly. When provided with a latent vector and a target drowsiness class, G is trained to generate synthetic drowsiness vectors that fit the distribution of the associated target class. C is trained with real and synthetic samples, jointly with G and D, and is tasked with learning representations for final drowsiness prediction from real-world test samples.

Specifically, G takes as input a 100-dimensional noise vector $\mathbf{z}$, sampled from a gaussian distribution [12], and a 3-dimensional target class vector $\mathbf{c}$ to condition the generated samples towards a particular drowsiness state [16]. To match input dimensions, $\mathbf{c}$ is embedded in a 100×3 space and concatenated channel-wise with $\mathbf{z}$. This concatenated tensor is then passed through a set of 2D convolutional and dense layers with leaky ReLU [17] activation, generating a 1800-dimensional synthetic sample $\mathrm{G}(\mathbf{z}, \mathbf{c})$. This sample is reshaped to 18×100 and passed through D to get predictions based on its realness and target class association. Similar to G, D is constructed with convolutional layers and two dense layers with sigmoid and softmax activations for realness (D_{src}) and class association (D_{cls}) predictions respectively.

Additionally, we train the classification network C with real training samples, augmented using synthetic samples $\mathrm{G}(\mathbf{z}, \mathbf{c})$, and use it to predict the drowsiness class of each input sample. Similar to D, C reshapes the input tensor $\mathbf{x}$ to 18×100 and passes it through convolution layers and then uses global average pooling for dimensionality reduction and softmax activation for drowsiness prediction $\mathrm{C}(\mathbf{x})$. An illustration of the proposed framework can be seen in Fig. 1.

2.2. Loss Function

Adversarial Loss: As done typically[1], we set D's objective to distinguish between real and synthetic samples as:

$$L_D = -\sum_{i=1}^{N} \log(\mathrm{D}_{src}(\mathbf{x}_i)) - \sum_{i=1}^{N} \log(1 - \mathrm{D}_{src}(\mathrm{G}(\mathbf{z}_i, \mathbf{c})))] \tag{1}$$

where $\mathbf{x}_i$ is a real sample from drowsiness class $\mathbf{c}$, $\mathbf{z}_i$ a random latent vector, and N the batch size. G is then trained to synthesize realistic samples using a frozen D's predictions as:

$$L_G = -\sum_{i=1}^{N} \log(\mathrm{D}_{src}(\mathrm{G}(\mathbf{z}_i, \mathbf{c}))) \tag{2}$$

Intuitively, D's weights are leveraged to tune G's hallucinations to match the distribution of real samples from class $\mathbf{c}$ and produce more realistic samples as training progresses.

Classification Loss: To ensure the target class association $\mathbf{c}$ of a synthetic vector $\mathrm{G}(\mathbf{z}, \mathbf{c})$ is preserved, we add a classification loss L_{cls} that uses D's softmax prediction D_{cls} as:

Table 1: Class-wise performance of our drowsiness classifier (C) with and without GAN based joint training. We evaluate model performance in terms of Accuracy and ROC-AUC. Best viewed in color.

Model	Alert	Slightly Drowsy	Mod/Extremely Drowsy	Macro Average
Classifier only (wo/ GAN) [4]	0.747,0.901	0.481,0.772	0.820,0.936	0.683,0.869
Joint training from scratch (proposed)	**0.841**,0.905	0.497,**0.785**	0.813,**0.956**	**0.717**,**0.882**
Joint training from snapshot, unfreeze after 0 epochs (proposed)	0.737,0.945	**0.540**,0.754	**0.849**,0.894	0.709,0.865
Joint training from snapshot, unfreeze after 25 epochs (proposed)	0.762,**0.950**	0.527,0.767	0.829,0.902	0.706,0.873
Joint training from snapshot, unfreeze after 50 epochs (proposed)	0.828,0.901	0.450,0.769	0.835,0.952	0.704,0.874
Joint training from snapshot, unfreeze after 75 epochs (proposed)	0.821,0.900	0.455,0.767	0.839,0.947	0.705,0.871

$$L_{cls} = -\sum_{i=1}^{N}\sum_{j=1}^{3}(c_i)_j \log(\mathrm{D}_{cls}(\mathrm{G}(\mathbf{z}_i, \mathbf{c}))_j) \quad (3)$$

where j denotes the index of association of the target and predicted class labels.

Correlation Loss: To generate more variations in the synthetic vectors, while preserving their "realness", we add an additional correlation loss L_{corr} on top of L_G. After each iteration of generator training, we calculate the Pearson correlation between real samples $\mathbf{x}$ and synthetic samples $\mathrm{G}(\mathbf{z}, \mathbf{c})$ for a particular drowsiness class $\mathbf{c}$. Ideally, the correlation should be as close to 1 as possible, so we calculate L_{corr} as:

$$L_{corr} = 1 - \frac{1}{N}\sum_{i=1}^{N}\frac{Cov(\mathrm{G}(\mathbf{z}_i, \mathbf{c})_i, \mathbf{x}_i)}{\sigma_{\mathrm{G}(\mathbf{z}_i,\mathbf{c})}\sigma_{\mathbf{x}_i}} \quad (4)$$

where *Cov* denotes covariance between two vectors and σ the standard deviation for particular training batch of size N.

Joint Optimization Loss: Taking inspiration from CycleGAN [18], we jointly train C along with G and D using both real and synthetic vectors. However, unlike CycleGAN [18], we formulate C's task not to reconstruct the input but to predict the level of drowsiness from other real feature vectors. After each iteration of GAN training, a set of synthetic vectors for the two non-alert classes are generated by feeding G different noise values and their corresponding target classes. These synthetic vectors are then added to our original drowsiness training set to augment the target sparse classes and balance the training distribution. We train C with the augmented data using a categorical cross entropy loss as shown below:

$$L_{opt} = -\sum_{i=1}^{N}\sum_{j=1}^{3}(y_i)_j \log(\mathrm{C}(\mathbf{x}_i)_j) \quad (5)$$

where N is the batch size for training the classifier, $\mathbf{x}$ a real input sample and y its actual association for the drowsiness class j. This ground truth class association is compared with C's prediction $\mathrm{C}(\mathbf{x}_i)$ for loss computation.

Full Loss: The full training objective L of our framework is calculated as the weighted sum of the different losses:

$$L = L_G + \lambda_1 L_{cls} + \lambda_2 L_{corr} + \lambda_3 L_{opt} \quad (6)$$

We tune the loss coefficients λ_1, λ_2, and λ_3 empirically.

2.3. Training Regimes

Joint training from scratch: In this regimen, we train all three models (generator G, discriminator D and classifier C) from scratch, initializing their weights randomly from a uniform distribution[19]. In this scenario, C learns representations from noisy synthetic vectors generated in the early epochs, before they mature in the later stage of G's training.

Joint training from snapshot: Inspired by [20], we pre-train C solely on real drowsiness vectors and save the snapshot that produces the best validation accuracy. During training of the joint framework, C is initialized from this snapshot while G and D are again initialized randomly. Starting from a frozen set of weights, C can be unfrozen at different points of training depending on the desired maturity of G's features.

3. EXPERIMENTS

Training Details: For training our model we use the shift-worker dataset from[4]. Each sample is a representation of a 10-second input video clip and contains a single drowsiness label. Each frame of the input video clip is processed with the Affdex SDK[5], which outputs a 18-dimensional descriptor, consisting of estimates of head-pose, facial expressions and emotions of the driver. This sequence of descriptors is resampled into a 18×100-dimensional input vector. After pruning samples with missing features, we ended up with 20,126 (Alert: 10,639, Slightly Drowsy: 3,650, Mod/Extremely Drowsy: 5,837), 2,235 (Alert: 1,182, Slightly Drowsy: 405, Mod/Extremely Drowsy: 648) and 7,427 (Alert: 4,016, Slightly Drowsy: 1,247, Mod/Extremely Drowsy: 2,164) samples for training, validation and testing respectively. As evident from these numbers, both the drowsy classes are sparsely populated compared to the alert class in all three datasets. We aim to mitigate this bias by synthetically augmenting these classes during training.

We train our classifier (C), generator (G) and discriminator (D) for 100 epochs with a batch size of 16 samples using the Adam optimizer[22] with a base learning rate of 10^{-4}. The learning rate for C is decayed gracefully using cosine scheduling to prevent overfitting while it remains constant for G and D. We apply Dropout[23] and label smoothing[14] during training as regularization. For training the jointly optimized model (Section 2.1), we empirically set λ_1, λ_2 and λ_3 as 1, 1 and 10 respectively. We build all models using Tensorflow [24] and Keras [25] and train them on a NVIDIA Tesla V100 card. The classifier snapshot that generates the best validation accuracy during training is used for evaluation.

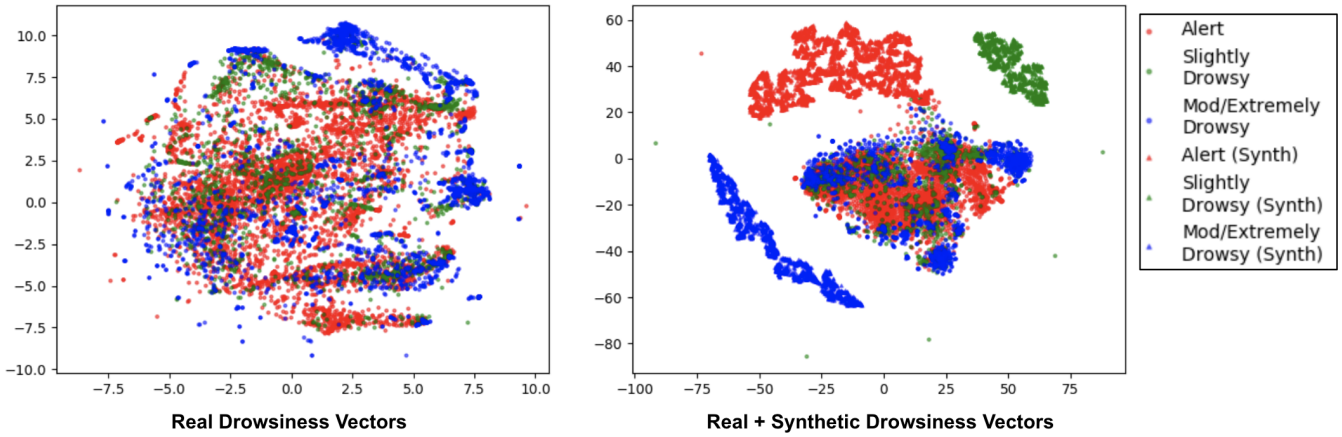

Fig. 2: tSNE visualization: As can be seen, the drowsiness classes are deeply entangled (left) but synthetic vectors stretch the manifold pushing real samples further away from each other (right), similar to [21]. Consequently, jointly training the classifier network with the GAN improves its performance.

Results: To evaluate classifier performance during testing, we use the accuracy and ROC-AUC metrics, similar to [4]. We gauge the effectiveness of the synthetic features generated by G by comparing the jointly optimized C with the stand-alone classifier (*i.e.* C trained without GAN[4]). If the samples synthesized by G are realistic, they should mitigate the dataset bias during joint training and tune C's representations to better classify test samples from all three classes. Along with initializing C from scratch for joint training, we also try fine-tuning it from a pre-trained snapshot, as mentioned in Section 2.3. To estimate the optimal unfreezing point of C's weights, we try fine-tuning it after 0, 25, 50 and 75 epochs. The results can be seen in Table 1.

As expected, training the classifier with GAN based augmentation improves its performance for both the drowsy classes when compared with stand-alone training (*i.e.* without GAN). Surprisingly, the joint training also improves C's performance on the Alert class, suggesting well separated boundaries between all three classes are formed when synthetic samples are added during training. The synthetic samples can stretch the feature manifold[21] and push real samples from different classes further away from each other, as visualized in Figure 2, improving inter-class separation and consequently model performance.

Additionally, we find joint training the classifier C from scratch to be slightly better than initializing it from a pre-trained snapshot. When trained from scratch, C first learns representations from noisy samples generated by G during the initial training epochs, which further regularizes its weights[26, 27]. Then as training progresses, D pushes G to improve the quality of the generated samples, helping C to better learn the authentic representation of the drowsy classes. However, when fine-tuning from a snapshot, C's weights are already tuned to separate authentic samples from the three classes. Therefore, when unfreezing early (*i.e.* after 0 or 25 epochs), the noisy synthetic samples from the early iterations actually deteriorate its performance on the drowsy classes. When unfreezing late (*i.e.* after 75 epochs), C does learn better features from the matured samples but does not experience the full variance produced by G through the whole 100 epochs. Thus unfreezing midway (50 epochs) can be a good option, if not training from scratch.

Table 2: Ablation study results: we ablate each loss component of the model to evaluate their contribution.

Model	Macro Avg. Accuracy	Macro Avg. ROC-AUC
wo/ L_G	0.676	0.861
wo/ L_{cls}	0.680	0.860
wo/ L_{corr}	0.681	0.863
wo/ L_{opt}	0.673	0.860
Full model	**0.717**	**0.882**

Ablation Study: To further analyze the contribution of each component, we ablate them individually from our GAN framework. The results, as shown in Table 2, suggest the adversarial (L_G) and joint optimization (L_{opt}) losses to be key in generating realistic synthetic samples while the classification (L_{cls}) and correlation (L_{corr}) losses act as regularizers. The overall best performance is, of course, achieved by the full model with all the losses combined.

4. CONCLUSION

In this paper, we focused on the problem of building a driver drowsiness detection system. Building such systems are challenging, in part due to real-world drowsy driving datasets being unbalanced. We alleviated the class imbalance problem by introducing a GAN based joint training approach to synthesize examples of sparse classes directly in the feature-space. Our GAN-based framework simultaneously generates realistic examples of sparse drowsy classes while using the generated samples to improve the performance of a separate drowsiness classifier. We demonstrated the effectiveness of our approach on a challenging drowsiness dataset [4], which suffers from severe class imbalance. The GAN based joint training not only boosts classifier performance for the sparse classes (Table 1) but improves inter-class separation between all three classes (Fig. 2). As future work, we plan to extend our framework to address dataset imbalance in other domains.

5. REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NeurIPS*, 2014.

[2] "Drowsy driving nhtsa reports, 2015," *Available here:* https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812446.

[3] M.Q. Khan and S. Lee, "A comprehensive survey of driving monitoring and assistance systems," *Sensors*, vol. 19, no. 11, pp. 2574, 2019.

[4] A. Joshi, S. Kyal, S. Banerjee, and T. Mishra, "In-the-wild drowsiness detection from facial expressions," in *Human Sensing and Intelligent Mobility Workshop, IEEE Intelligent Vehicles Symposium*, 2020.

[5] D. McDuff, A. Mahmoud, M. Mavadati, M. Amr, J. Turcot, and R. el Kaliouby, "Affdex sdk: A cross-platform real-time multi-face expression recognition toolkit," in *CHI Extended Abstracts*, 2016.

[6] I. Masi, A.T. Tran, T. Hassner, J.T. Leksut, and G. Medioni, "Do we really need to collect millions of faces for effective face recognition?," in *ECCV*, 2016.

[7] S. Banerjee, W. Scheirer, K. Bowyer, and P. Flynn, "Fast face image synthesis with minimal training," in *WACV*, 2019.

[8] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *CVPR*, 2020.

[9] R. Or-El, S. Sengupta, O. Fried, E. Shechtman, and I. Kemelmacher-Shlizerman, "Lifespan age transformation synthesis," in *ECCV*, 2020.

[10] A. Tewari, M. Elgharib, G. Bharaj, F. Bernard, Perez P. Seidel, H-P, M. Zollhofer, and C. Theobalt, "Stylerig: Rigging stylegan for 3d control over portrait images," in *CVPR*, 2020.

[11] Y. Choi, Y. Uh, J. Yoo, and J-W Ha, "Stargan v2: Diverse image synthesis for multiple domains," in *CVPR*, 2020.

[12] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2016.

[13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.

[14] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *NeurIPS*, 2016.

[15] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *NeurIPS*, 2017.

[16] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *ICML*, 2017.

[17] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," in *arXiv:1505.00853*.

[18] J-Y. Zhu, T. Park, P. Isola, and A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *ICCV*, 2017.

[19] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *AISTATS*, 2010.

[20] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever, "Generative pretraining from pixels," in *ICML*, 2020.

[21] S. Beery, Y. Liu, D. Morris, J. Piavis, A. Kapoor, N. Joshi, M. Meister, and P. Perona, "Synthetic examples improve generalization for rare classes," in *WACV*, 2020.

[22] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[23] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research (JMLR)*, vol. 15, no. 56, pp. 1929–1958, 2014.

[24] M. Abadi et al., "Tensorflow: A system for large-scale machine learning," in *12th USENIX Conference on Operating Systems Design and Implementation*, 2016, pp. 265–283.

[25] F. Chollet et al., "Keras," https://github.com/fchollet/keras, 2015.

[26] S. Yun, D. Han, S.J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *ICCV*, 2019.

[27] E. Schonfeld, B. Schiele, and A. Khoreva, "A u-net based discriminator for generative adversarial networks," in *CVPR*, 2020.