IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Joint Learning of Generative Translator and Classifier for Visually Similar Classes

BYUNGIN YOO[1,2], TRISTAN SYLVAIN[3], YOSHUA BENGIO[4], and JUNMO KIM[5]

[1]School of Electrical Engineering, Korea Advanced Institute of Science and Technology (e-mail: byungin.yoo@kaist.ac.kr)
[2]Computer Vision Lab., Samsung Advanced Institute of Technology (e-mail: byungin.yoo@samsung.com)
[3]Montreal Institute for Learning Algorithms (e-mail: tristan.sylvain@gmail.com)
[4]Montreal Institute for Learning Algorithms (e-mail: yoshua.bengio@mila.quebec)
[5]School of Electrical Engineering, Korea Advanced Institute of Science and Technology (e-mail: junmo.kim@kaist.ac.kr)

Corresponding author: junmo.kim (e-mail: junmo.kim@kaist.ac.kr).

ABSTRACT In this paper, we propose a Generative Translation Classification Network (GTCN) for improving visual classification accuracy in settings where classes are visually similar and data is scarce. For this purpose, we propose joint learning from a scratch to train a classifier and a generative stochastic translation network end-to-end. The translation network is used to perform on-line data augmentation across classes, whereas previous works have mostly involved domain adaptation. To help the model further benefit from this data-augmentation, we introduce an adaptive fade-in loss and a quadruplet loss. We perform experiments on multiple datasets to demonstrate the proposed method's performance in varied settings. Of particular interest, training on 40% of the dataset is enough for our model to surpass the performance of baselines trained on the full dataset. When our architecture is trained on the full dataset, we achieve comparable performance with state-of-the-art methods despite using a light-weight architecture.

INDEX TERMS Artificial neural networks, Feature extraction, Image classification, Image generation, Pattern analysis, Semisupervised learning

I. INTRODUCTION

GENERATIVE models have received significant interest in the past years. Although recent models can generate realistic and diverse data, more study is needed to ascertain whether such methods can also be useful in enhancing classification accuracy on hard settings such as when classes are visually similar, or there is a scarcity of training data. For example, face liveness detection in biometrics is a crucial problem where it is hard to distinguish between real faces and printed fake faces, because examples from the two classes are very similar [1], [4], [19], [22], [33], [36], [37]. In this application, obtaining a high true acceptance ratio (TAR) and a low false acceptance ratio (FAR) is important as a high TAR is essential for user convenience, whereas a low FAR results in better security. This paper is motivated by two questions:

- If two classes, A and B, are visually very similar, how can we improve classifiers by employing cross-class generative models?
- When data is scarce, how can we use generative models to learn better representations for the visually similar classes?

In practice, most past approaches to solving these two questions fall in two categories. The first one is using complex models, which are often hard to train, and make it difficult to perform fast inference in settings with limited computing resources, such as on a smartphone. The second is to collect large amounts of training data, which is costly, time-consuming, and not always straightforward. In this paper, we propose a Generative Translation Classification Network that uses the translation model of visual classes to assist the training of the classifier via exploiting joint learning. If the translation model is able to effectively augment the quantity of training samples we expect both the issue of having closely distributed classes, and the lack of sufficient amounts of training data to be mitigated. We should note the similarity of our proposed method with the way the brain learns. There is biological evidence that long-term memory is formed by the collaboration between the hippocampus and the prefrontal cortex [29]. The hippocampus recalls slightly distorted

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

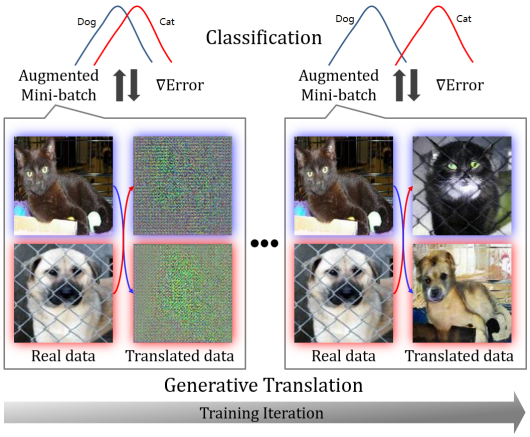


FIGURE 1: A generative translation model progressively generates samples to augment data in mini-batch for joint training of a classifier.

memories, in a way which inspired our translation network generating variations on training examples. The prefrontal cortex uses such memories, and has a role analogous to the classifier we used. In summary, our contributions are the following:

- To augment mini-batch data (AMB) during training, we use inter-class translated samples based on joint learning of a translation model and a classifier. Specifically, half the training samples seen by the classifier are inter-class translated samples that are stochastically generated (ST), while the other half of the samples are real samples of training data to preserve the original data distribution. This is a novel attempt to couple a generative translation network and a classifier in a unified architecture for improving classification of visually similar classes.
- Early on during joint learning, translated samples are of poor quality. We use adaptive fade-in training (AF) for the classifier, automatically adjusting the importance of real and translated samples to gradually adapt the influence from translated samples. We design a novel quadruplet loss (QL) that helps preserve intra-class distribution and taking inter-class distribution apart, even though generated samples are used for training. This is due to the fact that this loss encourages similarity between the embeddings of real and intra-class translated images, and dissimilarity between the embeddings of inter-class samples.

II. RELATED WORK

Since the introduction of generative adversarial networks (GANs) [14], many GAN-based generative models have been introduced, e.g. [3], [6], [9], [25], [28]. A typical issue is mode collapse, reduced diversity of generated samples, which would be detrimental to improve classification accuracy in our setup. In addition to this, our specific need for inter-class generated samples is difficult to meet with classical GANs, hence we focus on adversarial translation models. Domain translation has received renewed interest in recent years, thanks in large part to the use of adversarial methods. The first results using adversarial training used paired samples [16]. More recent developments using cycle consistency allow for unsupervised domain translation using unpaired samples [21], [35], [42]. Finally, newer methods extend this approach to problems requiring many-to-many mappings or multi-modal data distributions [2], [7], [15], [18], [43]. The power of these approaches lies in being able to learn such transformations without requiring examples of a one-to-one mapping between training data in source and target domains.

Multiple works have used the discriminators of GANs as semi-supervised learning classifiers [8], [10], [11], [13], [34]. Typically, the semi-supervised classifiers take a tiny portion of labeled data and a much larger amount of unlabeled data from the same domain. The goal is to use both labeled and unlabeled data to train a neural network so that it can map a new data point to its correct class [27]. A simple yet effective idea for semi-supervised learning is to turn a classification problem with n classes into a classification problem with $n + 1$ classes, with the additional class corresponding to fake images [31]. However, the methods are not directly augmenting data of existing classes, since they assume generated data is a new class.

The feature matching loss [31] was introduced as an additional loss term to prevent instability of GANs from over-fitting on the discriminator. Specifically, the generator is trained to match the expected value of the features on an intermediate layer of the discriminator. While feature matching losses bring benefits, a cost function that can account for inter- and intra-class relationships is required in our case. Deep learning networks with variants of a triplet loss become common methods for face verification [32] and person re-identification [5]. Despite this interest, such losses have seldom been applied to generative models.

From another perspective, between-class data augmentation methods have been proposed [38]. Images are generated by mixing two training images belonging to different classes with a random ratio. The training procedure seeks to minimize the KL-divergence between the outputs of a trained model, and a target computed by interpolating the two one-hot target vectors of the initial examples using the same ratio. Even though the methods achieve superior results on visual recognition, it is not clear whether the classifier learns shaper and more diverse data on visually similar classes.

Data subrogation is an important technique that a reduced set of real credit card data is used to generate surrogate multivariate series of credit card data transactions [30]. The surrogate multivariate data are constrained to have the same covariance, marginal distributions, and joint distributions as the original multivariate data. The surrogate data are combined with real data to consist of the mixed dataset. The detection results obtained from the mixed dataset are comparable to those obtained from only real data, while the detection accuracy of the mixed dataset doesn't outperform

that of the real dataset.

In this paper, we use translated images obtained via a generative model to perform data augmentation in each mini-batch and train a classifier. Even though many previous works have applied generative models to classifier training [**?**], this work is first to focus on joint learning to improve classification accuracy for visually similar images.

III. PROPOSED METHODS

We design a unified deep network architecture that combines a classifier C and a generative translation model G to perform on-line data augmentation of the mini-batch. The proposed architecture is explained in Figure 2.

A. FORMULATION OF PROPOSED METHODS

Let $\{x_y^i : 1 \leq i \leq n, y \in Y\}$ be a dataset such that $x_y^i \in X$ is the i-th sample belonging to class $y \in Y$. We consider a learning algorithm that trains a classifier

$$C : X \rightarrow Y \tag{1}$$

by jointly using a generative translation model $G(\tilde{x}|x)$. Our goal is to improve the classifier C by utilizing the on-line translated samples $\tilde{x}$ as additional training data.

More formally, C is a classifier to discriminate two-class cases, where Y=$\{A, B\}$. Generative translation models G are employed to produce $\tilde{x}$ from x as:

$$\tilde{x}_A^i = G_{BA}(x_B^i, z_{BA}) \ , \tag{2}$$

$$\tilde{x}_B^i = G_{AB}(x_A^i, z_{AB}) \ , \tag{3}$$

where x_A^i and x_B^i are real samples of class A and B, z_{BA} and z_{AB} are tensors of random noise used for stochastic translation, and $\tilde{x}_A^i$ and $\tilde{x}_B^i$ are translated samples across the classes. One of our important contributions is to add a translator into the full pipeline for a classifier and to jointly train the translator and the classifier. Namely, G_{AB} and G_{BA} provide diverse and challenging data continuously for training classifier C. So, we suppose G_{AB} and G_{BA} generate effective images from the standpoint of decision boundary of the classifier.

1) Definition of augmented mini-batch

The augmented mini-batch that is input to the classifier C during training is defined as follows:

$$AMB_k = \{x_A^i\}_{i=1}^m \cup \{\tilde{x}_A^i\}_{i=1}^m \cup \{x_B^i\}_{i=1}^m \cup \{\tilde{x}_B^i\}_{i=1}^m \ , \tag{4}$$

where k is the training iteration, m is the number of samples in each set. The $\tilde{x}$'s in AMB_k increase diversity of training data, while x's preserve original distribution of training data.

2) Loss of stochastic translator

Visual translation objective across classes is that a source class borrow underlying structure from a target class, while maintain style of the source class. To meet the needs, cycle consistency losses, L_{cyc}^A and L_{cyc}^B, are used to train stochastic translators, G_{AB} and G_{BA}. The objective is expressed as:

$$L_{cyc}^A = \mathbb{E}_{x_A \sim P_{data}}[\|G_{BA}(\tilde{x}_B, z_{BA}) - x_A\|_1] \ , \tag{5}$$

$$L_{cyc}^B = \mathbb{E}_{x_B \sim P_{data}}[\|G_{AB}(\tilde{x}_A, z_{AB}) - x_B\|_1] \ . \tag{6}$$

For generating realistic images, an adversarial loss L_{adv}^A is applied to train D_A and G_{BA}. Similarly, an adversarial loss L_{adv}^B is applied to train D_B and G_{AB}. The objective is expressed as:

$$\begin{aligned} L_{adv}^A = \ & \mathbb{E}_{x_A \sim P_{data}}[\log(D_A(x_A))] \\ & + \mathbb{E}_{x_B \sim P_{data}}[\log(1 - D_A(G_{BA}(x_B, z_{BA})))] \ , \end{aligned} \tag{7}$$

$$\begin{aligned} L_{adv}^B = \ & \mathbb{E}_{x_B \sim P_{data}}[\log(D_B(x_B))] \\ & + \mathbb{E}_{x_A \sim P_{data}}[\log(1 - D_B(G_{AB}(x_A, z_{AB})))] \ . \end{aligned} \tag{8}$$

3) Adaptive fade-in learning

In the early stages, translated samples in AMB_k are relatively poor and relying on them too much could be detrimental for training classifier C. We therefore design an adaptive fade-in loss (AF) that adapts the importance given to real and translated samples during training. The objective is defined as:

$$\begin{aligned} L_{cls} = \ & \alpha \cdot \mathbb{E}_{x_A \sim P_{data}}[-\log(P(y = A|x_A))] \\ & + (1 - \alpha) \cdot \mathbb{E}_{x_B \sim P_{data}}[-\log(P(y = A|G_{BA}(x_B, z_{BA})))] \\ & + \beta \cdot \mathbb{E}_{x_B \sim P_{data}}[-\log(P(y = B|x_B))] \\ & + (1 - \beta) \cdot \mathbb{E}_{x_A \sim P_{data}}[-\log(P(y = B|G_{AB}(x_A, z_{AB})))] \ , \end{aligned} \tag{9}$$

where α and β are parameters between 0 to 1. We use a categorical cross-entropy loss L_{cls} on the Softmax output of C. The classification loss L_{cls} is used to train C, G_{AB}, and G_{BA} by backpropagation. In the equation 9, the parameters α and β control the relative importance of four training inputs, x_A, $\tilde{x}_A$=$G_{BA}(x_B, z_{BA})$, x_B, and $\tilde{x}_B$=$G_{AB}(x_A, z_{AB})$ in AMB_k. Specifically, α controls the relative weight given to real data x_A and augmented data $\tilde{x}_A$ to train a classifier C. Similarly, β controls the relative importance given to real data x_B and augmented data $\tilde{x}_B$ to train the classifier C.

4) Quadruplet loss for inter/intra-class

We design a quadruplet loss for the proposed architecture that enforces explicit relationships between classes. The objective is defined as:

$$\begin{aligned} L_{quad} = \ & \\ & [\|f_C(x_A) - f_C(\tilde{x}_A)\|_2^2 - \|f_C(x_A) - f_C(\tilde{x}_B)\|_2^2 + \eta_a]_+ \\ & + [\|f_C(x_B) - f_C(\tilde{x}_B)\|_2^2 - \|f_C(x_B) - f_C(\tilde{x}_A)\|_2^2 + \eta_b]_+ \\ & + [-\|f_C(x_A) - f_C(x_B)\|_2^2 - \|f_C(\tilde{x}_A) - f_C(\tilde{x}_B)\|_2^2 + \eta_c]_+ \ , \end{aligned} \tag{10}$$

where $[\cdot]_+ = max(\cdot, 0)$, and $f_C(\cdot)$ denotes the embedding features of x_A, $\tilde{x}_A$, x_B, and $\tilde{x}_B$ in AMB_k at the last feature layer of the classifier C. The thresholds η_a, η_b, and η_c are margins that are enforced between positive and negative pairs. Likewise L_{cls}, L_{quad} is used to train C, G_{AB}, and G_{BA}. The quadruplet loss encourages the similarity of intra-class samples and the dissimilarity of inter-class samples, which is useful for classification. Specifically, the intention of the designing the quadruplet loss considers all six pairs out of the four elements, x_A, x_B, $\tilde{x}_A$, and $\tilde{x}_B$. The first and second

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

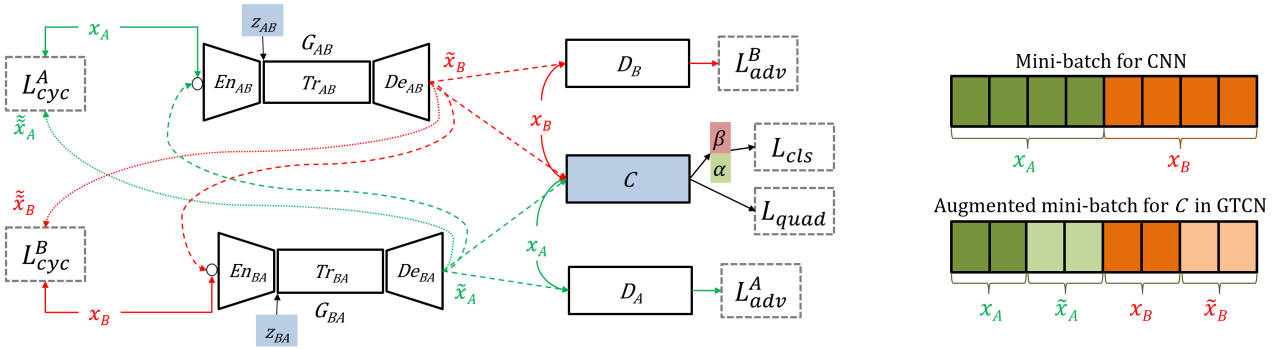


FIGURE 2: Overview of the proposed Generative Translation Classification Network : (Left) G_{AB} is a translator from class A to B, while G_{BA} is a corresponding translator from class B to A. G_{AB} consists of three main blocks, an encoder En_{AB} (three convolutional layers), transformer Tr_{AB} (nine residual blocks), and decoder De_{AB} (two deconvolutional layers and one convolutional layer). Likewise, G_{BA} consists of En_{BA}, Tr_{BA}, and De_{BA}. Random noise $\{z_{AB}, z_{BA}\} \in \mathbb{R}^{32\times32\times3}$ for stochastic translation is sampled from a uniform distribution over $[-1, 1]$ and concatenated to the output maps of En_{AB} and En_{BA}. D_A and D_B are the corresponding discriminators of adversarial training for classes A and B respectively. C is a simple classifier consisting of six convolutional layers over classes A and B. α and β are the trainable parameters of the adaptive fade-in loss. x_A and x_B are real samples from classes A and B respectively. The corresponding translated samples are denoted by $\tilde{x}_B$ and $\tilde{x}_A$. The cyclic reconstructed samples are $\tilde{\tilde{x}}_A = G_{BA}(\tilde{x}_B)$ and $\tilde{\tilde{x}}_B = G_{AB}(\tilde{x}_A)$. Note that only C is used at test-time. Details of loss functions $L*$ are in proposed methods section. (Right) Comparison of mini-batch structure. A mini-batch used for training C in GTCN contains real and translated samples, while the baseline CNNs are only trained on real samples.

items in equation 10 are aim to minimize intra-class variation and maximizing means of inter-classes. The final term acts differently, it aims to auxiliary regulate the inter-class means.

5) Overall training objective

The training objective of a classifier C in GTCN is:

$$L_C = L_{cls} + L_{quad} \ . \tag{11}$$

The training objective of translator G in GTCN is:

$$L_G = L_C + L^A_{adv} + L^B_{adv} + \lambda \cdot (L^A_{cyc} + L^B_{cyc}) \ , \tag{12}$$

where λ is a weight parameter to adjust the relative importance of the cyclic consistency losses. Finally, we aim to optimize:

$$G^*_{AB}, G^*_{BA} = \arg \min_{\{G_{AB},G_{BA}\}} \max_{\{D_A,D_B\}} L_G \ , \tag{13}$$

$$C^* = \arg\min_{\{C\}} L_C \ . \tag{14}$$

Since the translators G_{AB}, G_{BA} and the discriminators D_A, D_B are jointly optimized with C, G_{AB} and G_{BA} generate effective translated samples that assist improvement of classification accuracy for C.

B. NETWORKS MODEL

As the baseline translator network, we adapt the implementation of CycleGAN. [1]. All of the classifiers in the experiments use the simple architecture consisting of six convolution layers, so as to run on performance-critical systems such as smartphones. The parameters are denoted:

$$\theta_C = \{w^1, w^2, w^3, w^4, w^5, w^6, w^s\}. \tag{15}$$

The specific parameters for the network are given in Table 1. As θ_C only consists of 73,904 (k=2) / 75,952 (k=4) parameters, the trained networks Cs have a small memory footprint and can be run on smart devices in real-time without GPU acceleration.

TABLE 1: Design of a light-weight classifier C. Resolution of input image x is 128×128. $Conv$ stands for a convolutional layer, BN for batch normalization, $ReLU$ for rectified linear units, and $Pool$ corresponds to a pooling layer. FC denotes a fully connected layer. k is the number of classes for $y \in Y$.

Layers	Parameters	Outputs
$Conv1$-BN-$ReLU$	w^1: $3 \times 3 \times 3$, kernels=16	$128 \times 128 \times 16$
$Pool1$	max: 2×2, stride=2	$64 \times 64 \times 16$
$Conv2$-BN-$ReLU$	w^2: $3 \times 3 \times 16$, kernels=16	$64 \times 64 \times 16$
$Pool2$	max: 2×2, stride=2	$32 \times 32 \times 16$
$Conv3$-BN-$ReLU$	w^3: $3 \times 3 \times 16$, kernels=32	$32 \times 32 \times 32$
$Pool3$	max: 2×2, stride=2	$16 \times 16 \times 32$
$Conv4$-BN-$ReLU$	w^4: $3 \times 3 \times 32$, kernels=32	$16 \times 16 \times 32$
$Pool4$	max: 2×2, stride=2	$8 \times 8 \times 32$
$Conv5$-BN-$ReLU$	w^5: $3 \times 3 \times 32$, kernels=64	$8 \times 8 \times 64$
$Conv6$-BN-$ReLU$	w^6: $3 \times 3 \times 64$, kernels=64	$8 \times 8 \times 64$
$Pool6$	average: 2×2, stride=2	$4 \times 4 \times 64$
$Dropout$	-	1024
FC	w^s:$1024 \times k$	k
$Softmax$	-	$P(y\|x)$

IV. EXPERIMENTS

A. EXPERIMENTAL SETUP AND CONFIGURATION

The proposed methods are evaluated with datasets, those are (1) *Face liveness* and (2) *Dogs vs. Cats* for visually similar binary classes. The datasets represent human faces and animals, and as a result have very different characteristics in terms of underlying structure and styles. The face liveness detection dataset[2] [41] contains very similar fake and real face images that have strong underlying structure and

[1] https://junyanz.github.io/CycleGAN/

[2] http://www.cbsr.ia.ac.cn/english/FASDB_Agreement/Agreement.pdf

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

FIGURE 3: Example from face liveness and dogs vs. cats. (First row) The first column shows real face images. The second, third, and fourth column show fake face images those are corresponding to video display, printed photos, and face masks with real eyes. (Second row) The first column shows cat images, others show dog images.

different style of texture. Note that no subjects are present in both training and test sets. Additionally, we evaluate on the dogs vs. cats classification dataset[3], because they share part of similar style although they are belong to the different species. We explore varying the amount of training data available by training on different subsets containing 100%, 80%, 60%, and 40% of the examples of the original training set. The specific configuration is described in Table 2 and examples images of datasets are shown in Figure 3.

We trained all the models in the experiments from scratch without pre-training and extra datasets. The mini-batches used to train the CNNs contained eight real samples, while four real samples and four translated samples were present in the mini-batches used for GTCNs and other compared models. All of the networks' parameters were optimized using the Adam optimizer. We do not vary the learning rate for the first half of epochs and linearly decay the rate to zero over the next half of epochs. The base learning rate is 0.0002 and the number of training epochs was set to 100. To prevent over-fitting, data augmentation transforms were applied such as rotation, intensity, color adjustment, and scaling variation. Regarding the hyper parameters of the translator network, λ was set to 10 in all experiments. The hyperparameters for the quadruplet loss were set to: $\{\eta_a/\eta_b$=2, η_c=6$\}$ for face liveness, $\{\eta_a/\eta_b$=0.5, η_c=8$\}$ for dogs vs. cats.

Since binary classification is the problem of deciding to which class $y \in Y = \{A, B\}$ a given test image x belongs, we utilize logit values obtained before calculating the Softmax instead of using the Softmax output that was used for training. Outputs of the FC layer in the classifier C are utilized to calculate a score for class A as: $SC(y = A|x; \theta_C) = \frac{FC_A - FC_B}{2}$, where FC_A is a logit value for class A, FC_B is a logit value for class B, $SC(y = A)$ is the score for class A. The calculated $SC(y = A)$ is employed to decide on the class as follows:

$$C(x; \theta_C) = \begin{cases} y = A & \text{if } SC(y = A) \geq th \\ y = B & \text{otherwise} \end{cases}, \qquad (16)$$

where th is an acceptance threshold to decide if x is a class A. For example, if th is set to be high, then we can calculate TAR for overall test samples at low FAR.

[3]https://www.kaggle.com/c/dogs-vs-cats

TABLE 2: Configuration of experimental datasets and subsampled variants. LV* and DC* are reconfigured training datasets.

Face liveness	LV100	LV80	LV60	LV40	Test
Number of subjects	20	16	12	8	30
Live face	10,891	8,333	6,443	4,493	15,904
Fake face	34,165	26,557	20,188	13,511	49,862
Dogs vs. cats	DC100	DC80	DC60	DC40	Test
Dogs	12,500	10,000	7,500	5,000	6,253
Cats	12,500	10,000	7,500	5,000	6,235

B. TRAINING WITH SMALL DATA VOLUMES

First, we evaluate performance of baseline CNNs on different subsets of the training set to study the effect of data scarcity. In addition to this, other compared methods and the proposed GTCN are trained with only 40% of training data. We compared our method to between-class learning (BC/BC$^+$) [38] and semi-supervised learning with the classical $N + 1$ class setting [31]. In terms of generative models, we consider a least-squares GAN (LSGAN) [24] and a variational autoencoder (VAE) [17] as alternative methods to be compared. Table 3 and Table 4 show evaluation results. As the two binary classes are visually similar and there is therefore a lack of diverse data, baseline CNNs have low accuracy despite using deep networks and 100% of training data in both datasets. The overall sparsity of training data causes a poor true acceptance rate. Interestingly, BC/BC$^+$ did not fare better than CNNs in the range of low false acceptance rates, although mean accuracy for those methods was higher than for CNNs. We hypothesize this is due to the fact that BC/BC$^+$ may be hard to produce good mixing data in cases where classes are very similar, caused by the proximity of the manifolds corresponding to those classes. All deep generative models outperformed the CNN baseline for LV40 dataset. In particular, VAE-based methods achieve a good accuracy, most likely because such methods generate diverse samples. For all evaluation datasets, GTCNs that were trained with 40% of the dataset clearly outperformed all of other compared methods including CNNs trained on 100% of training data. With GTCNs that were trained with 20% of the dataset, this results in lower accuracy compared to CNNs trained on 100% of training data. As a result, we consider that a dataset comprising 40% of the examples is an empirical lower-bound on "acceptable" performance in this context.

C. TRAINING WITH THE FULL DATASET

In this experiment, we used 100% of the training dataset to verify the scalability of the proposed methods. We added CNN-256 models that are trained and tested on images of size 256×256 pixels, while other models use images of size 128×128 pixels. Table 5 and Table 6 show evaluation results. In a similar fashion to the experiments using reduced versions of the datasets, the performance of data augmentation methods BC/BC$^+$ and the compared deep generative models that mainly employ intra-class data augmentation show

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

TABLE 3: Evaluation results of training with small volume of the face liveness dataset. ACC is mean accuracy. Cells in columns of FAR show percentage value of TAR.

Model	Dataset	ACC	FAR=$\frac{1}{100}$	FAR=$\frac{1}{1k}$	FAR=$\frac{1}{5k}$	FAR=$\frac{1}{50k}$
CNN	LV40	74.71	43.42	22.52	16.90	9.75
CNN	LV60	81.16	50.26	27.85	20.82	13.20
CNN	LV80	86.79	64.13	41.38	27.77	22.29
CNN	LV100	84.87	69.33	45.38	32.77	21.72
BC	LV40	88.62	43.13	17.78	12.61	5.28
BC$^+$	LV40	86.09	28.10	11.78	7.65	5.48
LSGAN	LV40	79.62	60.88	37.47	30.25	23.58
VAE	LV40	90.57	67.93	42.71	32.94	17.54
Semi-sup.	LV40	89.89	58.49	41.95	33.63	21.64
GTCN	LV40	**92.26**	**74.81**	**62.36**	**54.04**	**39.01**

TABLE 4: Evaluation results of training with small volume of the dogs vs. cats dataset. EER is the equal error rate.

Model	Dataset	ACC	FAR=$\frac{1}{100}$	FAR=$\frac{1}{1k}$	FAR=$\frac{1}{5k}$	EER
CNN	DC40	92.74	79.49	54.82	37.26	7.50
CNN	DC60	93.38	81.92	65.71	44.03	6.60
CNN	DC80	93.63	82.90	66.43	42.85	6.25
CNN	DC100	93.96	82.98	63.96	58.83	6.06
BC	DC40	92.82	77.13	28.77	5.16	7.23
BC$^+$	DC40	92.33	69.94	23.08	4.78	8.14
LSGAN	DC40	93.13	79.44	54.88	25.07	6.84
VAE	DC40	92.89	79.10	56.97	50.97	7.12
Semi-sup.	DC40	92.67	76.26	57.64	37.21	7.18
GTCN	DC40	**94.28**	**84.15**	**67.41**	**54.18**	**5.66**

TABLE 5: Evaluation results of training with full volume of the face liveness dataset(LV100). All of models use images of 128×128 pixels, except the CNN-256, which uses 256×256. * are results of score fusion based model.

Model	ACC	FAR=$\frac{1}{100}$	FAR=$\frac{1}{1k}$	FAR=$\frac{1}{5k}$	FAR=$\frac{1}{50k}$
CNN	84.87	69.33	45.38	32.77	21.72
CNN-256	91.09	81.68	64.51	58.87	48.16
BC	90.17	56.93	23.84	16.53	10.92
BC$^+$	91.19	52.97	29.28	21.20	8.51
LSGAN	93.88	81.21	50.97	37.19	26.24
VAE	95.50	86.71	77.78	70.09	61.63
Semi-sup.	94.65	76.75	43.26	31.18	18.04
GTCN	**97.65**	**93.62**	**86.74**	**81.81**	**76.73**
CNN*	95.76	93.15	82.12	79.41	61.94
CNN-256*	97.69	96.34	91.57	83.62	74.16
GTCN*	**99.03**	**98.87**	**95.99**	**90.73**	**85.76**

TABLE 6: Evaluation results of training with full volume of the dogs vs. cats dataset(DC100). CNN-256 uses 256×256.

Model	ACC	FAR=$\frac{1}{100}$	FAR=$\frac{1}{1k}$	FAR=$\frac{1}{5k}$	EER
CNN	93.96	82.98	63.96	58.83	6.06
CNN-256	95.52	89.83	78.85	69.03	4.51
BC	93.79	78.46	32.98	17.93	6.49
BC$^+$	93.68	83.80	38.83	19.44	7.00
LSGAN	95.21	86.09	72.46	58.41	4.81
VAE	94.58	84.88	68.24	60.14	5.49
Semi-sup.	94.86	85.32	67.30	59.50	5.18
GTCN	**95.61**	**88.44**	**75.22**	**66.72**	**4.37**

limited performance. VAEs and BC methods show relatively

TABLE 7: Recall results of training with full volume of the face liveness dataset (LV100) for sub-categories

Model	Real Face	Video Face	Photo Face	Mask Face
CNN	96.28	83.05	73.51	89.35
GTCN	93.68	97.29	98.04	99.98

TABLE 8: Performance comparisons with the deep learning based face liveness detection methods

Method	EER	Image Resolution	Methods
Yang [40]	4.95	128×128	AlexNet + SVM
DPCNN [20]	4.50	224×224	VGG19 + SVM
Atoum [4]	2.67	128×128	CNN (10 patch + depth)
Nguyen [26]	1.70	224×224	VGG19 + MLBP +SVM
CNN*	3.78	128×128	CNN (Face + Context)
CNN-256*	2.53	256×256	CNN (Face + Context)
GTCN	3.49	128×128	CNN (Face)
GTCN	2.09	128×128	CNN (Context)
GTCN*	**1.02**	128×128	CNN (Face + Context)

TABLE 9: Comparison of architectures in term of number of parameters, number of layers, MACs (Multiply-Accumulate Operation), and latency on Mobile Application Processor

Base Model	Parameters	Layers	MACs	Speed @Mobile AP
AlexNet [40]	61M	8	181M	100ms @CPU
VGG19 [20], [26]	143M	19	19.7G	170ms @GPU
Light-weight C	74K	6	27M	15ms @CPU

good performance in the face liveness dataset, since faces are highly structured images. However, in both evaluation datasets, GTCNs outperform all of other compared methods including CNNs trained with a larger input, 256×256 pixels. As a matter of course, a small size of input can lead to two advantages, 1) light-weight inference and 2) recognizing far-distance images. Effectiveness of GTCNs for each sub-category in the face liveness dataset such as real face, video face, photo face, and mask face is shown in Table 7. Recall=$\frac{\text{True Positives}}{\text{True Positives+False Negatives}}$ is chosen as an evaluation metric. In sub-category of fake faces, the recall is clearly improved, while the recall of real faces is negligibly degraded. Lastly, we use score fusion models to achieve the best accuracy in the face liveness detection. Formally, we consider two classifiers, C_{rs} and C_{ct}. The classifier C_{rs} that discern local texture, shape and reflection pattern is given by

$$SC_{rs} = C_{rs}(x_{rs}; \theta_{rs}), \forall x_{rs} \in Resize(FD(I)), \quad (17)$$

where SC_{rs} is a liveness score of a patch x_{rs} that is a resized image of a detected face region, FD is a face detector [39], and θ_{rs} are learned parameters. Similarly, the classifier C_{ct} that discriminates global shapes and contextual pattern is given by

$$SC_{ct} = C_{ct}(x_{ct}; \theta_{ct}), \forall x_{ct} \in Resize(I), \quad (18)$$

where SC_{ct} is a liveness score of a patch x_{ct} that is a resized contextual image from an input image I, and θ_{ct} are learned parameters. To decide whether the given image I is live or

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

FIGURE 4: Examples of image patches: resized detected face x_{rs}, and resized contextual x_{ct} from input image I. Note that x_{rs} and x_{ct} are images of 128×128 pixels.

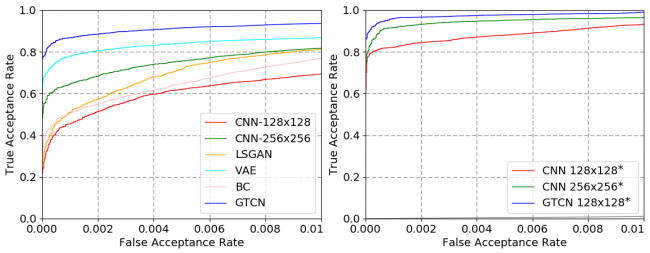


FIGURE 5: Receiver Operating Characteristic (ROC) comparison of models trained with full volume of the face liveness dataset. (Left) ROC curves of single models (Right) ROC curves of two patches based models

TABLE 10: Ablation experimental results of the proposed methods on face liveness and dogs vs. cats datasets

Face liveness	CNN	Separate	+Joint	+AF	+AF/QL	+AF/QL/ST
Accuracy	84.87	94.93	96.43	95.35	97.04	**97.65**
FAR=$\frac{1}{50k}$	21.72	27.15	67.10	55.71	74.42	**76.73**
Dogs vs. cats	**CNN**	**Separate**	**+Joint**	**+AF**	**+AF/QL**	**+AF/QL/ST**
Accuracy	93.96	95.05	95.44	95.72	94.93	**95.61**
FAR=$\frac{1}{5k}$	58.83	42.25	64.38	57.37	60.29	**66.72**

not, the final liveness score is calculated by

$$SC = \alpha_{rs} \times SC_{rs} + \alpha_{ct} \times SC_{ct}, \tag{19}$$

where α_{rs} and α_{ct} are coefficient values to combine scores from two models to perform liveness prediction with a late fusion method. We set α_{rs}=1 and α_{ct}=0.6, respectively. Example of patches are shown in Figure 4.

The proposed methods outperform previous state-of-the-art models and score fusion based models of CNN and CNN-256 as shown in table 8 without pretraining with extra datasets, large networks, and combining SVM with traditional hand craft features. Figure 5 shows the ROC curves for single models and two patches based models. GTCNs achieve superior true acceptance rates in the range of low false acceptance rates. Note that our two patches based model is lighter and faster than others, since we used a light-weight CNN that has just 73,904 parameters. In Table 9, there are specific comparison with the compared methods that employ other CNN architecture such as AlexNet and VGG19, while our proposed architecture consists of just 6 convolution layers. In our implementation, the ensemble of the two light-weight networks for two facial patches is running in 30 msec by using single mobile CPU on Samsung Galaxy S10.

V. DISCUSSION AND ANALYSIS

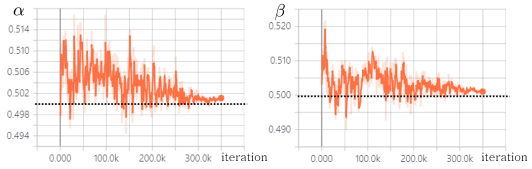


FIGURE 6: Example variation of α and β for adaptive fade-in learning

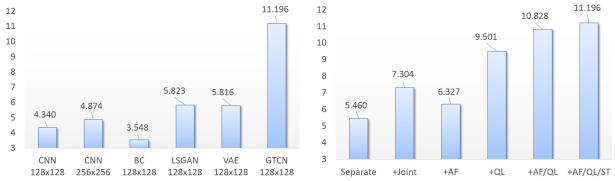


FIGURE 7: Comparison of Fisher's criterion scores. Vertical axis on the charts represents score J. (Left) Compared methods (Right) Ablation study results of the proposed GTCN

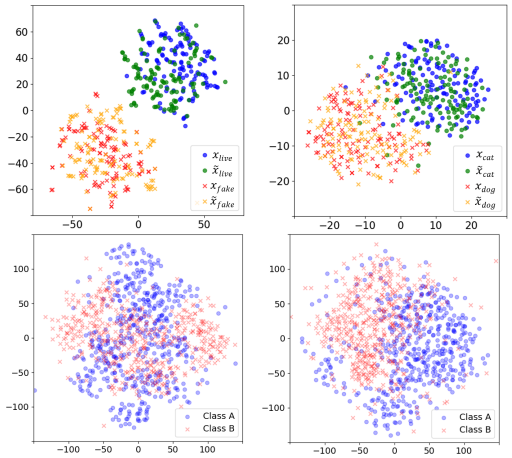


FIGURE 8: First two plots show examples of t-SNE analysis for training data on GTCN: (1) Face liveness and (2) Dogs vs. cats. The augmented data points $\tilde{x}$ achieve a good coverage of all the area of the embedding space corresponding to their true class in binary classification. We can consider that they are good at augmenting training data in that regard. Second two plots show comparison of methods by t-SNE analysis for LV40 test dataset: (3) CNN and (4) the proposed GTCN

A. ABLATION STUDY

We perform an ablation study, the results of which are shown in Table 10. As first, we trained G_{AB} and G_{BA} separately and used translators with fixed parameters to augment data for training C. $Seperate$ learning shows better accuracy then baseline CNN, because translators generate diverse data. However, proposed $Joint$ learning is superior to $Separate$ learning. One of the disadvantages of $Separate$ learning is that it takes longer to train, since two models are trained sequentially. To analyze the effect of each component of the GTCN, learning options were adapted sequentially. AF stands for adaptive fade-in learning, QL for quadruplet loss, and ST corresponds to stochastic translation. Because translated images could be sometimes suppressed by applying AF during joint training, the performance are not much improved

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

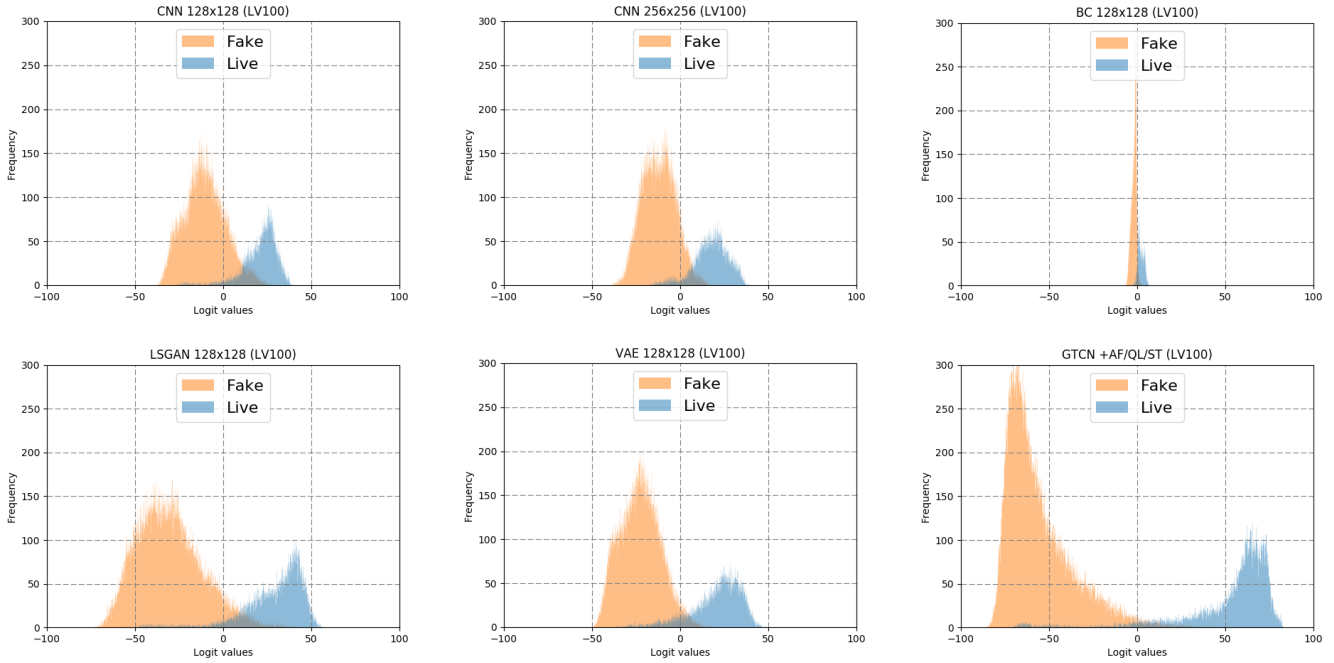


FIGURE 9: Histograms of logit values for liveness test dataset. The compared models are trained on LV100 dataset. (1) CNN 128×128 (2) CNN 256×256 (3) BC 128×128 (4) LSGAN 128×128 (5) VAE 128×128 (6) GTCN 128×128

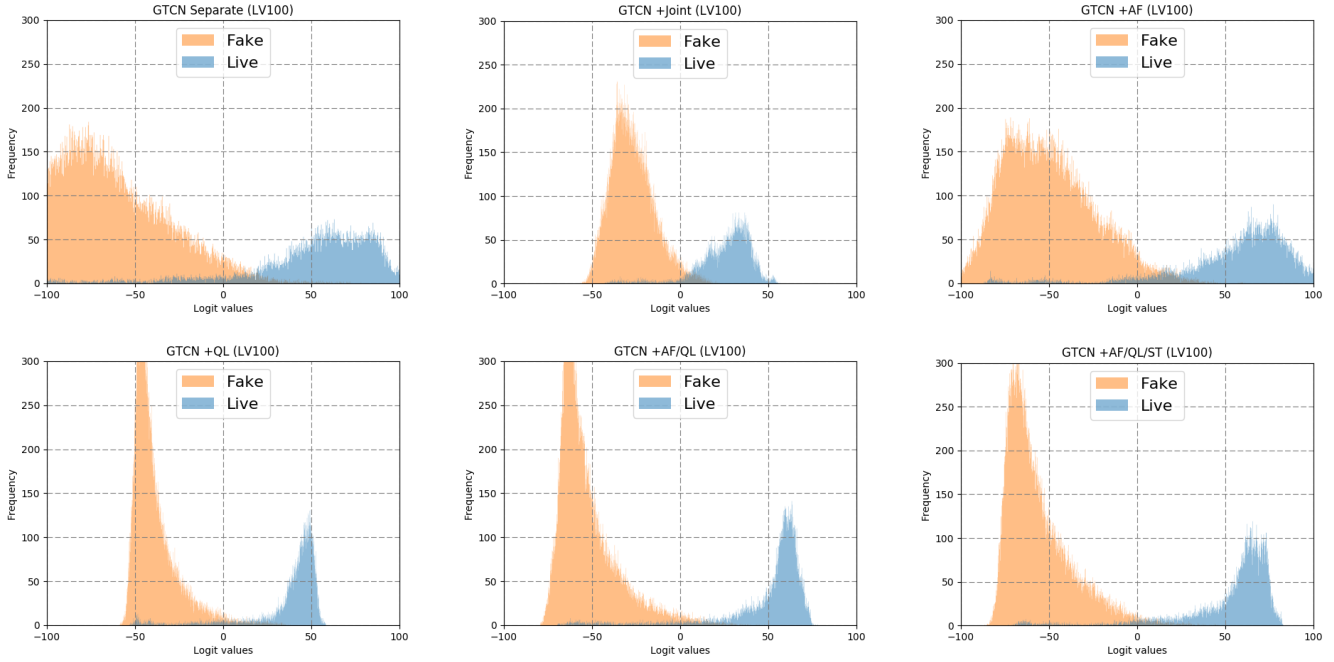


FIGURE 10: Histograms of logit values for liveness test dataset. The compared models are trained on LV100 dataset. (1) GTCN Separate (2) GTCN +Joint (3) GTCN +AF (4) GTCN +QL (5) GTCN +AF/QL (6) GTCN +AF/QL/ST

on binary classification. Figure 6 shows the example values taken by α and β for the AF method during training of the face liveness dataset. At the start of training, α and β were higher than 0.5, because real images are more confidently classified than generated images, while the values have almost converged to 0.5 at the end of training in the example. However, when AF, QL and ST are applied simultaneously, the performance of GTCNs is fairly improved, since ST contributes data diversity, QL increases fisher's criterion, and AF controls noisy images.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

FIGURE 11: Example from artist datasets. The first column shows Van Gogh's painting. The second, third, and fourth column show Monet, Cezanne, and Ukiyoe's one.

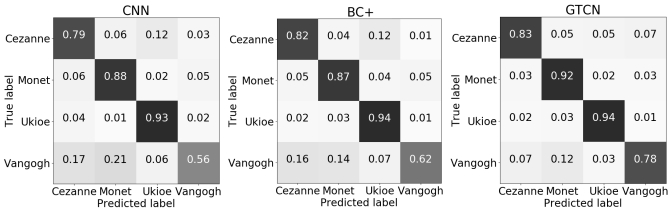


FIGURE 12: Comparison of confusion matrix for compared methods on the artist dataset

B. CHARACTERISTICS FOR WITHIN AND BETWEEN CLASSES

Properly defining what we mean by visual similarity is both critical and complicated. In quantitative terms, visual similarity of two classes can be measured by Fisher's criterion score J [12]. Fisher's score J in the experiment is defined as:

$$J = \frac{|\mu_A - \mu_B|^2}{\sigma_A^2 + \sigma_B^2}, \tag{20}$$

where μ_A and μ_B are the mean of the logits for distribution A and B respectively. σ_A and σ_B denote the standard deviations of the same logits. If the score is smaller than a specific threshold, we define the two classes as being similar and hard to discriminative. In qualitative terms, we assume that visual similarity can be decomposed into two main aspects: style and underlying structure. The comparison of Fisher's criterion scores for face liveness detection are shown in Figure 7. The proposed GTCN shows the best score, while other generative models show better scores than baseline CNNs. However, BC achieves lower score than the CNNs. In the ablation study results, utilizing all the proposed methods, Joint/AF/QL/ST, results in the best score. To clarify understanding for the scores, the histogram analysis of logit values for liveness test dataset are shown in Figure 9 and Figure 10. CNN256×256 seems to outperform CNN128×128. In case of BC, intra-class variance and mean difference between classes are both reduced. In VAE and GAN, intra-class variance and inter-class margin are both increased. The proposed GTCN enlarges margin well beween live class and fake class. In the ablation study, Joint, AF, QL, and ST show different characteristics to form the class distribution. Joint learning makes smaller variance within each class than Separate learning, while AF enlarges margin between classes. QL apparently reduces variance of each class, while it enlarges margin between the classes. Additionally, we provide the results of a t-distributed stochastic neighbor embedding t-SNE [23] visualization in Figure 8. Since GTCNs produce translated data from given real data, the feature space of augmented training samples are visualized. The feature space of the classifiers for test samples of the face liveness dataset is also visualized. The t-SNE of the GTCN with our proposed methods results in a sharper distinction between examples of similar classes, highlighting the fact that the learned representation is of higher quality.

TABLE 11: Configuration of experimental datasets and sub-sampled variants. AT* are reconfigured training datasets.

Artist	AT100	AT80	AT60	AT40	Test
Cezanne	291	231	173	115	292
Monet	597	477	358	238	586
Ukiyoe	412	329	247	164	413
Van Gogh	200	160	120	80	200

TABLE 12: Comparison of mean accuracy on multi-class artist dataset

Model	AT100	AT80	AT60	AT40
CNN	83.21	80.95	77.68	69.42
CNN-256	87.28	86.54	83.21	72.42
BC	83.08	82.41	78.95	69.62
BC^+	84.74	82.74	79.55	68.95
Semi-sup.	86.81	84.68	80.55	71.75
GTCN	**88.81**	**86.81**	**81.55**	**75.62**

TABLE 13: Ablation experimental results of the proposed methods on the artist datasets

Artist	CNN	Separate	+Joint	+AF	+AF/QL	+AF/QL/ST
Accuracy	83.21	82.61	85.14	85.88	86.21	**88.81**

C. MULTI-CLASS STYLE CLASSIFICATION

We extend the proposed method to a multi-class style classification setting. As an example of the cases with lack of data, artists usually draw a limited number of paintings in their lifetime, which makes artist classification a challenging problem. We constructed the artist dataset[4] that contains paintings from four painters as a multi-class classification experiment. The specific configuration is described in Table 11 and examples images of datasets are shown in Figure 11. Paintings have little overall structure, despite the styles chosen being similar, which creates a challenging problem of artist classification. This requires a slight modification of our proposed approach. $G_{AB}(\cdot)$ and $G_{BA}(\cdot)$ are trained as two translators across multiple-classes. A mini-batch size is set to four and we sample examples from the mini-batch belonging to different classes. Specifically, we use the same architecture as the one in the paper, which has only two generators, via a simple trick of setting minibach size to be four consisting of only one x_A, one $\tilde{x}_A$, one x_B, and one $\tilde{x}_B$ meaning that each minibatch handles only two classes at a time. To calculate the probability for multi-class k from a given test

[4]https://github.com/GTCN/datasets

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

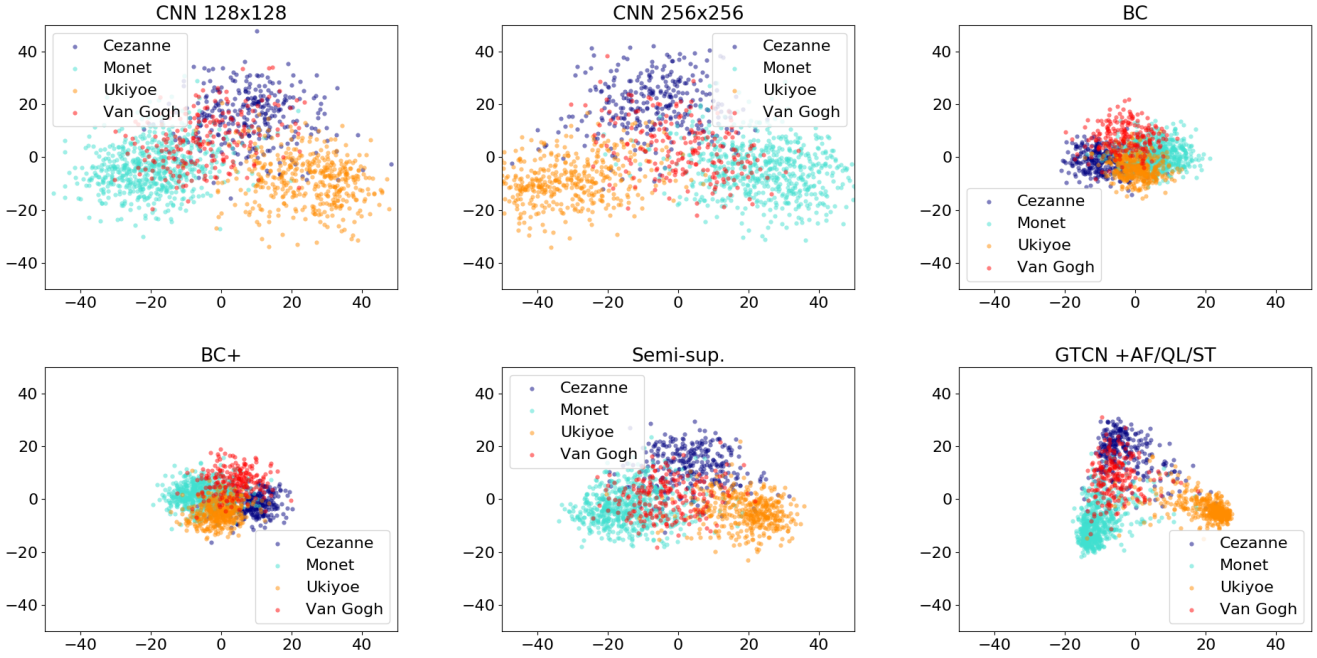


FIGURE 13: PCA analysis of logit values for artist test dataset. The compared models are trained on AT100 dataset. GTCN apparently enlarges margin between classes and reduce variance of within class. (1) CNN 128×128 (2) CNN 256×256 (3) BC 128×128 (4) BC+ 128×128 (5) Semi-sup. 128×128 (6) GTCN 128×128

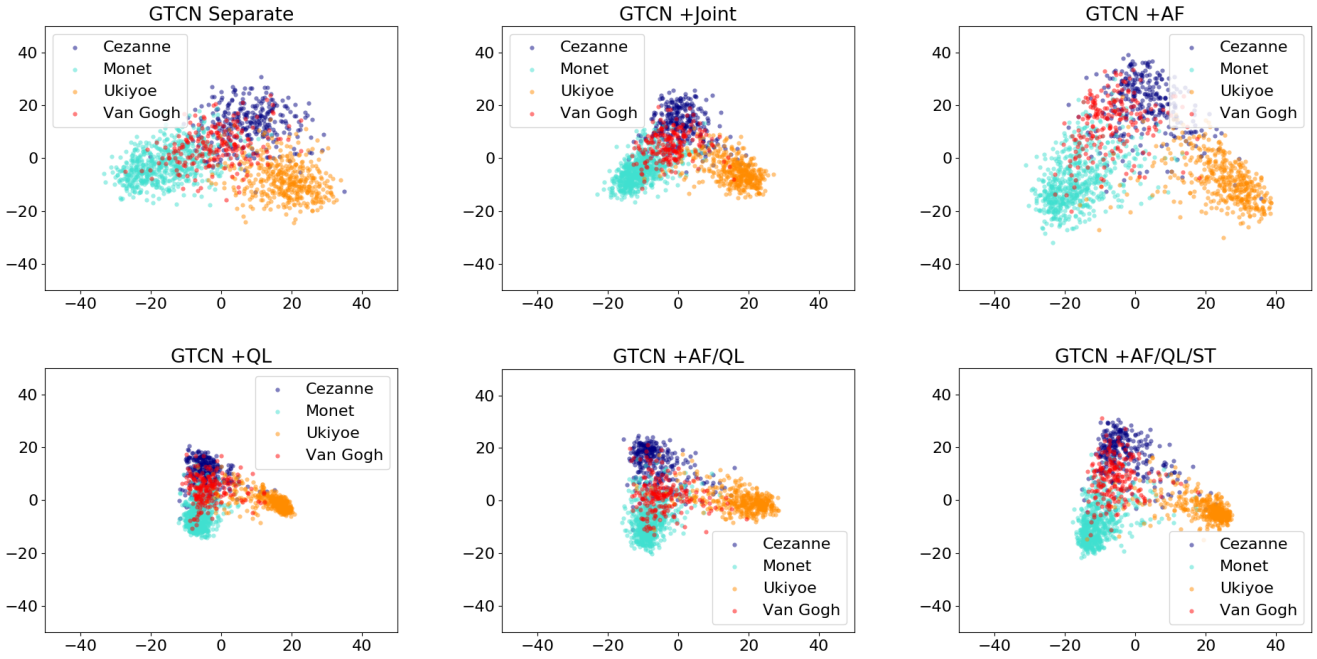


FIGURE 14: PCA analysis of logit values for artist test dataset. The compared models are trained on AT100 dataset. GTCN +AF/QL/ST enlarges margin between classes and reduce variance of within class for transforming to separable distribution of four classes. (1) GTCN Separate (2) GTCN +Joint (3) GTCN +AF (4) GTCN +QL (5) GTCN +AF/QL (6) GTCN +AF/QL/ST

image x, we employ the softmax output of the network. The function is defined by $P(y = i|x; \theta_C) = \frac{exp(FC_i)}{\sum_{j=1}^{k} exp(FC_j)}$, where $1 \leq i \leq k$ is a class-id, x is a given image, and k is the number of classes. We choose the class-id with the maximum probability among k probabilities as the recognized class-id of x. The hyperparameters for the quadruplet loss were set to: $\{\eta_a/\eta_b$=0.25, η_c=1.5$\}$ for the artist dataset. After training GTCNs, we additionally fine-tune the classifier with a cross-entropy loss for a few epochs. The method quickly stabilizes and improves the overall accuracy of the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

FIGURE 15: Qualitative comparison of data augmentation methods to train a classifier. First four rows are binary class examples of the face liveness and the dogs vs. cats dataset. Last three rows are multi-class examples of the artist dataset. In the examples, BC uses 0.33 and 0.66 as mixing ratio between two images.

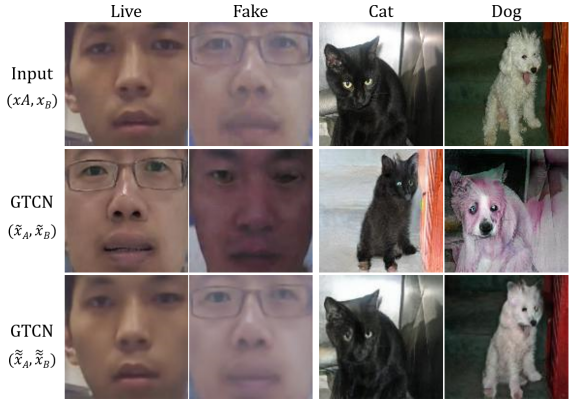


FIGURE 16: Example input images, translated images, and cyclic reconstructed images of face liveness and dogs vs. cats datasets

classifier in GTCNs for a multi-class problem. By fine-tuning, we assume weights related to noisy samples of multi-class may be truncated. To perform a fair comparison, we tried applying fine-tuning to other compared methods such as CNNs and BC/BC$^+$s. However, the fine-tuning is only useful for GTCN. Evaluation results are shown in Table 12 and GTCNs outperform compared methods. The accuracy for the four classes improved evenly for the GTCN as shown in Figure 12, because translated samples were used for training. We perform an ablation study, the results of which are shown in Table 13. Note that this is more like a simple modification for multi-class classification rather than carefully designed extension. However, it is noteworthy that because of joint training of GAN and classifier, the proposed method achieves 88.81% of accuracy for artist dataset significantly outperforming the baseline CNN (83.21%) or separate training of GAN and CNN (82.61%). In terms of characteristics for within and between classes, 2D PCA of logit values for artist test dataset are shown in Figure 13 and Figure 14, to analyze multi-class classification results. Characteristics of comparison with other methods and ablation study are very similar to binary classification cases. Especially, class Van Gogh has usually larger variance than other classes. We suppose the reason of the overlapping with other artist classes is that he affected or was affected to other artist styles. So, one of hardest classes is Van Gogh class in this problem. Even though Vangogh is a hard class, GTCN shows apparently improvement to CNN and BC+ in Figure 12. Overall results show the proposed GTCN enlarges margin between classes and reduce variance of within class.

D. VISUAL EXAMPLES OF TRAINED IMAGES

The GTCN uses progressively translated images as a part of the mini-batches. Note that the main purpose of the GTCN is not to generate realistic images but rather to improve the classifier's accuracy. Nonetheless, a visual inspection of translated/generated samples gives insights into why certain methods work better than others.

As shown in Figure 15, all of the compared models generate visually different images from given inputs to construct the augmented mini-batch. The images generated by the VAE are blurry and seem to wrongly interpolate across classes. BC attempts to mix up images, but seems to go pear-shaped at generating images that realistically belong to the target class, especially for the images of the artist dataset that display less overall structure. However, the GTCN attempts to borrow structure and shape information from samples of different classes while usually preserve texture and style, so the classifier can learn with sharper and more diverse training data. We visualize examples of input images, translated images, and cyclic reconstructed images for face and dogs vs. cats datasets in Figure 16. To show more examples of augmented training images, visually informative pairs were chosen in Figure 17.

VI. CONCLUSION

Our proposed model, GTCN, was trained with a novel joint learning approach. In this work, a generative translation model and a classifier are trained via three innovations: the augmented mini-batch technique, adaptive fade-in learning, and the quadruple loss. Since translators provide challenging

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3042302, IEEE Access

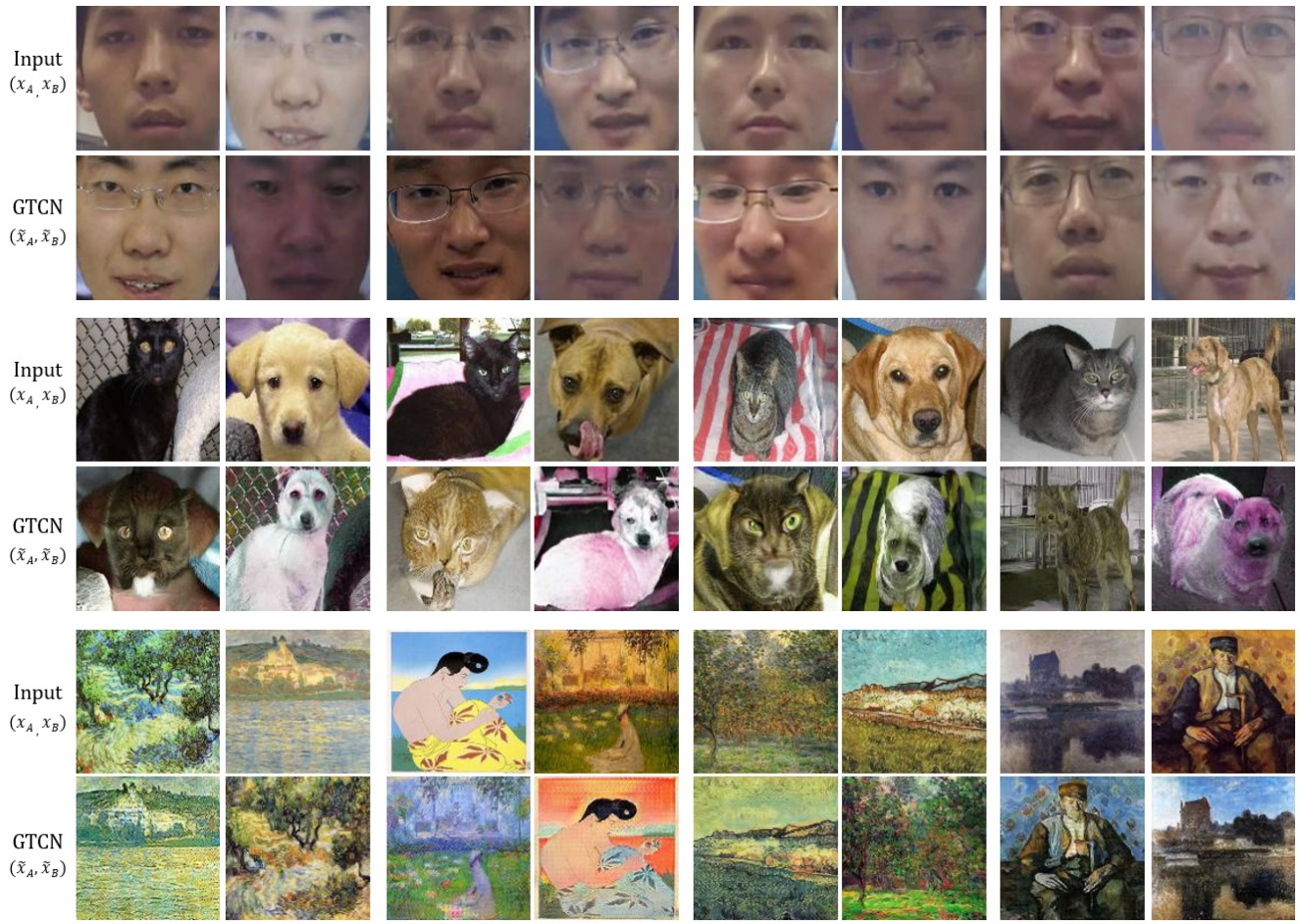


FIGURE 17: Examples of augmented training images for GTCN: (Top) Face liveness dataset (Middle) Dogs vs. cats dataset (Bottom) Artist dataset In the examples, translated samples attempt to borrow structure and shape information from samples of different classes while usually preserving texture and style. Even though the translated images are sometimes visually weird, the images in the inter-class space also contribute to improving of the classifier's accuracy for visually similar images in terms of shape and style.

data incrementally during training a classifier, accuracy can be improved by employing the augmented inter-class data. After the end of training, we perform inference using only the light-weight classifier of GTCN. Our method trained on a small subset of the whole dataset achieves a greater accuracy than the baselines trained on the full dataset. When training on the full dataset, we surpass comparable state-of-the-art methods despite using low resolution images and a smaller network architecture for our method. We believe our work can benefit classification tasks that suffer from visual similarity, diversity, and lack of data.

REFERENCES

[1] Y. Akbulut, A. Şengür, Ü. Budak, and S. Ekici. Deep learning based face liveness detection in videos. In Artificial Intelligence and Data Processing Symposium (IDAP), 2017 International, pages 1–4. IEEE, 2017.

[2] A. Almahairi, S. Rajeswar, A. Sordoni, P. Bachman, and A. Courville. Augmented cyclegan: Learning many-to-many mappings from unpaired data. arXiv preprint arXiv:1802.10151, 2018.

[3] A. Antoniou, A. Storkey, and H. Edwards. Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340, 2017.

[4] Y. Atoum, Y. Liu, A. Jourabloo, and X. Liu. Face anti-spoofing using patch and depth-based cnns. In Biometrics (IJCB), 2017 IEEE International Joint Conference on, pages 319–328. IEEE, 2017.

[5] W. Chen, X. Chen, J. Zhang, and K. Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), volume 2, 2017.

[6] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In Advances in Neural Information Processing Systems, pages 2172–2180, 2016.

[7] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. arXiv preprint arXiv:1711.09020, 2017.

[8] L. Chongxuan, T. Xu, J. Zhu, and B. Zhang. Triple generative adversarial nets. In Advances in Neural Information Processing Systems, pages 4091–4101, 2017.

[9] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath. Generative adversarial networks: An overview. IEEE Signal Processing Magazine, 35(1):53–65, 2018.

[10] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. R. Salakhutdinov. Good semi-supervised learning that requires a bad gan. In Advances in Neural Information Processing Systems, pages 6513–6523, 2017.

[11] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. arXiv preprint arXiv:1606.00704, 2016.

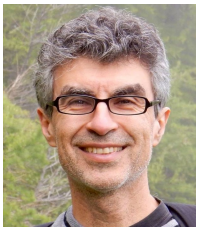
[12] R. A. Fisher. The use of multiple measurements in taxonomic problems. Annals of eugenics, 7(2):179–188, 1936.

[13] Z. Gan, L. Chen, W. Wang, Y. Pu, Y. Zhang, H. Liu, C. Li, and L. Carin. Triangle generative adversarial networks. In Advances in Neural Informa-

tion Processing Systems, pages 5253–5262, 2017.
[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.
[15] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. arXiv preprint arXiv:1804.04732, 2018.
[16] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. arXiv preprint, 2017.
[17] D. P. Kingma and M. Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.
[18] H.-Y. Lee, , H.-Y. Tseng, J.-B. Huang, M. K. Singh, and M.-H. Yang. Diverse image-to-image translation via disentangled representations. In European Conference on Computer Vision, 2018.
[19] H. Li, P. He, S. Wang, A. Rocha, X. Jiang, and A. C. Kot. Learning generalized deep feature representation for face anti-spoofing. IEEE Transactions on Information Forensics and Security, 13(10):2639–2652, 2018.
[20] L. Li, X. Feng, Z. Boulkenafet, Z. Xia, M. Li, and A. Hadid. An original face anti-spoofing approach using partial convolutional neural network. In 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA), pages 1–6. IEEE, 2016.
[21] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In Advances in Neural Information Processing Systems, pages 700–708, 2017.
[22] Y. Liu, A. Jourabloo, and X. Liu. Learning deep models for face anti-spoofing: Binary or auxiliary supervision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 389–398, 2018.
[23] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. Journal of Machine Learning Research, 9(Nov):2579–2605, 2008.
[24] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In Computer Vision (ICCV), 2017 IEEE International Conference on, pages 2813–2821. IEEE, 2017.
[25] M. Mirza and S. Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.
[26] D. T. Nguyen, T. D. Pham, N. R. Baek, and K. R. Park. Combining deep and handcrafted image features for presentation attack detection in face recognition systems using visible-light camera sensors. Sensors, 18(3):699, 2018.
[27] A. Odena. Semi-supervised learning with generative adversarial networks. arXiv preprint arXiv:1606.01583, 2016.
[28] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. arXiv preprint arXiv:1610.09585, 2016.
[29] A. R. Preston and H. Eichenbaum. Interplay of hippocampus and prefrontal cortex in memory. Current Biology, 23(17):R764–R773, 2013.
[30] A. Salazar, G. Safont, and L. Vergara. Surrogate techniques for testing fraud detection algorithms in credit card operations. In 2014 International Carnahan Conference on Security Technology (ICCST), pages 1–6. IEEE, 2014.
[31] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In Advances in Neural Information Processing Systems, pages 2234–2242, 2016.
[32] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 815–823, 2015.
[33] X. Song, X. Zhao, and T. Lin. Face spoofing detection by fusing binocular depth and spatial pyramid coding micro-texture features. arXiv preprint arXiv:1803.04722, 2018.
[34] J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. arXiv preprint arXiv:1511.06390, 2015.
[35] Y. Taigman, A. Polyak, and L. Wolf. Unsupervised cross-domain image generation. arXiv preprint arXiv:1611.02200, 2016.
[36] D. Tang, Z. Zhou, Y. Zhang, and K. Zhang. Face flashing: a secure liveness detection protocol based on light reflections. arXiv preprint arXiv:1801.01949, 2018.
[37] G. Tian, T. Mori, and Y. Okuda. Spoofing detection for embedded face recognition system using a low cost stereo camera. In Pattern Recognition (ICPR), 2016 23rd International Conference on, pages 1017–1022. IEEE, 2016.
[38] Y. Tokozume, Y. Ushiku, and T. Harada. Between-class learning for image classification. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
[39] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, volume 1, pages I–I. IEEE, 2001.
[40] J. Yang, Z. Lei, and S. Z. Li. Learn convolutional neural network for face anti-spoofing. arXiv preprint arXiv:1408.5601, 2014.
[41] Z. Zhang, J. Yan, S. Liu, Z. Lei, D. Yi, and S. Z. Li. A face antispoofing database with diverse attacks. In Biometrics (ICB), 2012 5th IAPR international conference on, pages 26–31. IEEE, 2012.
[42] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2223–2232, 2017.
[43] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. In Advances in Neural Information Processing Systems, pages 465–476, 2017.

BYUNGIN YOO has been a Research Staff Member at Samsung Advanced Institute of Technology (SAIT), Suwon, Korea, since 2005. He is currently a Principal Researcher and a Project Leader of Deep Image Processing Project in Computer Vision Lab. of SAIT. Prior to joining Samsung, he conducted research projects on Distributed Systems, Military Simulators, and Computer Antivirus Software from 1999 to 2005. He received his B.S. and M.S. degrees in Computer Science and Engineering from Dongguk University, Seoul, Korea, in 1997 and 1999, and the Ph.D. candidate at Statistical Inference and Information Theory lab in Korea Advanced Institute of Science and Technology (KAIST) under the supervision of Pr. Junmo Kim. He was a Visiting Researcher at Montreal Institute for Learning Algorithms, Canada in 2018 on a scholarship from Samsung. His research interests include computer vision and image processing based on exploiting pattern recognition and deep learning algorithms.

TRISTAN SYLVAIN obtained his engineering diploma (equiv. MSc.) at Ecole Polytechnique in 2013, and MSc. in Computer Science at the University of Oxford in 2014. He is currently a Ph.D. student in Deep Learning at the Mila lab in Montréal under the supervision of Pr. Yoshua Bengio and Pr. Devon Hjelm. His research interests include Computer Vision, Generalization and Generative models. He currently focuses on understanding the fundamental principles that help generalization in different computer vision problems including zero-shot learning.



YOSHUA BENGIO is recognized as one of the world's leading experts in artificial intelligence and a pioneer in deep learning. Since 1993, he has been a professor in the Department of Computer Science and Operational Research at the Université de Montréal. CIFAR's Learning in Machines & Brains Program Co-Director, he is also the founder and scientific director of Mila, the Quebec Artificial Intelligence Institute, the world's largest university-based research group in deep learning. In 2018, Yoshua Bengio ranked as the computer scientist with the most new citations worldwide, thanks to his many high-impact contributions. In 2019, he received the Killam Prize and the ACM A.M. Turing Award, the Nobel Prize of Computing, jointly with Geoffrey Hinton and Yann LeCun for conceptual and engineering breakthroughs that have made deep neural networks a critical component of computing. Concerned about the social impacts of this new technology, he actively contributed to the development of the Montreal Declaration for Responsible Development of Artificial Intelligence.

JUNMO KIM received the B.S. degree from Seoul National University, Seoul, South Korea, in 1998, and the M.S. and Ph.D. degrees from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2000 and 2005, respectively. From 2005 to 2009, he was with the Samsung Advanced Institute of Technology (SAIT), South Korea, as a Research Staff Member. He joined the faculty of KAIST, in 2009, where he is currently as an Associate Professor of electrical engineering. His research interests are in image processing, computer vision, statistical signal processing, machine learning, and information theory.