Automatic synthetic document image generation using generative adversarial networks: application in mobile-captured document analysis

Quang Anh BUI
PurchEase company
Paris, France
Email: anh@purchease.com

David MOLLARD
PurchEase company
Paris, France
Email: david@purchease.com

Salvatore TABBONE
LORIA UMR 5073, Université de Lorraine
Nancy, France
Email: tabbone@loria.fr

***Abstract*—In this paper, we propose a method using Generative Adversarial Networks for automatically synthesizing document images that are similar to real printed documents captured by mobile phone's camera in unconstrained environment. We focus on the simulation of image defects for unconstrained mobile image acquisition procedure (non-uniform illumination, defocusing, optical and mechanical deformations, vibrations, noise in electronic components,...). Our approach is proven to be low-cost as it only requires a collection of real document images without any annotation. Experimental results show the effectiveness of our approach to improve OCR (Optical Character Recognition) recognition rate in a mobile-captured document images framework. Although in this paper, we focus on modern printed document images, our proposed approach could be extended to another type of documents, including historical one.**

***Keywords*—synthetic document image generation, Generative Adversarial Networks, mobile-captured documents.**

I. Introduction

In recent years, supervised learning methods have shown great effectiveness in many fields, including document analysis and recognition. But, such kind of methods are still limited by a poor generalization or over-fitting due to the insufficient quantity of ground-truth data. Studies ([1], [2]) showed that performances of analysis and recognition systems could be improved with an introduction of a big amount of real annotated data. However, typically, it is hard to obtain due to the high cost of human resource for data annotation, specifically, in the case of historical documents. To overcome this obstacle, data augmentation methods, which consist of introducing computer generated data with high variability and controlled ground-truth in addition to real annotated data, have been proposed to reduce over-fit and boost generalization capability of supervised learning systems. Such methods usually aim at reducing the divergence / distance between distribution of real data and the distribution of generated data. Or, in other words, data augmentation methods usually tend to generate realistic data.

In the field of document analysis and recognition, in the last two decades, several data augmentation methods have been proposed. They usually aim at generating (automatically or semi-automatically) realistic synthetic document images, in term of content (writings font, writings style, representations, etc.) and/or in term of defects and distortions caused by image acquisition procedure. We found several data augmentation methods in the literature. Some introduce transformations (random translations, rotations, flips, noises, etc.) to the real annotated data-set [3] or generate content images applying transformations to simulate physical degradations and distortions [4], [5]. Other focus on handwriting documents rearranging elements extracted from real images [6] or generating binary document images whose writings are similar to real ones [7]. Among recent works about data augmentation, we could mention DocCreator [5], an iterative software that allows creating realistic document images, using several techniques to automatically/semi-automatically create content images (that are still perfectible) and applying several degradation models like ink degradation, phantom character, adaptive blur, bleed-through, 3D paper deformation. This tool employs manual works (human manipulations) and additional data collection (background paper collection, 3D model collection, etc.) to make generated document images highly realistic. However, this solution still have high cost due to human resources and additional data collection.

Without manual works and additional data collection, the realistic level of synthetic document images generated by most of the data augmentation methods is still low. This is because most data augmentation methods usually use hand-engineered models that include a very limited set of known variables. So they usually have limited capability of approximating the distribution of real document images which is very complex. To overcome these drawbacks, we propose an approach to automatically generate synthetic printed document image which produce highly realistic document images with minimal cost of human resource and data collection.

As we mentioned above, the key in data augmentation methods is to reduce divergence between the distribution of real document images and the distribution of synthetic document images. We realized that, Generative Adversarial Networks, generative models which are effective for reducing divergence/distance between distributions of generated samples

and target distributions, could be used in synthetic document image generation. Synthetic content document images could be addressed in two ways: simulation of document content (text content, text font/writing style) or simulation of image defects (such as: non-uniform illumination, defocusing, ...) caused by image acquisition procedure. In this paper, we place ourselves in the context of applications for camera-based document analysis, specifically, printed documents captured by mobile phone's camera in uncontrolled environment. In this context, physical degradation of document and image acquisition distortions are usually the most important elements to concern. So, we focus on the on simulation of image defects caused by mobile camera-based image acquisition procedure. In section 2, we present an overview of Generative Adversarial Networks models and we discuss its application for synthetic documents image generation. Then, we describe our proposed approach and we discuss its applications (section 3). Section 4 provides an experimental evaluation protocol to evaluate the quality of generated images and their effectiveness in the improvement of the document image recognition system. Section 5 is devoted to our conclusion and gives some perspectives to our works.

II. Overview of Generative Adversarial Networks models (GANs)

Generative Adversarial Networks (GANs) are a class of generative models that have been proven to be effective for reducing divergence / distance between distributions of generated samples and target distribution. The first GAN model was introduced by Goodfellow et al. [8], which contains 2 neural networks: the Generator (G) and the Discriminator (D). The basic idea is to simultaneously train these two neural networks to reach their objectives. While the Discriminator has the objective of discriminating between real samples and fake samples, the Generator has the objective of generating fake samples that are close to real samples by fooling the Discriminator. More precisely, let:

- p_{data} be the real data distribution and p_z a noise prior distribution,
- G be the generator, a differentiable function represented by a neural network model with parameters θ_G which transforms samples z^i (sampled from p_z) to fake samples $G(z^i)$. The distribution of generated fake samples is p_g,
- D be the discriminator, a differentiable function represented by a neural network model with parameters θ_D which returns a single output scalar for a given input data. $D(x)$ represents the probability that x came from the real data distribution p_{data} rather than p_g.

The training method is presented in algorithm 1. Authors of [8] proved that algorithm 1 converges and p_g converges to p_{data}.

After the introduction of the vanilla GAN, many variations have been introduced: CGAN [9], InfoGAN[10], WGAN [11], DualGAN [12], CycleGAN [13], etc. Each method is designated for a specific task. Among them, there are several GAN methods that could be used for synthetic document image generation problem, such as GAN-CLS [14], DualGAN [12], MUNIT [15].

repeat
  for *k steps* **do**
    - Sample m samples $\{z^1, z^2, ..., z^m\}$ from p_z;
    - Sample m examples $\{x^1, x^2, ..., x^m\}$ from real data distribution p_{data};
    - Update parameters θ_D of discriminator D by ascending its gradient : $\nabla_{\theta_D} \frac{1}{m} \sum_{i=1}^{m} [log(D(x^i)) + log(1 - D(G(z^i)))]$
  end
  - Sample m samples $z^1, z^2, ..., z^m$ from p_z;
  - Update parameters θ_G of generator G by descending its gradient : $\nabla_{\theta_G} \frac{1}{m} \sum_{i=1}^{m} log(1 - D(G(z^i)))$
until *convergence*;

Algorithm 1: Vanilla GAN algorithm

GAN-CLS [14] is a method for generating images from text description. Its model consists of 2 neural networks: the Generator G and the Discriminator D. Generator G has the role of generating synthetic images conditioned on text description $\varphi(t)$. Discriminator D has the role of discriminating between generated images and real images, conditioned on text description $\varphi(t)$. Typically, GAN-CLS is used in the generation of natural images: images of birds, flowers, dish plates, etc. With the principles of GAN-CLS, this method could be used in the generation of synthetic document images. However, to the best of our knowledge, no synthetic document generation method using GAN-CLS has been proposed, probably because using GAN-CLS requires a big amount of annotation data (set of pairs: real image x and it's text description h). This limitation sometimes make synthetic document generation based on GAN-CLS not profitable.

Two other methods (DualGAN[12], MUNIT [15]) are designed for unsupervised image domain translation. They requires minimal amount of annotation data, and thus are suitable in our context. In the following subsections, we give a more details about these two methods.

A. DualGAN: Unsupervised Dual Learning for Image to Image Translation

DualGAN [12] is a method for unsupervised image domain translation, for example, from photo images to sketch images, day scene images to nigh scene images (see [12]). A similar method to DualGAN is CycleGAN [13]. DualGAN and CycleGAN both aim for image domain translation without requiring paired training data (i.e. a photo image and a sketch image of the same person) to bridge the two image domains. Its model consists of two Generators: G_A, G_B and two Discriminators: D_A, D_B. By simultaneously parameters of generators and discriminators, we can have a system that can transform an image of a domain to an realistic image of another domain. More precisely, let:

- U be an image domain, for example, scenes images taken in summer. V is another image domain, for example, scenes images taken in winter,
- G_A be a generator, a differentiable function represented by a neural network model with parameters θ_A, which transforms an image of domain U to domain V,
- G_B be another generator, a differentiable function represented by a neural network model with parameters θ_B, which transforms an image of domain V to domain U,
- u be a real image of domain U and v be a real image of domain V. $G_A(u) \in V$ is a fake image, generated from u by G_A. $G_B(v) \in U$ is a fake image, generated from v by G_B, $G_A(G_B(v)) \in V$ is a fake image, generated from $G_B(v)$. $G_B(G_A(u)) \in U$ is a fake image, generated from $G_A(u)$. As we can see, $G_B(G_A(u))$ is the reconstruction of u through generators G_A and G_B. It is expected to be the same as u. $G_A(G_B(v))$ is the reconstruction of v through generators G_B and G_A. It is expected to be the same as v,
- D_A be a discriminator, a differentiable function represented by a neural network model with parameters ω_A, discriminates real images and fake images of domain V,
- D_B be a discriminator, a differentiable function represented by a neural network model with parameters ω_B, discriminates real images and fake images of domain U,
- $l_{dA}(u, v) = D_A(G_A(u)) - D_A(v)$ is the loss function of discriminator D_A. $l_{dB}(u, v) = D_B(G_B(v)) - D_B(u)$ is the loss function of discriminator D_B,
- $l_g(u, v) = \lambda_U ||u - G_B(G_A(u))|| + \lambda_V ||v - G_A(G_B(v))|| - D_B(G_B(v)) - D_A(G_A(u))$ is the loss function of generators G_A and G_B. λ_U and λ_V are two constant parameters.

repeat
  for *k steps* **do**
    - Sample images $u_1, u_2, ..., u_m \in U; v_1, v_2, ..., v_m \in V$;
    - Update parameters ω_A of discriminators D_A to minimize: $\frac{1}{m} \sum_{i=1}^{m} l_{dA}(u_k, v_k)$;
    - Update parameters ω_B of discriminators D_B to minimize : $\frac{1}{m} \sum_{i=1}^{m} l_{dB}(u_k, v_k)$;
    - $clip(\omega_A, -c, c); clip(\omega_B, -c, c)$ where c is the fixed clipping parameter
  end
  - Sample images $u_1, u_2, ..., u_m \in U; v_1, v_2, ..., v_m \in V$;
  - Update parameters θ_A, θ_B of generators G_A, G_B to minimize : $\frac{1}{m} \sum_{i=1}^{m} l_g(u_k, v_k)$;
until *convergence*;

Algorithm 2: Dual GAN algorithm

The training method is presented in algorithm 2. More details about neural network structures of generators and discriminators are given in [12]. To increase the diversity of generated images, authors introduced noises in form of dropout [16] and applied to several layers of generators.

B. MUNIT: Multimodal Unsupervised Image-to-Image Translation

MUNIT [15] is another method for unsupervised image domain translation. It is aimed to generate high diversity output images from a given source image. The authors assumed that each image is generated from a content latent code c that is shared by both domains, and a style latent code s that is specific to each domain. For example, an image x_1 from domain 1 is generated from content code c_1 and style code $s_1 \in$ domain 1. If we switch the style code $s1$ with a style code $s2 \in$ domain 2, we can generate another image $x_{1\to2} \in$ domain 2 which is the transformation of image $x_1 \in$ domain 1. The goal is then to learn encoders that encode image to latent codes and generators that generate images from latent codes. More precisely, suppose that :

- x_1 is a real image of domain 1, x_2 is a real image of domain 2,
- E_1, a differentiable function represented by a neural network model with parameters θ_{E_1}, is the encoder that encodes images of domain 1 to latent codes. c_1 and s_1 are respectively content and style latent codes generated from x_1 by E_1: $(c_1, s_1) = E_1(x_1)$,
- G_1, a differentiable function represented by a neural network model with parameters θ_{G_1}, is the generator that generate domain 1 images from latent codes. $\hat{x_1}$ is the image generated from latent codes (c_1, s_1): $\hat{x_1} = G_1(c_1, s_1)$. $\hat{x_1}$ is the reconstruction of x_1 and it is expected to be the same as x_1,
- E_2, a differentiable function represented by a neural network model with parameters θ_{E_2}, is the encoder that encodes images of domain 2 to latent codes. c_2 and s_2 are respectively content and style latent codes generated from x_2 by E_2: $(c_2, s_2) = E_2(x_2)$,
- G_2, a differentiable function represented by a neural network model with parameters θ_{G_2}, is the generator that generate domain 2 images from latent codes. $\hat{x_2}$ is the image generated from latent codes (c_2, s_2): $\hat{x_2} = G_2(c_2, s_2)$. $\hat{x_2}$ is the reconstruction of x_2 and it is expected to be the same as x_2,
- $\tilde{s_1}$ is a style latent code, drawn from a prior distribution $q(s_1) \sim N(0, I)$. $x_{2\to1} = G_1(c_2, \tilde{s_1})$ is the image transformed from domain 2 to domain 1. Image $x_{2\to1}$ is expected to be as similar as real images of domain 1,
- $\hat{c_2}$ and $\hat{s_1}$ are latent codes encoded from $x_{2\to1}$ by E_1 : $(\hat{c_2}, \hat{s_1}) = E_1(x_{2\to1})$. $\hat{c_2}$ is expected to be the same as c_2 and $\hat{s_1}$ is expected to be the same as $\tilde{s_1}$,
- $\tilde{s_2}$ is a style latent code, drawn from a prior distribution $q(s_2) \sim N(0, I)$. $x_{1\to2} = G_2(c_1, \tilde{s_2})$ is the image transformed from domain 1 to domain 2. Image $x_{1\to2}$ is expected to be as similar as real images of domain 2,
- $\hat{c_1}$ and $\hat{s_2}$ are latent codes encoded from $x_{1\to2}$ by E_2 : $(\hat{c_1}, \hat{s_2}) = E_2(x_{1\to2})$. $\hat{c_1}$ is expected to be the same as c_1 and $\hat{s_2}$ is expected to be the same as $\tilde{s_2}$,
- D_1, a differentiable function represented by a neural network model with parameters ω_{D_1}, is the discriminator

that discriminates real images of domain 1 and fake images that are generated by generator G_1,
- D_2, a differentiable function represented by a neural network model with parameters ω_{D_2}, is the discriminator that discriminates real images of domain 2 and fake images that are generated by generator G_2,
- $p(x_1)$ and $p(x_2)$ are respectively distributions of real images of domain 1 and domain 2,
- $p(c_1)$ and $p(c_2)$ are respectively distributions of content latent codes generated from real images of domain 1 and domain 2 by encoders E_1 and E_2,
- $p(s_1)$ and $p(s_2)$ are respectively distributions of style latent codes generated from real images of domain 1 and domain 2 by encoders E_1 and E_2.

Let the image reconstruction losses be:
- $L^1_{recon} = \mathbb{E}_{x_1 \sim p(x_1)}[\|G_1(c_1, s_1) - x_1\|_1]$
- $L^2_{recon} = \mathbb{E}_{x_2 \sim p(x_2)}[\|G_2(c_2, s_2) - x_2\|_1]$

Let the latent reconstruction losses be:
- $L^{c_1}_{recon} = \mathbb{E}_{c_1 \sim p(c_1), \tilde{s_2} \sim q(s_2)}[\|\hat{c_1} - c_1\|_1]$
- $L^{s_2}_{recon} = \mathbb{E}_{c_1 \sim p(c_1), \tilde{s_2} \sim q(s_2)}[\|\hat{s_2} - \tilde{s_2}\|_1]$
- $L^{c_2}_{recon} = \mathbb{E}_{c_2 \sim p(c_2), \tilde{s_1} \sim q(s_1)}[\|\hat{c_2} - c_2\|_1]$
- $L^{s_1}_{recon} = \mathbb{E}_{c_2 \sim p(c_2), \tilde{s_1} \sim q(s_1)}[\|\hat{s_1} - \tilde{s_1}\|_1]$

Let the adversarial losses be:
- $L^{x_1}_{GAN} = \mathbb{E}_{c_2 \sim p(c_2), \tilde{s_1} \sim q(s_1)}[log(1 - D_1(x_{2\to1}))] + \mathbb{E}_{x_1 \sim p(x_1)}[log(D_1(x_1))]$
- $L^{x_2}_{GAN} = \mathbb{E}_{c_1 \sim p(c_1), \tilde{s_2} \sim q(s_2)}[log(1 - D_2(x_{1\to2}))] + \mathbb{E}_{x_2 \sim p(x_2)}[log(D_2(x_2))]$.

The objective is to find parameters $\theta_{E_1}, \theta_{E_2}, \theta_{G_1}, \theta_{G_2}, \omega_{D_1}, \omega_{D_2}$ using stochastic gradient descent by reducing the following loss:

$L_{final} = L^{x_1}_{GAN} + L^{x_2}_{GAN} + \lambda_x(L^{x_1}_{recon} + L^{x_2}_{recon}) + \lambda_c(L^{c_1}_{recon} + L^{c_2}_{recon}) + \lambda_s(L^{s_1}_{recon} + L^{s_2}_{recon})$

where $\lambda_x, \lambda_c, \lambda_s$ are predefined parameters that control the importance of reconstruction terms.

Authors of [15] proved that, when optimal is reached, we have $p(c_1) = p(c_2), p(s_1) = q(s_1), p(s_2) = q(s_2)$. This suggest the content space becomes domain invariant and style space becomes domain specific. Then, it allows domain transformation while retaining image's content (see [15] for more details about neural network structures of encoders, generators and discriminators).

III. Proposed method

We remarked that, real mobile captured document images can be considered as a combination of two sources: content (clear, undistorted binary text image) and effects/distortions. So, we can consider the problematic of synthetic document image generation as a problem of image domain translation: clear, undistorted binary text images to distorted text images. We then use unsupervised image domain translation methods (discussed in the above section) to resolve this problem. There are 2 main advantages by using this: 1) Our generation system have low cost as it requires only a collection of real images (without any annotation on these images). 2) Besides from generating realistic images by transforming from binary images to real images, we can perform image binarization by transforming real images to binary image. (The later advantage is not discussed further in this paper. This idea and must be investigated in future works.)

More precisely, our proposed method has two modules: content image generation and realistic image generation by transformation from content image.

A. Content image generation

Content images are binary images (black text on white background), generated by using a text drawing software (in our case Cairo Graphics [17]) with a various type of fonts (Arial, Times New Roman, Courrier, Monaco, etc.). Drawn text are arranged by lines. Depending on the application, text font size and/or line space could be fixed or randomly chosen. After this step, a content binary image are created. This image is perfectible and must be transformed to a realistic image by the next module described in the following subsection.

B. Realistic image generation by transformation from content image

We transforms content binary images to realistic images by stimulating physical degradation and image acquisition distortions, using GAN-based unsupervised image domain translation methods. We propose two configurations:

- First configuration: M-DualGAN. We prepare two training sets: content binary images training set (generated by the first module described above) and real mobile captured document images training set. We base on DualGAN method for the image transformation problem with following modification: to increase the diversity of generated realistic images, we make a modification on the image reconstruction loss term (see section previous section). Instead of directly comparing 2 images of the real images domain (by calculating $||u - G_B(G_A(u))||$), we compare images in feature space by calculating $||f(u) - f(G_B(G_A(u)))||$ where $f(u)$ is a feature extraction function, represented by a neural network model with pre-trained parameters. The neural network model for $f(u)$ is the CNN part of the CRNN network [18] which is usually used for OCR.
- Second configuration: M-MUNIT. Following the same protocol as in the first configuration with two training sets (i.e two training sets: content binary images and real mobile captured document images) , we propose several modifications for MUNIT.

  * Firstly, to accelerate the training process, specifically the matching of the encoded style distributions $p(s_1)$ and $p(s_2)$ to their Gaussian priors $q(s_1)$ and $q_(s_2)$, we introduce two style latent code discriminators: D^s_1 and D^s_2. These discriminators distinguish style codes generated by encoders and style codes sampled from prior distributions $q(s_1)$ and $q(s_2)$. Then, we introduce the following losses added to the final loss:
  - $L^{s_1}_{GAN} = \mathbb{E}_{s_1 \sim p(s_1)}[log(1 - D^s_1(s_1))] + \mathbb{E}_{\tilde{s_1} \sim q(s_1)}[log(D^s_1(\tilde{s_1}))]$,

- $L^{s_2}_{GAN} = \mathbb{E}_{s_2 \sim p(s_2)}[log(1 - D^s_2(s_2))] + \mathbb{E}_{\tilde{s_2} \sim q(s_2)}[log(D^s_2(\tilde{s_2}))]$.

* We replace the GAN loss in the original paper by WGAN loss with gradient penalty (WGAN-GP) [19] which measure the distance between two distributions using Wasserstein distance [20]. In order to force the Lipschitz constrain on the discriminator, authors in [19] propose a gradient penalty term to the loss. With WGAN-GP, we can improve the stability of learning, get rid of mode-collapse and diminished gradient problems. The final loss becomes:

$L_{final} = L^{x_1}_{WGAN-GP} + L^{x_2}_{WGAN-GP} + \lambda_{sGAN}(L^{s_1}_{GAN} + L^{s_2}_{GAN}) + \lambda_x(L^{x_1}_{recon} + L^{x_2}_{recon}) + \lambda_c(L^{c_1}_{recon} + L^{c_2}_{recon}) + \lambda_s(L^{s_1}_{recon} + L^{s_2}_{recon})$,

where $\lambda_{sGAN}, \lambda_x, \lambda_c, \lambda_s$ are predefined parameters controlling the importance of the corresponding terms.

* As discussed in section I, we only want to stimulate physical degradations and image acquisition distortions. We don't want to transform high level styles such as text font or writing styles. So, we want to retain low-level information in the content encoder. Then, we propose a new network structure as follow:

- Content encoder: cin_f3k32s2, cin_f3k64s2, resin_f3k64, resin_f3k64,
- Style encoder: c_f3k8s2, c_f3k16s2, c_f3k32s2, c_f3k64s2,glob_avg_pool, mlp_8,
- Decoder: res_adain_f3k64, deconv_ln_relu_f3k32s2, deconv_tanh_f3k3s2,
- Discriminator: c_f3k8s2, c_f3k16s2, c_f3k32s2, c_f3k32s2, mlp_1,
- Style code discriminator: mlp_relu_4, mlp_1

where $cin\_f3k32s2$ denotes a convolution operation with receptive field 3, stride 2 and output channels 32, followed by Instance Normalization [21] and ReLU activation [22]. $resin\_f3k64s1$ is a Residual block with Instance Normalization, receptive field 3 and output channel 64. c_f3k8s2 describes a convolution operation with receptive field 3, stride 2 and output channels 8, followed by LeakyReLU activation [23]. mlp_8 denotes a linear operation with output neurals 8 and res_adain_f3k64 a Residual block with Adaptive Instance Normalization [21], receptive field 3 and output channel 64. $deconv\_ln\_relu\_f3k32s2$ defines a deconvolution operation [24] with receptive field 3, stride 2 and output channels 32, followed by Layer Normalization [25] and ReLU activation. $glob\_avg\_pool$ is the global average pooling operation.

IV. Experiments

A. Synthetic document image generation results

In this section, we present implementation details about our proposed synthetic document image generation method and show its results.

First, we collect a set of 50000 real images of supermarket receipts that are captured by various users using mobile phone cameras in unconstrained conditions. Then, each image is cropped at random position with the windows size of 512x512 pixels. Figure 1.A shows examples of real images. Next, we generate a set of 50000 binary images of size 512x512, using method described in section III-A (see Figure 1.B). We use a collection of 50 text fonts (Arial, Times New Roman, Courrier, Monaco, etc.) and text size randomly selected for each image. The text in each line is formed by a sequence of 3 – 10 words that are randomly selected from a collection of about 2 million words extracted from super market receipts.

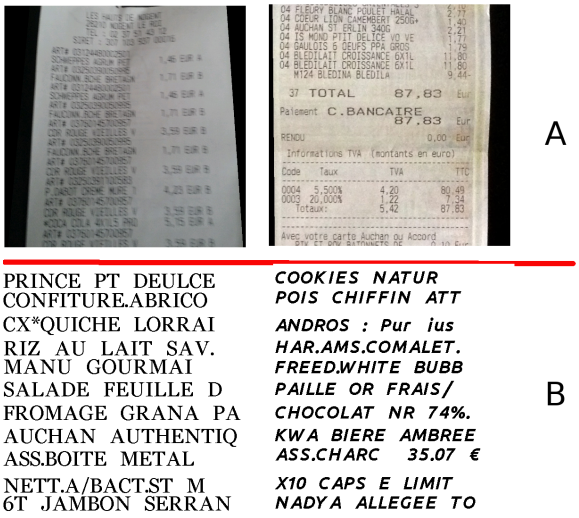

Fig. 1. Training images. First row (A): Real images of supermarket receipts captured by mobile phone cameras. Second row (B): Binary images.

We use these two collections of data for training the system (as described in section III-B) for the configurations: M-DualGAN and M-MUNIT. For M-MUNIT, we fixed parameters $\lambda_{sGAN} = 10, \lambda_x = 20, \lambda_c = 5, \lambda_s = 10$. Figure 2 shows the result of our synthetic printed document image generation. We trained GAN models on a single desktop machine with following configuration: CPU: Intel core i7 7700k, Memory: 32GB RAM, GPU: Nvidia Geforce GTX 1080 Ti with 11GB RAM. It takes 5 days to train the model that generate these images. As we can see, the proposed generation system has successfully simulates defects caused by mobile camera acquisition procedure (see Figure 2 rows B and C), such as non uniform illumination, high exposure defect, out of focus (blur) effects, ink degradation, phantom character or incorrect white-balance.

B. Correctness of synthetic image's ground-truth

In this section, we provide an experimental protocol to evaluate the correctness of the ground-truth of synthetic images generated by our proposed method. Firstly, 100000 binary line images are generated by using the method presented in section III-A. We use a collection of 50 font types. For each line image, a font size is randomly chosen. We generate synthetic line images using 2 configurations (M-DualGAN, M-MUNIT) introduced in section III-B with the implementation

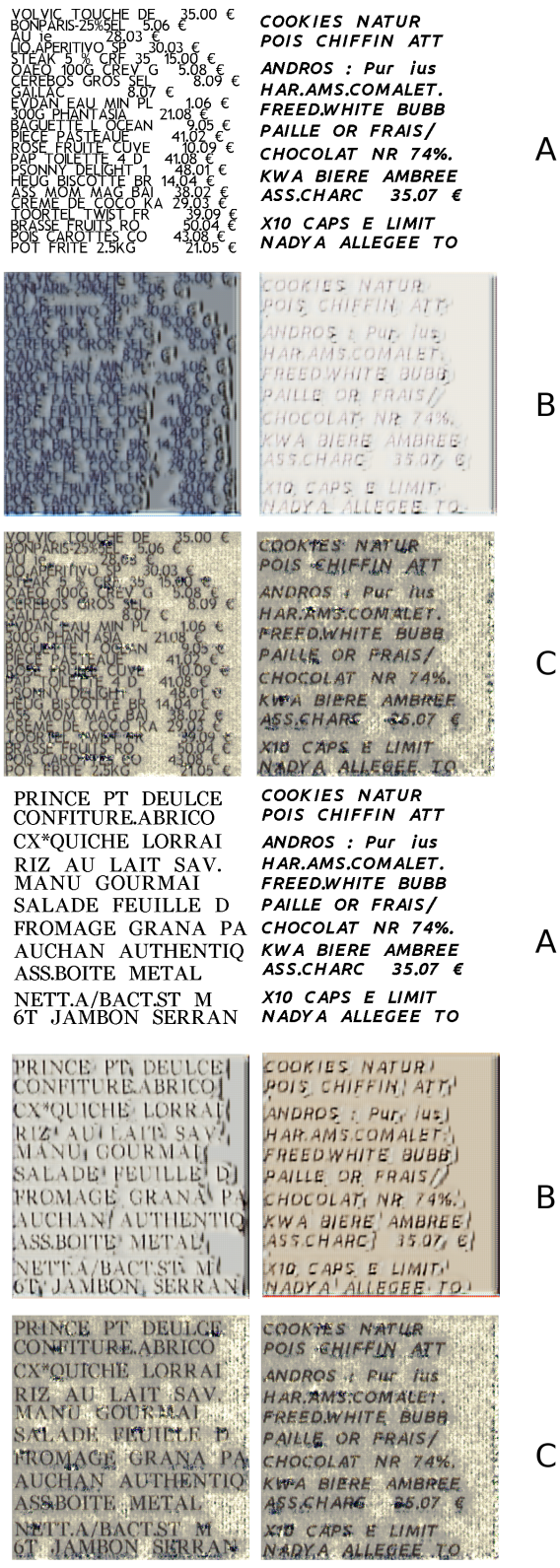

Fig. 2. Generated images. Rows (A): Binary images. Rows (B): Synthetic images transformed from binary images in A by M-MUNIT. Rows (C): Synthetic images transformed from binary images in A by M-DualGAN.

TABLE I
CORRECTNESS OF SYNTHETIC IMAGE'S GROUND-TRUTH

Configuration	Correctness measure
M-DualGAN	0.99
M-MUNIT	0.91

presented in section IV-A. Then, we have 2 line image sets: synthetic images generated by M-DualGAN and synthetic images generated by M-MUNIT. We hire Amazon MTurk workers to type the text they see in each image. An image is annotated by 2 workers. A line image is considered correct if one of annotations match the text that is used to generate this line image. The correctness measure is defined as the ratio of correct line image over total line images (see table I).

These result shows us that, the ground-truth of images generated by our method (which is the text used for generate binary images) are very good but we miss some line images because some characters are degraded and sometimes vanished due to the physical degradations which are applied on the whole image whatever the text position.

C. Application on Mobile-captured document analysis

In this section, we provide experiments to evaluate the effectiveness of our proposed synthetic image generation method for boosting performances of Optical Character Recognition (OCR) system. Our experiments are specified for mobile-captured documents representing supermarket receipts.

We define six datasets:

- Dataset 1: 100000 line images extracted from real images of supermarket receipts captured by various users using mobile phone cameras in unconstrained conditions. Receipts in this collection come from the same retailer, so that they have the same text font,
- Dataset 2: 200000 line images extracted from real images of supermarket receipts that come from 5 different retailers (different from retailer of receipts in dataset 1), captured by various users using mobile phone cameras in unconstrained conditions. Receipts in this collection have 5 different text fonts (different from text font of documents in dataset 1). This collection is divided into 2 subsets: 100000 line images are used for training, the others are used for testing.
  In the above 2 data sets, line images are extracted by the text line segmentation method presented in [26]. Ground-truth annotations are obtained by Amazon MTurk workers.
- Dataset 3: Simple fake line images collection. Firstly, 100000 binary line images are generated using the method presented in section III-A. We use a collection of 50 font types. For each line image, a font size is randomly chosen. After creation of the binary images, noises such as gaussian noise, salt and pepper noise are introduced,
- Dataset 4: 100000 synthetic line images generated by M-DualGAN configuration introduced in section III-B. Binary line images are generated by using the method

TABLE II
EXPERIMENTS ON OCR'S PERFORMANCE

Test	Training images from	Test images from	Recognition rate
1	Data set 2	Data set 1	0.822
2	Dataset 2 + Dataset 3	Dataset 1	0.838
3	Dataset 2 + Dataset 4	Dataset 1	**0.853**
4	Dataset 2 + Dataset 5	Dataset 1	**0.926**
5	Dataset 2 + Dataset 6	Dataset 1	**0.915**
6	Dataset 2	Data set 2	0.945
7	Dataset 2 + Dataset 3	Dataset 2	0.931
8	Dataset 2 + Dataset 4	Dataset 2	0.942
9	Dataset 2 + Dataset 5	Dataset 2	**0.952**
10	Dataset 2 + Dataset 6	Dataset 2	0.948
11	Dataset 3	Dataset 2	0.326
12	Dataset 4	Dataset 2	0.572
13	Dataset 5	Dataset 2	**0.730**
14	Dataset 6	Dataset 2	0.651

presented in section III-A. We use a collection of 50 font types. For each line image, font size is randomly chosen. We apply the implementation presented in section IV-A,

- Dataset 5: 100000 synthetic line images generated by M-MUNIT configuration introduced in section III-B. Binary line images are generated by using the method presented in section III-A. We use a collection of 50 font types. We apply the implementation presented in section IV-A,
- Dataset 6: 30000 synthetic line images extracted from 100 synthetic images created by DocCreator tool.

A recognition system based on neural networks presented on [18] is used for our evaluation. For training and testing, we apply different configurations and the recognition rate is defined as follows:

The performance of a OCR system on a line image l_j is determined by the recognized text (w_j^r) and the annotation w_j^a of the line image, calculated as follows:

$$perf(OCR, l_j) = 1 - \frac{\min\left(length(w_j^a), dist(w_j^r, w_j^a)\right)}{length(w_j^a)} \quad (1)$$

Where $perf(OCR, l_j)$ is the performance of the OCR on the line image l_j; $dist(w_j^r, w_j^a)$ is the edit distance between the text recognized by the OCR and the groundtruth (defined by hand through a groundsourcing framework). Eq 1 is normalized by the length of the text line ($length(w_j^a)$) in the groundthruth in the way if the text line recognized by the OCR is equal to the groundtruth the performance score of the OCR will be equal to 1 since the edit distance will be zero in that case. The overall performance of an OCR system on the whole data set S is defined as the mean:

$$\frac{1}{|S|} \sum_{l_j \in S} perf(OCR, l_j) \quad (2)$$

See Table II for results of our evaluation.

As we can see in the table II, test 1 and 5 demonstrate the classical regularization effect. More precisely, when the recognition system, which is trained with dataset 2, recognizes images from dataset 1 (whose text font is totally different from images of dataset 2), the recognition rate drops from 0.945 to 0.822. Data augmentation methods, as demonstrated in tests 2, 3, 4, 5 improve the performance of the recognition system in this situation. In this perspective, we can remark that our proposed methods (with M-DualGAN and M-MUNIT) are better than simple data augmentation method (dataset 3). The M-MUNIT configuration provides better results than M-DualGAN configuration (see tests 3, 4, 8, 9), because M-MUNIT generates more realistic images than M-DualGAN.

In tests 11 to 14, we experiment the scenario where only synthetic images are used for training and real images are used for testing. Results showed that even in extreme case where no annotation data is available, a recognition system using our proposed method could yield a considerable good result (0.730 – M-MUNIT).

The M-MUNIT is slightly better than DocCreator, because, in our opinion, the image defects simulation in DocCreator is based on hand-engineered models that include a limited set of known variables. So DocCreator has limited capability and flexibility of approximating the distribution of real document images which are very complex. Our method, in the other hand, try to approximate distribution of real document images by training, thus have higher flexibility.

V. CONCLUSION AND PERSPECTIVES

In this paper, we proposed an automatic synthetic document image generation method using Generative Adversarial Networks (GANs). Our method is based on GANs that are usually used for image domain transformation task. This method consists of creating binary images and transforming binary images to realistic images by stimulating image defects. Our approach is proven to be low-cost as it only requires a collection of real document images without any annotation. Experimental results show the effectiveness of our approach to improve OCR recognition rate in a mobile-captured document images framework. Although in this paper, we mainly focus on modern printed document images, our proposed approach could be extended to another type of documents, including historical one.

Another feature of our proposed method is that, besides from generating realistic images by transforming from binary images to real images, we can perform image binarization by transforming real images to binary image. This idea must be investigated in future works.

To overcome the unwanted effect discussed in section IV-B, future works will be devoted to control the quality (distortion/defect degree) of generated images.

REFERENCES

[1] A. Halevy, P. Norvig, and F. Pereira, "The unreasonable effectiveness of data," 2009.

[2] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 843–852.

[3] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," *arXiv preprint arXiv:1708.04896*, 2017.

[4] A. Fischer, M. Visani, V. C. Kieu, and C. Y. Suen, "Generation of learning samples for historical handwriting recognition using image degradation," in *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing*, 2013, pp. 73–79.

[5] N. Journet, M. Visani, B. Mansencal, K. Van-Cuong, and A. Billy, "Doccreator: A new software for creating synthetic ground-truthed document images," *Journal of imaging*, vol. 3, no. 4, p. 62, 2017.

[6] F. Yin, Q.-F. Wang, and C.-L. Liu, "Transcript mapping for handwritten chinese documents by integrating character recognition model and geometric context," *Pattern Recognition*, vol. 46, no. 10, pp. 2807–2818, 2013.

[7] E. Ishidera and D. Nishiwaki, "A study on top-down word image generation for handwritten word recognition," in *Proceedings of the 7th International Conference on Document Analysis and Recognition (ICDAR)*, 2003, p. 1173.

[8] 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*, 2014, pp. 2672–2680.

[9] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 5967–5976.

[10] 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*, 2016, pp. 2172–2180.

[11] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 214–223. [Online]. Available: http://proceedings.mlr.press/v70/arjovsky17a.html

[12] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 2868–2876.

[13] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[14] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *Proceedings of the 33rd International Conference on Machine Learning-Volume 48*. JMLR. org, 2016, pp. 1060–1069.

[15] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[17] Cairo graphics. [Online]. Available: https://www.cairographics.org

[18] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 11, pp. 2298–2304, 2017.

[19] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777.

[20] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 214–223.

[21] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 1510–1519.

[22] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[23] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *International Conference on Machine Learning (ICML)*.

[24] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 2528–2535.

[25] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[26] J.-C. Wu, J.-W. Hsieh, and Y.-S. Chen, "Morphology-based text line extraction," *Machine Vision and Applications*, vol. 19, no. 3, pp. 195–207, 2008.