Data Augmentation via Adversarial Networks for Optical Character Recognition/ Conference Submissions

Victor Storchan
Analytics Consulting, CIB
BNP Paribas
Paris, France
victor.storchan@bnpparibas.com

Jocelyn Beauschene
Sloan School of Management
MIT
Boston, USA
jocelyn.beauchesne@polytechnique.edu

***Abstract*—With the ongoing digitalization of ressources accross the industry, robust OCR solutions (Optical Character Recognition) are highly valuable. In this work, we aim at designing models to read typical damaged faxs and PDF files and training them with unlabeled data. State-of-art deep learning architectures require scalable tagged datasets that are often difficult and costly to collect. To ensure compliance standards or to provide reproducible cheap and fast solutions for training OCR systems, producing datasets that mimic the quality of the data that will be passed to the model is paramount. In this paper we discuss using unsupervised image-to-image translation methods to learn transformations that aim to map clean images of words to damaged images of words. The quality of the transformation is evaluated through the OCR brick and these results are compared to the Inception Score (IS) of the GANs we used. That way we are able to generate an arbitrary large realistic dataset without labeling a single observation. As a result, we propose an end-to-end OCR training solution to provide competitive models.**

I. Introduction

Transcribing text from fax images is the very first step to enable Document Intelligence tasks that proactively helps analysts of the banking sector to conduct their analyses. Digitalized data from the financial industry, typically faxs from contracts, is highly sensible, and their contents can not be publicly exposed. Moreover, we noticed that faxs datasets' quality was likely to substantially vary according to the different modalities that are used to acquire the image (i.e. scanner's performance, processes and steps involved in the fax generation etc). However, training a fully-supervised OCR model requires to be able to have a scalable way to label data. To our best knowledge, available techniques accross the industry to perform such a task are either using Mechanical Turk tools [Neuberg(2017)] or are attempting to use handcrafted transformations to adapt their synthetic data distribution to the real domain. Additionally, this work is mainly done on natural images dataset [Borisyuk et al.(2018)], [Wang et al.(2012)]. In this work, we chose to take advantages of Adversarial Networks techniques to fully learn these transformations. We reformulate the problem as being able to translate images from a clean source domain to a damaged target domain, the one of our real dataset. Unsupervised image-to-image translation is the task of learning a joint distribution from images drawn from two marginal distributions that represent a source domain and a target domain. In other words, it is the task of learning the conditional distribution that given an image in the source domain, will describe the corresponding images in the target domain. The idea is derived from the original paper from I. Goodefellow who introduced Generative Adversarial Networks as a way to learn intractable probabilities distributions by jointly training a generative model G which goal is to generate new data given a training data distribution and a discriminative model D which aims to discriminate i.e. classify between the two classes [Goodfellow et al.(2014)]. In this paper we evaluated different GANs architectures to generate realistic synthetic datasets for training an end-to-end OCR solution. We are considering four GANs architectures that enable unsupervised image-to-image translation without paired examples. Meaning that at training time, we are unable to expose to the model the pair of a source image and its corresponding damaged version. We have two unpaired buckets of images. Thus we can't address the problem with conditional generative model or a simple regression (quote same paper than in MUNIT paper). We need to use ways that enable supervsion at the set level. Let us first describe the four types of models that we will use:

- CycleGan: We refer to the paper [Zhu et al.(2017)]. The main idea of CycleGan is to introduce a cycle consistency loss to ensure a good reconstruction of the source image and avoid the mode collapse. It is added to the traditional adversarial loss. CycleGan is made of two residual translation networks.
- UNIT: [Liu et al.(2017)] In the UNIT model, the assumption is that the the source and target domains can be mapped to a shared latent space. Compared to MUNIT, no assumptions concerning the modalities of the image is done here. In practice, the shared latent space assumption is implemented with a weight sharing constraint between

Identify applicable funding agency here. If none, delete this.

the encoding and decoding functions on their last layers. For more details, we refer to (quote paper)

- MUNIT: In MUNIT [Huang et al.(2018)], the shared latent space assumption is refined to a partial version of it. Assuming that each image is encoding two modalities, style and content, only the content can be shared between the two domains. The style comes from prior distributions that describe the two domain specific style spaces. It is particularly relevant for images of text where the font and the texture of the letter are interpreted as being the style. In this framework, the alphabet is the content. Compared to what is done in UNIT and CycleGan, in MUNIT, only a partial cycle-consistency loss is used to prevent the network from degenerating to a deterministic function. Instead, the authors only used a style cycle consistency loss a training time.
- Augmented CycleGan: Augmented CycleGan [Almahairi et al.(2018)] is the version of CycleGan that enables many-to-many mapping. Like UNIT and MUNIT, the transformation is now stochastic by using a prior distribution to generate an image that is encoding missing latent information.

Let's note that we will compare these methods where the transformation is learnt against handcrafted functions that we designed to mimic the transformations. In UNIT, the stochastic aspect of the learnt transformations comes from the dropout layers and the Gaussian Encoder function that output the latent code. In MUNIT, the stochastic aspect comes from the fact that we are able to sample from different style codes to provide multimodal outputs.

To get back to the generation of a realistic synthetic dataset for training our OCR, we model the problem as a one-to-many translation. The "one" part is the clean generated image domain. The "many" part is referring to the damaged datasets that are supposed to be obtained through different modalities, thus that contain different image qualities. For example, one dataset can be very pixelated due to bad scanner device, another one can suffer from a very noisy background and so on.

II. Motivation and related work

Data Augmentation for OCR. A pretty dense litterature on image augmentation for performing OCR into the wild has been published. They mainly deal with natural images that contain text and aimed to optimize a two steps pipeline: text detection and text recognition. For instance, Ankush Gupta Andrea Vedaldi and Andrew Zisserman in [Gupta et al.(2016)] obtained good results to overlay synthetic text to existing backgroud natural images. In [Naseer & Zafar(2018)], the authors used a real dataset for Urdu recognition but they are directly cropping images of different qualities in documents which force them to label the data.

Deep learning OCR architectures. In vision, Convolutional Neural Networks introduced by Y. LeCun imposed the idea of local connectivity and are commonly accepted as being the natural architecure when dealing with volumes from images. Reccurent Neural Networks are a way to encode in a directed graph the temporal dynamics of a sequence [Sak et al.(2014)]. For a couple of years, the state-of-art deep networks that are used to provide robust OCR solutions are based on Convolutional Neural Networks and Recurrent Neural Networks. Typically, as [Baoguang et al.(2017)] puts it in its Figure 1, the role of the CNN is to compute the optical feature sequences from the image. They are then passed to a bi-directional LSTM that both takes into account context from both directions of the sentence and captures an implicit language model. Then an additionnal operation of transcription is needed to translate the per-window predictions to a readable sequence. This is the role of the Connectionist Temporal Classification (CTC) that we used with a greedy decoder. We find out that using a beam search with $k \in \{3, .., 5\}$ marginally improved the result while the processing time was substantially increased.

III. Generative Adversarial Networks: learning to augment the data

A. DeepFont: sampling from the font distribution

An essential part of generating a synthetical realistic OCR dataset is to find which fonts have been used in the original dataset in order to mimic it. By implementing and training DeepFont on a real dataset, it is possible to do just so without labeling any real images. Furthermore, frequency of occurence for each font can be computed then used as a font distribution during the generation of the synthetical training dataset.
It is to be noted that the prior use of DeepFont is key to train CycleGAN, as fonts that are morphologically too different from the ones in the real dataset are easily flagged by the discriminative network and forces in turn the generative network to collapse such samples. In other words, one needs to assert that the source and target domains are morphologically close enough to facilitate the transition from one domain to the other.
Thus, before being able to generate a dataset, the work consists in training a font classifier. Following the idea and the work of [Wang et al.(2015)], training these kind of classifiers is a two steps process. First, we trained an encoder-decoder with a mix of synthetic data and real images cropped in faxs to learn an embedding layer at the output of the encoder. Then the encoder is plugged on a softmax classifier that is trained in a fully supervised manner: we generate images of words with known fonts and train the network to guess the right font.
We noticed that the first fully connected layer of the classifier was accountable for almost 90% of the 83 millions parameters of the nework. That is an issue to enable fast inference, espacially without GPU in production. Because we aim to use DeepFont to keep the correct font in the outputed document of our OCR, we reduced the size of this layer. Whereas the paper is computing at each iteration the best rank-k approximation of the fully connected layers, we chose to apply SVD to the dense matrices that encode the weights of the three fully connected layers at the end of the training time. Likewise, the size of the network is reduced from 320MB to 50MB After bootstrapping, we obtain 82% of accuracy at $r@3$.

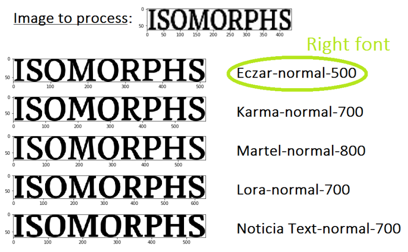


Fig. 1: Example of DeepFont r@5 result

B. Dataset generation

With no labeled data there is no insight on the actual characters or words distributions. Hence the need for a generic text generator. However, experiments showed that training the OCR engine with random strings yields terrible results. Indeed, according to (article about language model learned by an LSTM) a RNN learns an implicit language model. Hence the confusion between "|" and "l" when both occur at the same rate.
We carefully generated a corpus of word that is designed to answer a twofold objective: first we want the letter distribution and the vocabulary to mimic the one of our dataset. Second, we want to create a dataset that generalize well without knowing in advance the distribution of our dataset.
Given an alphabet $\mathcal{A}$, we call $\mathcal{R} = \cup_{k=1}^{14} C^k$ the set of random string of length 1 to 14. These random strings are built to teach our network about special characters, numbers, codes that are present in the dataset. Additionally, let $\mathcal{F}$ be a french dictionnary and $\mathcal{E}$ an english one. As the sets $\mathcal{R}$, $\mathcal{F}$, and $\mathcal{E}$ have very different sizes, we construct our lines with the following procedure:

- Pick uniformly at random a string length m between 1 and 4.
- Pick $\mathcal{S}_m \in \cup_{r+f+e=m} \mathcal{R}^r \mathcal{F}^f \mathcal{E}^e$ at random.
- Pick a random m-tuple t_m from S_m.

C. GANs for image-to-image translation: comparison of different architectures

As stated in the introduction, the key step to deal with unlabeled data for OCR is to produce a synthetic dataset that mimic the real datasets. To this purpose, we are using 4 different architectures of GANs described above. They all do image-to-image translation but does not behave the same way when facing to the diversity of fonts or data overall quality. Moreover they are not all providing output diversity which is something very appreciated in the context of training an OCR. Thus, we trained these 4 solutions to produce 4 datasets generated from a single clean dataset. We collected one additional dataset made of handcrafted transformations. These datasets will be feeding 5 OCR networks at training time. Although we reimplemented the GANs (respectively CycleGan, AugmentedCycleGan, MUNIT and UNIT), we refer to the original papers for full details about them.
To train our GANs, we constructed the dataset using the following technique: we build two non aligned buckets X and Y. X is containing the damaged (i.e pixelated) images whereas Y is the set of the clean images. We assume that the data quality is not uniform within X. This assumption will not be taken into account by CycleGan which can only sample from a unique distribution.
Implementations: We followed the recommanded training procedures for CycleGan, UNIT and MUNIT. However, to train AugmentedCycleGan, we thought that the Gaussian prior $p(z)$ that was introduced on the latent space was not relevant for our usecase. We prefered to use a uniform distribution as we supposed that the different data qualities were equally represented in the original dataset.
We cropped $32 * 32$ images in the words that are synthetically generated for Y and directly in our dataset for X as follow.

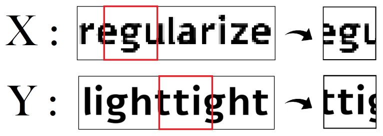


Fig. 2: Method to build the datasets to train GANs

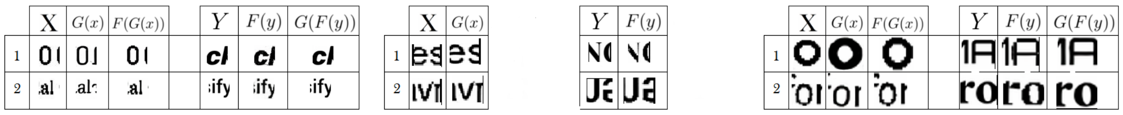
Here we only provide examples of output images that one might have thanks to each of these networks.

IV. End-to-end OCR training: learning to read the data

Scanned documents are often noisy, especially if those documents were sent by fax, printed and only then scanned. However, for privacy and compliance reasons, the use of a plateform such as Mechanical Turk was not an option. In this work, we successfully developped a unsupervised pipeline to generate images of text which mimic the font and noise distribution of a real dataset. The OCR engine trained on this synthetical dataset outperforms Tesseract and ABBYY Finereader on real examples by more than 45 percents in terms of mean edit distance. Moreover, to demonstrate the performance of the use of GANs, we compare an OCR trained with handcrafted transformation carefully implemented to mimic the dataset with the automated transformation from adversarial training.

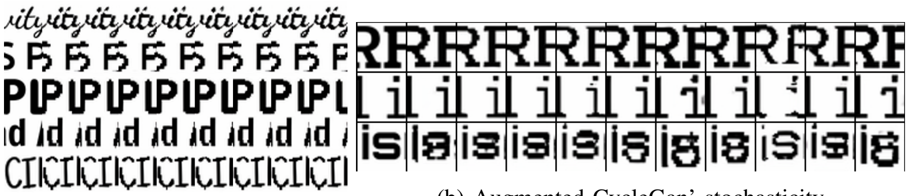
A. The preprocessing unit: engineering for text detection

Implementing an end-to-end solution for OCR implies a two steps process: text detection and text recognition. The first step is a purely engineering step which consist to rescale the image while preserving its ratio, applying some cleaning filters, and normalise its dpi. Additionally, the image is thresholded with a greyscale to recover the contours of the darker lines (close to black). Then we implemented rules to remove the layout of the pdfs that are not lines of words. Among others, we trained a classifier that separate image of text from image of logos or pure noise.



(a) CycleGan Results (b) UNIT results (c) AugmentedCycleGan results

Fig. 3: Reconstruction from UNIT, CycleGan and its augmented version



(a) MUNIT' stochasticity (b) Augmented CycleGan' stochasticity

Fig. 4: Diversity from MUNIT and Augmented CycleGan

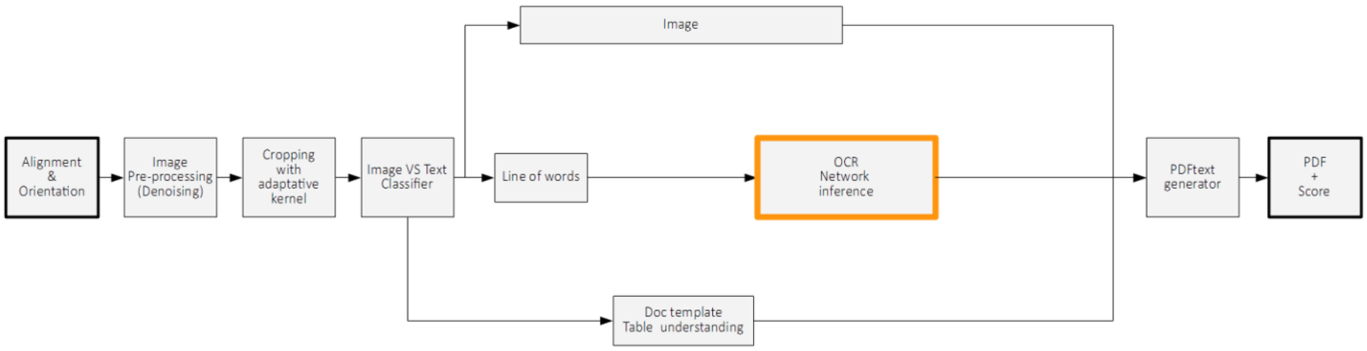


Fig. 5: Pipeline of the end-to-end OCR solution

B. The Reading Unit

Cropping each single word within faxs is a very complex task, as some parts of the layout can damage such an operation for example. To avoid this dead end, we set up a reading unit that will pass horizontally over the sequence of characters of a cropped line image. Moreover, to put more weight on the center of the image which is the part that matter the most, we multiply the 2D signal by a gaussian kernel that will reduce the signal on the side of the image. It means that the piece of letters on the side of the image will count less than the letter at the center. Eventually, we fix the step of the window to 4 pixels which appeared to be a good tradeoff. It behaves like a sliding window. As input features, we record at each step of the window the proportion of each letter that is present in it. Let's note that if a letter appears twice within the window, the percentages (or proportions) add them up. As a consequence, the CNN's job is more about dealing with regression than classification.

C. The architecture of the OCR network

We chose to use a well known architecture for our OCR pipeline, modifying the layers of a typical VGG to enable fast inference without loosing too much accuracy for our task.

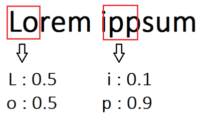


Fig. 6: Input and Output of the CNN part of our OCR network

This VGG, as stated above is performing a regression task to estimate the horizontal percentage of filling of the sliding window per letter. After that, we stack a Bi-LSTM that will learn an underlying language model. Taking the proportions outputed by the CNN as input, it is trained with CTC loss. It is to be noted that the use of Recurrent Neural Networks for OCR which seems to be a natural choice can be removed.

D. OCR Training

The OCR engine is trained in 3 steps :

- First training a small VGG-like network to predict the weighted percentage of a letter present in the sliding window ;
- Training a LSTM with a CTC loss on a synthetical set of perfect sequence

Network	Mean edit distance to GT
Default Tesseract	3.119
Default ABBYY	2.532
Synthetic filter	1.684
CycleGan filter	1.319
AugmentedCycleGan filter	1.288
UNIT filter	1.282
MUNIT filter	1.199

TABLE I: Performance on the OCR task of our networks trained with transformations produced by different GANs

- Fine-tuning the network (CNN+RNN) end to end using a CTC loss

As one could expect, we found that training the RNN model was significantly harder than the CNN. That is why we chose to train both networks separately before finetuning the whole system. This yields a small, fast and robust OCR Engine beating ABBYY Finereader on our real dataset by a margin of 45 percents in terms of mean edit distance.

V. Evaluation and Results

A. Evaluation Metric

The principal task we want to evaluate is the pure text recognition task, meaning that we will not consider the full OCR task which is more complex and involves the preprocessing of our data. As the purpose of the work is to evaluate how the different image-to-image translations tasks are performing, we decided to only consider the edit distance between the infered word and the ground truth. To this purpose, we labelled manually 1200 word images that were cropped from the real dataset. Let's denote this evaluation set as the "EvalSet". Eval set is passed in input to the 4 OCR models that have been trained with the 4 datasets produce by the 4 GANs. Although this evaluation process is somewhat biased because it does not compare directly the output images of the GANs with the one of the EvalSet, we argue that as the final goal is to perform OCR, we want to select the Generative Adversarial Network that will provide the best training dataset with respect to the true data quality for the OCR task.
The edit disstance between two words w_1 and w_2, $d(w_1, w_2)$ is define by computing the minimum sequence of edit transformations that map w_1 on w_2. The set of these transformation is: $\{Insertion, Deletion, Substitution\}$

B. Results

In the table I are the results that we obtain:

As a result, it is clear that we benefit from data augmentation that was provided by the different GANs. If these results are evaluating a pure OCR task, we can nevertheless put them in regards to the popular Inception Score (IS) that have been shown to correlate with human scoring. It is to be noted that [Huang et al.(2018)] introduce a new Conditional Inception Score (CIS) that measures diversity of outputs conditioned on a single image. We won't use this metric as one would

GAN Type	IS score
CycleGan	0.717
AugmentedCycleGan	0.842
UNIT	1.000
MUNIT	1.181

TABLE II: IS scores for the different GANs. Unsurprisingly, these scores correlate well with the performance on the OCR task.

have received a zero CIS score for CyCleGan as it outputs a single image given an input. In contrary, as noted in [Barratt & Sharma(2018)], the IS score introduced in [Salimans et al.(2016)] measures how the inception network is confident there is a single object in the output image and how the generative algorithm outputs highly diverse images from all the different classes. In other words, we have:

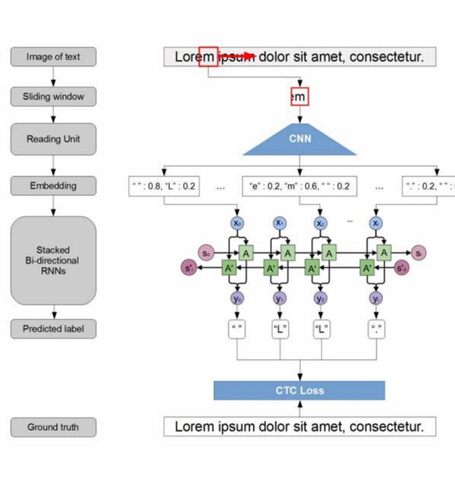
$$IS(G) = \mathbb{E}_{x \sim p_g}[D_{KL}(p(y|x) \| p(y)))]$$

where $x \sim p_g$ means that x is drawn from the generated distribution over new samples p_g and $D_{KL}(p(y|x)\|p)$ is the KL-divergence between $p(y|x)$, representing the conditional class distribution and p(y) which is the marginal class distribution (our model distribution). Consistently with the results we obtained when evaluating the different GANs through the prism of the OCR task, we can more directly estimate their IS scores. We summarized the result in II. In short, MUNIT is achieving the best performance. It is interesting to note that it is achieving its goal to provide diversity (which is taken into account in the Inception Score. Moreover, its ability to decouple the style and the content seems to be a winning strategy for our purpose.

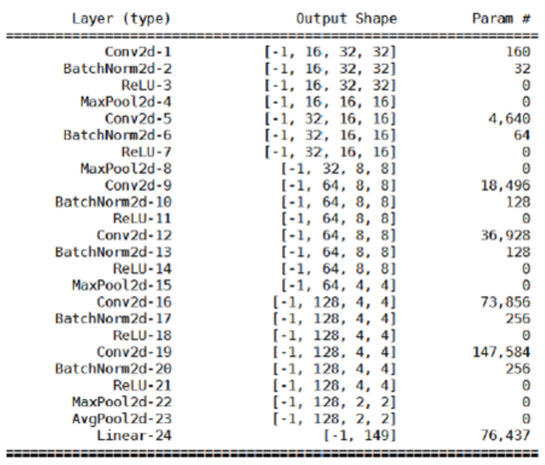
We can conclude that the more the GAN reflects high diversity of images from all different classes, the more accurate will be your OCR engine. Moreover, we can note that even if the synthetic transformations gave a worse performance compared to what is achieved with GANs, it is still better than default OCR bricks like Tesseract or ABBYY on data of bad quality. Eventually, we observed that taking advantage of a simple decomposition of the image as being the result of its content (i.e. the word that is written in it) and its style (i.e. a combination of the quality of the image, the font, the texture etc..) lead us to significantly improve our results. It is to be noted that the use of Recurrent Neural Networks for OCR which seems to be a natural choice can be removed. Actually, recent work [Borisyuk et al.(2018)] showed that it is possible to only use convolutions to model the dependency between characters. The main advantage is that it not only reduces the training time but it is also faster at inference time.

VI. Conclusion and Future Work

In this paper, we presented a new approach for data augmentation for OCR. Our appraoch benefits from the recent advances of image-to-image translation systems that clearly



(a) OCR network while training


```
          Layer (type)               Output Shape         Param #
================================================================
              Conv2d-1           [-1, 16, 32, 32]             160
         BatchNorm2d-2           [-1, 16, 32, 32]              32
                ReLU-3           [-1, 16, 32, 32]               0
           MaxPool2d-4           [-1, 16, 16, 16]               0
              Conv2d-5           [-1, 32, 16, 16]           4,640
         BatchNorm2d-6           [-1, 32, 16, 16]              64
                ReLU-7           [-1, 32, 16, 16]               0
           MaxPool2d-8             [-1, 32, 8, 8]               0
              Conv2d-9             [-1, 64, 8, 8]          18,496
        BatchNorm2d-10             [-1, 64, 8, 8]             128
               ReLU-11             [-1, 64, 8, 8]               0
             Conv2d-12             [-1, 64, 8, 8]          36,928
        BatchNorm2d-13             [-1, 64, 8, 8]             128
               ReLU-14             [-1, 64, 8, 8]               0
          MaxPool2d-15             [-1, 64, 4, 4]               0
             Conv2d-16            [-1, 128, 4, 4]          73,856
        BatchNorm2d-17            [-1, 128, 4, 4]             256
               ReLU-18            [-1, 128, 4, 4]               0
             Conv2d-19            [-1, 128, 4, 4]         147,584
        BatchNorm2d-20            [-1, 128, 4, 4]             256
               ReLU-21            [-1, 128, 4, 4]               0
          MaxPool2d-22            [-1, 128, 2, 2]               0
          AvgPool2d-23            [-1, 128, 2, 2]               0
             Linear-24                  [-1, 149]          76,437
================================================================
```

(b) architecture of our CNN network

Fig. 7: OCR Training steps

outperform any handmade transformations. Our better results being achieved by MUNIT, it demonstrates how diversity from multimodal GANs can benefit to the OCR task. Using this framework enables us to provide a simple procedure to train an OCR engine for fax documents bypassing the data labelling step. In this sense, the training is unsupervised in regards to the real dataset, i.e. decoupled from it. Not only evaluating the performance of the different GANs on the task of interest (OCR) lets us understand better how our synthetic training set reflects the real data, but it also tells us about the relevancy of each GAN architecture. Future work includes extanding this framework to handwritten datasets. One of the issue is that the diversity of datasets highly scales in this case. A very promising future direction of work would be to integrate a distillation framework over the synthetic dataset as [Wang et al.(2018)] proved its efficiency on several public dataset. The idea to compress all the information contained in a large dataset to reduce it to only few images to train the network would be extremely well suited for our task because not only it would be easier to compare how well the GANs are doing on a small number of images but it would also substantially reduce the training time. In a context where potential clients would ask us to provide them with a specific OCR brick trained for their data, it would be an interesting advantage.

Acknowledgments: The authors would like to thanks Hugues Even, Edouard d'Archimbaud and Thomas Gilles from BNP PARIBAS, Analytics Consulting at CIB who contributed to the development of the system and enabled access to the data. We would like to thanks Patrick Gallinari for his support.

References

[Almahairi et al.(2018)] 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.

[Baoguang et al.(2017)] Shi Baoguang, Bai Xiang, and Yao Cong. 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*, 39(11):2298–2304, 2017.

[Barratt & Sharma(2018)] S. Barratt and R. Sharma. A note on the inception score. *arXiv preprint arXiv:1801.01973*, 2018.

[Borisyuk et al.(2018)] F. Borisyuk, A. Gordo, and V Sivakumar. Rosetta: Large scale system for text detection and recognition in images. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 71–79, 2018.

[Goodfellow et al.(2014)] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, and Y ...and Bengio. Generative adversarial nets. *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[Gupta et al.(2016)] A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic data for text localisation in natural images. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2315–2324, 2016.

[Huang et al.(2018)] X. Huang, M. Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. *arXiv preprint arXiv:1804.04732*, 2018.

[Liu et al.(2017)] M. Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. *Advances in Neural Information Processing Systems*, pp. 700–708, 2017.

[Naseer & Zafar(2018)] A. Naseer and K. Zafar. Comparative analysis of raw images and meta feature based urdu ocr using cnn and lstm. *Int J Adv Comput Sci Appl*, 9(1):419–424, 2018.

[Neuberg(2017)] Brad Neuberg. Creating a modern ocr pipeline using computer vision and deep learning. https://blogs.dropbox.com/tech/2017/04/creating-a-modern-ocr-pipeline-using-computer-vision-and-deep-learning/, 2017.

[Sak et al.(2014)] H. Sak, A. Senior, and F. Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. *Fifteenth annual conference of the international speech communication association.*, 2014.

[Salimans et al.(2016)] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X Chen. Improved techniques for training gans. *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

[Wang et al.(2012)] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng. End-to-end text recognition with convolutional neural networks. *Pattern Recognition (ICPR), 2012 21st International Conference. IEEE*, pp. 3304–3308, 2012.

[Wang et al.(2018)] T. Wang, J.Y. Zhu, A. Torralba, and A.A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959v1*, 2018.

[Wang et al.(2015)] Z. Wang, J. Yang, H. Jin, E. Shechtman, A. Agarwala, J. Brandt, and T. S. Huang. Deepfont: Identify your font from an image. *Proceedings of the 23rd ACM international conference on Multimedia*, pp. 451–459, 2015.

[Zhu et al.(2017)] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*, 2017.