UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
POST-GRADUATE PROGRAM IN COMPUTER SCIENCE

Gabriel Rezende da Silva

Cattle face generation using an adversarial network with bounding boxes of facial subregions

Juiz de Fora

2024

Gabriel Rezende da Silva

Cattle face generation using an adversarial network with bounding boxes of facial subregions

Dissertation presented to the Post-Graduate Program in Computer Scienceof the Universidade Federal de Juiz de Fora as a partial requirement to obtain the Master's degree in Computer Science. Concentration: Computer Science.

Advisor: Ph.D. Marcelo Bernardes Vieira

Coadvisor: Ph.D. Luiz Maurílio da Silva Maciel

Juiz de Fora

2024

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF
com os dados fornecidos pelo(a) autor(a)

Rezende, Gabriel.
Cattle face generation using an adversarial network with bounding boxes of facial subregions / Gabriel Rezende da Silva. – 2024.
57 f. : il.

Advisor: Marcelo Bernardes Vieira
Coadvisor: Luiz Maurílio da Silva Maciel
Dissertação (Mestrado) – Universidade Federal de Juiz de Fora, Instituto de Ciências Exatas. Post-Graduate Program in Computer Science, 2024.

1. Cattle facial generation. 2. Deep learning . 3. Generative Adversarial Networks. 4. Bounding box module. II. Vieira, Marcelo Bernardes, orient. II. Maciel, Luiz Maurílio, coorient. III. Título.

Gabriel Rezende da Silva

Cattle face generation using an adversarial network with bounding boxes of facial subregions

Dissertação apresentada ao Programa de Pós-graduação em Ciência da Computação da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Ciência da Computação. Área de concentração: Ciência da Computação.

Aprovada em 27 de setembro de 2024.

BANCA EXAMINADORA

Prof. Dr. Marcelo Bernardes Vieira - Orientador
Universidade Federal de Juiz de Fora

Prof. Dr. Luiz Maurílio da Silva Maciel - Coorientador
Universidade Federal de Juiz de Fora

Prof. Dr. Jairo Francisco de Souza
Universidade Federal de Juiz de Fora

Prof. Dr. Gilson Antonio Giraldi
Laboratório Nacional de Computação Científica

Juiz de Fora, 13/09/2024.

Documento assinado eletronicamente por **Marcelo Bernardes Vieira**, **Professor(a)**, em 04/11/2024, às 16:00, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Luiz Maurílio da Silva Maciel**, **Professor(a)**, em 04/11/2024, às 17:46, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Jairo Francisco de Souza**, **Professor(a)**, em 04/11/2024, às 22:53, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Gilson Antonio Giraldi**, **Usuário Externo**, em 10/11/2024, às 06:18, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no Portal do SEI-Ufjf (www2.ufjf.br/SEI) através do ícone Conferência de Documentos, informando o código verificador **1983811** e o código CRC **285FE210**.

ACKNOWLEDGEMENTS

To my parents, Lúcia and Jorge, my sister, Patrícia, my fiancée, Virginia, and my friends, for all the love and emotional support.

To my advisors, Marcelo and Luiz, for their trust, patience, and dedication in guiding me throughout these years, and to the PPGCC professors, for all the knowledge shared. To the staff who have assisted me in various ways.

To the graduates, Mathews and Thaís, for their dedication in creating the original dataset, without which this work would not have been possible.

To the Group for Computer Graphics, Image and Vision for their contributions.

To Embrapa Dairy Cattle for the initiative that supports this project, particularly to researcher Bruno Campos de Carvalho, project's visionary.

To CAPES and FAPEMIG institutions for their financial support.

"Our intelligence is what makes us human, and AI is an extension of that quality." – (Yann LeCun)

RESUMO

A coleta de dados no âmbito da pecuária de precisão é um sério desafio. A escassez de conjuntos de dados de imagens anotadas para problemas relacionados à pecuária de precisão limita o potencial de algoritmos de aprendizado profundo. As informações aprendidas pelas Redes Adversárias Generativas (GANs) podem ser empregadas para melhorar o desempenho de tarefas de aprendizado de máquina, como classificação. Sub-regiões específicas da região facial de bovinos, como focinho, orelhas e olhos, são indicadores significativos de dor, desconforto e estresse nos animais, além de serem utilizadas para a identificação individual. Nesse contexto, o presente trabalho propõe uma nova arquitetura que reforça a localização espacial de sub-regiões faciais de interesse. O método proposto integra módulos de caixa delimitadora com uma GAN conhecida na literatura, denominada BigGAN. A hipótese é que a incorporação de informações de localização espacial, fornecidas por caixas delimitadoras de sub-regiões de interesse, como focinho, orelhas e olhos, pode aprimorar a geração de imagens sintéticas de faces de bovinos. Assim, propõe-se uma nova configuração desses módulos, baseada no provimento de múltiplos objetos em uma cena. Os módulos são acoplados em diferentes profundidades no gerador, formando uma pirâmide de resolução, e em um nível específico no discriminador. Também propõe-se a utilização de caixas multi-escalas para representar as sub-regiões de interesse. A premissa é suavizar as áreas de transição entre os objetos e o plano de fundo, ao mesmo tempo em que reforça as características dos objetos. Para treinar o modelo, filtrou-se um conjunto de dados pré-anotados para a tarefa de detecção, adaptando-o à tarefa de geração. O novo conjunto de dados inclui 9.495 faces de bovinos e um subconjunto de anotações de caixas delimitadoras. Para avaliar a viabilidade da proposta, realizou-se uma análise extensiva, tanto quantitativa quanto qualitativa. Os resultados experimentais mostraram ganhos proporcionados pela inserção de informações adicionais de sub-regiões de interesse via módulos de caixa delimitadora. A capacidade de inserir arbitrariamente a posição das sub-regiões permite possíveis aplicações como a geração facial personalizada pelo usuário.

Palavras-chave: Geração facial de bovinos. Aprendizado profundo. Redes Adversárias Generativas. Módulos de caixa delimitadora.

ABSTRACT

Data collection in precision livestock farming is a serious challenge. The scarcity of annotated image datasets for precision livestock problems limits the potential of deep learning algorithms. The information learned by the Generative Adversarial Networks (GANs) can be employed to improve the performance of machine learning tasks, such as classification. Specific subregions of the facial cattle region, such as the muzzle, ears, and eyes, are significant indicators of pain, discomfort, and stress in animals, as well as being used for individual identification. In this context, in the present work we develop a new architecture that enhances the spatial localization of facial interest subregions. The proposed method combines bounding box modules with a GAN known in the literature, named BigGAN. The hypothesis is that the incorporation of spatial location information, provided by interest subregions bounding boxes, such as the muzzle, ears, and eyes, can improve the generation of synthetic cattle faces images. Thus, we proposed a new configuration of these modules, based on providing multiple objects in a scene. The modules are attached at different depths in the generator, forming a resolution pyramid, and at a specific level in the discriminator. We also propose to use a multiscale boxes to represent the interest subregions. The premise is to smooth out the transitions between the objects and the background, while reinforcing the objects' features. To train the model, we filtered a pre-annotated dataset for the detection task and adapted it for the generation task. The new dataset includes $9,495$ cattle faces and a subset of bounding box annotations. To assess the feasibility of the proposal, we conducted extensive analysis, both quantitative and qualitative. The experimental results showed improvements provided by the inclusion of additional information for the interest subregions via bounding box modules. The capacity to arbitrarily insert the subregions' position allows for possible applications such as facial generation designed by user.

Keywords: Cattle facial generation. Deep learning. Generative Adversarial Networks. Bounding box modules.

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AvgPool	Average Pooling
BN	Batch Normalization
C2D	2-dimensional Convolutional Layer
CBN	Conditional Batch Normalization Layer
CNN	Convolutional Neural Network
D	Discriminator
FC	Fully Connected Layer
FID	Fréchet Inception Distance
G	Generator
GAN	Generative Adversarial Network
LeakyReLU	Leaky Rectified Linear Unit
PLF	Precision Livestock Farming
ReLU	Rectified Linear Unit
SN	Spectral Normalization
STN	Spatial Transformer Network
Tanh	Hyperbolic Tangent

CONTENTS

1 INTRODUCTION

Cattle facial features provide distinctive information about phenotype and animal welfare. This information have great importance for Precision Livestock Farming (PLF), whose aim "is to manage individual animals by continuous real-time monitoring of health, welfare, production/reproduction and environmental impact" (BERCKMANS, 2017). With the implementation of PLF technologies over the last years, one of the main concerns is the data acquisition.

Ensuring the quantity and quality of livestock data is a relevant challenge. Overall, cattle monitoring methods are based on electronic sensors. Specific sensors detect different physical and environmental conditions. Therefore, the merits of each sensor system need to be assessed, considering the expensive investment of some technologies, loss or noise measurement, and implantation process that can cause stress and lesions in the animals. (STYGAR et al., 2021). Other factors such as adverse weather conditions, farm location and devices with short battery life can hinder data collection (PAPAKONSTANTINOU et al., 2024).

Image data types can be acquired by cameras of different spectra or by the processing of other signal sources. In the context of PLF, a natural image captured by a conventional RGB camera carries a dense amount of information, from primary aspects related to the animal, such as individual identification, behavior and socialization, health status, etc., to secondary ones, such as confinement system, sky conditions (rainy, cloudy or sunny) and vegetation (indicating the season), etc., enabling sophisticated, non-invasive, and real-time computer vision systems (OLIVEIRA et al., 2021). Computer vision is a field of computer science dedicated to developing algorithms for extracting and processing information from digital images and videos (SZELISKI, 2022).

Recently, deep learning techniques became prominent in many computer vision applications, such as object detection (DIWAN; ANIRUDH; TEMBHURNE, 2023), image classification and segmentation (TAN; LE, 2019; MINAEE et al., 2021), and super-resolution (BASHIR et al., 2021), especially by the capacity of the Convolutional Neural Networks (CNNs) to process large data volumes. However, the scarcity of annotated image datasets for livestock activities is limiting the potential of deep learning algorithms (LU et al., 2022). Generative Adversarial Networks (GANs), introduced by Goodfellow et al. (2014), are capable of learning good data representations and generating realistic synthetic samples, being widely employed to augment existing datasets and improve the performance of machine learning models.

The primary motivation of this work is how to enhance facial generation of cattle introducing the spatial localization information provided by bounding boxes into a GAN. An annotation example is shown in Figure 1. The demarcated regions (muzzle, ears and

eyes) can be easily annotated semi-automatically, creating a facial layout. Those regions are good predictors of pain, discomfort and stress in cattle (GLEERUP et al., 2015). Furthermore, Singh, Devi and Varish (2021) used GANs in the cattle identification based on the muzzle pattern. In that sense, this work is inspired by the generation of scenes with multiple objects (HINZ; HEINRICH; WERMTER, 2019). The facial layout and background make up the scene, while the interest subregions are the objects. The premise is that this configuration can induce the generation of the face as a whole.

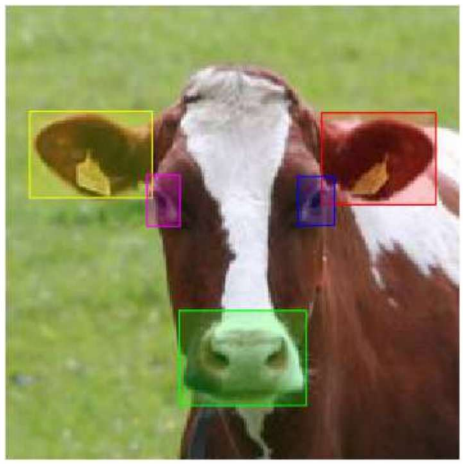


Figure 1 – Annotation example of the interest subregions: muzzle (green), left and right ears (red and gold, respectively), and left and right eyes (blue and purple, respectively).

Source: Created by the author (2024).

One of the most advanced states of the art in GANs is the human facial generation, trained on available datasets with hundreds of thousands of images. In contrast, this work has a dataset with only a few thousand of cattle faces, which requires the application of existing methods in the literature to deal with the GANs training under limited datasets. Zhao et al. (2020) proposed a differentiable data augmentation method in both real and fake images, and Tseng et al. (2021) added a regularization term to the discriminator loss function. Both works conducted their experiments on the leading class-conditional BigGAN (BROCK; DONAHUE; SIMONYAN, 2019). In a similar way, this work also uses the BigGAN, adapting its architecture into bounding box modules.

1.1 PROBLEM DEFINITION

The research hypothesis of this work is that it is possible to generate cattle faces where the spatial location information provided by bounding boxes of interest subregions, such as muzzle, ears and eyes, can be incorporated into a GAN, in order to quantitatively and qualitatively improve the generation of synthetic images. An important restriction of this work is the use of a limited dataset due to the inherent challenges of collecting and annotating a large number of samples for this problem.

1.2 OBJECTIVES

The main objective of this work is to generate cattle faces through the integration of modules that reinforce the spatial localization of salient facial regions with a GAN, improving the quality of generated images. Several experiments are conducted to quantitatively and qualitatively assess the synthetic images through generative model performance measures and visual analysis. The secondary objective is to explore new configurations of the bounding box modules that are more adapted to the facial generation. The expectation is that the model can assist phenotyping approaches, such as breed, age and pose classifications, and health status inference.

1.3 CONTRIBUTIONS

The main contributions of this work are:

1. Integration of bounding box modules with a GAN known in the literature, named BigGAN, to improve facial generation of cattle. This methodology can be adapted for other GANs;

2. A novel facial generation approach, reinforcing the spatial localization of interest subregions, such as muzzle, ears and eyes, to induce the generation of the face as a whole;

3. Generation quality gains provided by the insertion of additional information for the interest subregions via bounding box modules. Subregions can be inserted at arbitrary positions, allowing the control to specify the bovine pose. This can be useful for applications such as facial generation designed by user;

4. A filtered dataset for cattle facial image analysis and synthesis applications.

1.4 OUTLINE

This work is structured as follows: Chapter 2 presents relevant works in the literature to the focus of this dissertation; Chapter 3 describes the fundamentals that supports the proposed method; Chapter 4 details the proposed methods; Chapter 5 organizes the experiments and results; Chapter 6 discusses the conclusions obtained and possible future work.

2 RELATED WORK

This chapter highlights existing works in the literature related to this dissertation. At the limits of our knowledge, no works were found that specifically generate cattle faces. However, the problem of cattle face generation is intimately related to the facial generation of human and other animals, as well as conditional image generation.

The synthesis of human faces represents one of the most advanced fields in image generation. This progress is supported by the availability of large-scale datasets containing hundreds of thousands of samples, such as the CelebA (LIU et al., 2015) and FFHQ (KARRAS, 2019) datasets. Large data volumes typically lead to more robust and generalizable deep learning models.

Recent methods, characterized by their absence of convolution, have been shown to enhance the realism and diversity of the generations in certain circumstances. Zhang et al. (2022) explored the use of pure transformers (VASWANI, 2017) to build GANs for high resolution image synthesis on established benchmarks, e.g., FFHQ and CelebA-HQ (LIU et al., 2018), a high resolution version of the CelebA dataset. Zhang et al. (2022) emphasized the importance of attention mechanisms to balance computational efficiency and the capacity to model local features, otherwise, blocking artifacts caused by transformers can occur. In a different vein, Kim et al. (2022) utilized the CelebA dataset in their score-based diffusion (HO; JAIN; ABBEEL, 2020) model that employs a data-adaptive nonlinear diffusion process. A limitation of the approach lies in the training and inference time. Despite the promising results presented by those methods, CNNs retain specific advantages, such as better preservation of local spatial correlations and reduced parameter complexity. Therefore, convolutional-based networks are often the most suitable choice for tasks involving limited data.

Animal face datasets tend to be smaller and less common than human face datasets. Additionally, most datasets are not specialized in a single species, such as the AFHQ (CHOI et al., 2020) dataset. Therefore, various state-of-the-art studies on those benchmarks apply techniques to address data limitations.

Karras et al. (2020) proposed an adaptive discriminator augmentation mechanism that significantly stabilizes training, particularly in moderate data scarcity scenarios. In addition to exploring limited datasets such as AFHQ, the authors studied how the quantity of available training data affects GANs performance by artificially subsetting larger datasets, such as LSUN (YU et al., 2015). Because Karras et al. (2020) used a broader set of augmentation techniques that rely on dynamic tuning of the application probability, in extreme data scarcity scenarios, it can be more challenging to properly adjust such adaptive parameters. In this work, we adopt the approach proposed by Zhao et al. (2020), which applies a differentiable data augmentation on both the discriminator and generator

updates during the training phase. While the method restricts the transformation options, it stands out for its easy implementation, generalization across different architectures and the ability to generate high-fidelity results even with extremely small datasets.

Kumari et al. (2022) investigated the use of an ensemble of pretrained discriminators to optimize the performance of GANs. Their results demonstrated improvements in both limited data scenarios and large datasets. However, the reliance on multiple pretrained discriminators substantially increases memory requirements during training, posing challenges for setups with limited computational resources. In turn, Tseng et al. (2021) introduced a regularization technique for the discriminator's loss function, grounded in the LeCam divergence. In contrast to the empirical approach of Kumari et al. (2022), the method by Tseng et al. (2021) offers a solid theoretical foundation for stabilizing GAN training. Additionally, it is designed to complement data augmentation techniques, maximizing performance in scenarios of limited data availability.

In recent years, conditional scene generation has garnered significant attention from the image generation community, particularly in creating scenes with multiple objects. Various conditional image generation tasks have been developed using different subsets of annotations, including text-based image generation with natural language description of the scene (REED et al., 2016; ZHANG et al., 2018). These methods frequently employ datasets such as MS-COCO (LIN et al., 2014), which pair images with textual descriptions, thereby enabling models to learn the correspondence between language and visual content.

Other works use scene graphs as input and learn to generate layouts as intermediate representations (SYLVAIN et al., 2021), which utilize the Visual Genome dataset (KRISHNA et al., 2017) for training. This dataset provides detailed annotations of objects and their relationships within images, facilitating the learning of structured scene representations. Meanwhile, other methods focus on directly generating images from bounding box layouts (HINZ; HEINRICH; WERMTER, 2019). One advantage of generating from layouts is the controllability of the generation. This controllability enables the development of a tool for generation based on user design.

Bounding boxes layouts can enhance the information transmitted to the network. This in itself can mitigate data limitation. Therefore, in this dissertation, we propose a method capable of providing this information via bounding box modules, inducing the generation of the face as a whole.

3 FUNDAMENTALS

This chapter presents the fundamentals that underpin the proposal of this work. Section 3.1 details the multiple objects generation method based on bounding box modules that insert the spatial location information provided by bounding boxes into a GAN (HINZ; HEINRICH; WERMTER, 2019). Section 3.2 approaches the GANs theory through the BigGAN architecture (BROCK; DONAHUE; SIMONYAN, 2019). Lastly, Section 3.3 explains the regularization methods for GANs training under limited data (ZHAO et al., 2020; TSENG et al., 2021).

3.1 MULTIPLE OBJECTS GENERATION

Hinz, Heinrich and Wermter (2019) have introduced an image generation approach which allows to arbitrarily control the spatial location of objects within a scene by adding as inputs of a GAN only bounding boxes and their respective labels. Figure 2 shows an example of a scene from the MS-COCO dataset and its input configuration. An object is represented by a label l and a bounding box $\boldsymbol{b}_{box} = [x_{min}, y_{min}, w, h]$, where (x_{min}, y_{min}) are the coordinates of the top left corner, with width w and height h, normalized between 0 and 1.

Given a maximum number of objects o_{max}, a scene sets a dictionary $\boldsymbol{d}$ of fixed size o_{max} with bounding boxes indexed by their labels. The label $l = 0$ and the bounding box $\boldsymbol{b}_{box} = [-1, -1, -1, -1]$ are reserved to complete the dictionary $\boldsymbol{d}$ in scenes with a number of objects $o_{scene} < o_{max}$. Additionally, if available, a natural language caption can be provided describing the scene, e.g., “A woman, man, and a dog standing in the snow”. Subsection **3.1.1** presents the proposed customizations for a GAN to support these inputs, while Subsection **3.1.2** details how objects are manipulated through affine transformations parameterized by the coordinates of the bounding boxes.

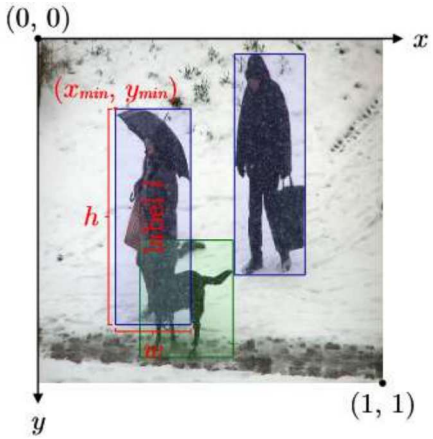


Figure 2 – Example of a scene from MS-COCO dataset and its input configuration.

Source: Created by the author (2024).

3.1.1 Global and object pathways

Both the generator and the discriminator are adapted into two modules, called global pathway and object pathway. The global pathway focuses on the scene layout and image background, while the object pathway focuses on individual objects. Figure 3 shows the global and object pathways method for the generator. Initially, the global pathway of the generator is responsible for creating a overall layout encoding of the scene. To do this, the label of each object is encoded as a one-hot binary vector $\boldsymbol{l}_{one\text{-}hot}$, where the digit 1 is assigned to the index of the respective object label l, while the other indexes are set to 0. The layout input is obtained by the spatial replication of each $\boldsymbol{l}_{one\text{-}hot}$ to $\boldsymbol{L}_{one\text{-}hot} \in \mathbb{R}^{h \times w \times c}$, given a resolution $h \times w$ and the number of channels $c = o_{data} + 1$ as function of the number of object classes o_{data} of the dataset, which are iteratively transformed and inserted into an empty canvas, initialized with zeros, at the locations given by their respective bounding boxes.

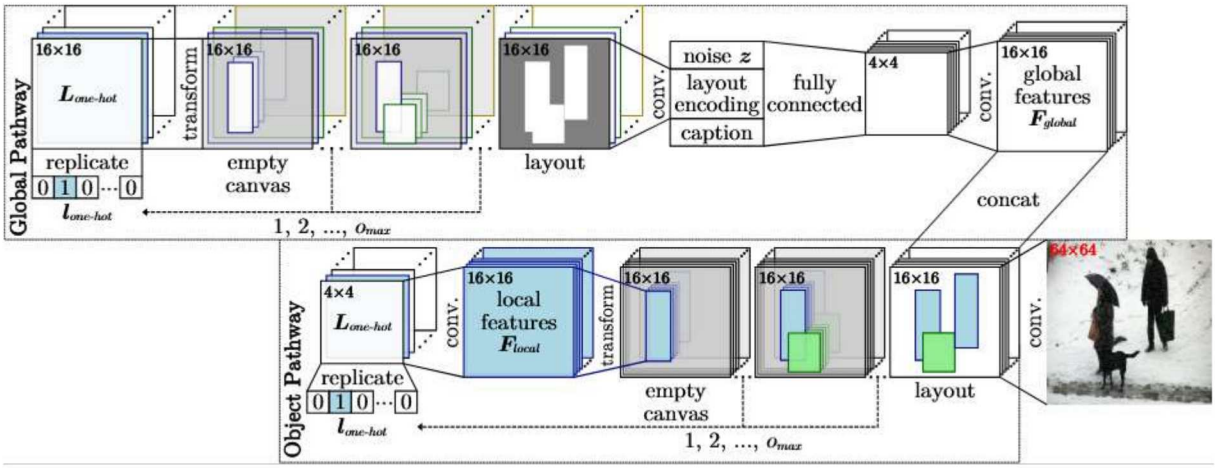


Figure 3 – Global and object pathways method for the generator.

Source: Created by the author (2024).

A series of convolutional layers are then applied in order to obtain a high-level latent representation of the layout encoding. Following this step, the layout encoding is concatenated with a random noise distribution $\boldsymbol{z}$ and, if available, a processed natural language caption of the scene. The output concatenation feeds a fully connected network, which in turn is reshaped as the input of further convolutional layers to generate a global feature map $\boldsymbol{F}_{global}$. The object pathway of the generator is responsible for generating a local feature map of the objects. In similar way to the global pathway, the one-hot binary vectors are spatially replicated at a predefined resolution. However, before being transformed and inserted into an empty canvas at the locations provided by their respective bounding boxes, convolutional layers are applied, creating high-level feature maps $\boldsymbol{F}_{local}$ inside the object regions, while areas outside of the bounding boxes remain zero. Finally, both the global and the object feature maps are concatenated and the result used by further convolutional layers to generate the image with its final resolution.

Figure 4 shows a global and object pathways method for the discriminator. The global pathway of the discriminator is not conditioned to the bounding box information. It takes as input only a real or a fake image, synthesized by the generator, and applies multiple convolutional layers to extract a global discriminative feature map $\boldsymbol{F}_{global}$. The first stage of the object pathway of the discriminator consists of extracting and transforming each object from within the region given by the bounding box and concatenating it with the respective spatially replicated one-hot binary vector $\boldsymbol{L}_{one\text{-}hot}$. Next, convolutional layers are applied, creating a local discriminative map $\boldsymbol{F}_{local}$. The features are newly transformed and reinserted into an empty canvas within the corresponding bounding box. Like the generator, the outputs of both global and object pathways are concatenated along the channel axis. New convolutional layers are applied in order to obtain a merged feature representation, which is concatenated with the sum of all one-hot binary vectors and, if available, a processed natural language caption of the scene, both spatially replicated at the same resolution as the merged representation. The output is then classified as real or fake.

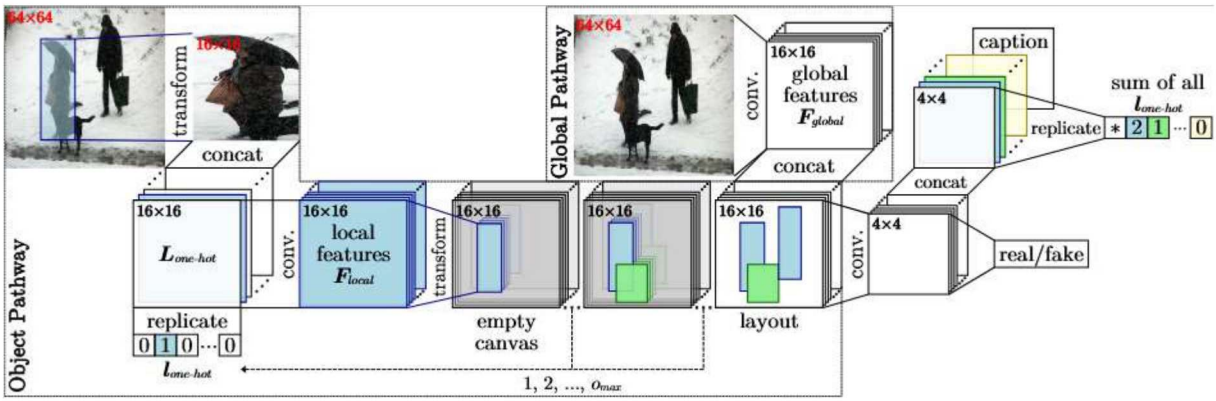


Figure 4 – Global and object pathways method for the discriminator. The initial position of the vector resulting from the sum of all $\boldsymbol{l}_{one\text{-}hot}$ receives a value (denoted by "$*$") equal to the difference between the maximum number of objects o_{max} and the number of objects in the scene o_{scene}.

Source: Created by the author (2024).

Hinz, Heinrich and Wermter (2019) used both the dataset MS-COCO that contains natural language descriptions of the scenes and GANs proposed for the text-to-image task (ZHANG et al., 2017; XU et al., 2018). Since the dataset used in this work does not have this type of annotation, and the BigGAN architecture is not designed to accept text information, the proposed method in this dissertation only takes as input the object label l and the bounding box $\boldsymbol{b}_{box}$.

3.1.2 Spatial Transformer Network

The insertion of spatially replicated one-hot binary vectors and high-level feature maps into empty canvas, as well as the extraction of objects from their regions given by

the coordinates of the bounding boxes, are carried out by a module based on a Spatial Transformer Network (STN) (JADERBERG et al., 2015). This module takes as input an image or a feature map $\boldsymbol{U} \in \mathbb{R}^{h \times w \times c}$ with height h, width w and number of channels c, and applies a spatial transformation T_θ conditioned on the particular input, where θ is the parameters, e.g., of an affine transformation matrix A_θ. Those parameters can be given as input or predicted through convolutional or fully connected networks. The output is set on a regular grid G, forming an feature map $\boldsymbol{V} \in \mathbb{R}^{h' \times w' \times c}$, where h' and w' are the new height and width, maintaining the same number c of channels.

The affine transformation case is expressed as

$$\begin{pmatrix} x_i^{\mathrm{s}} \\ y_i^{\mathrm{s}} \end{pmatrix} = T_\theta(G_i) = A_\theta \begin{pmatrix} x_i^{\mathrm{t}} \\ y_i^{\mathrm{t}} \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^{\mathrm{t}} \\ y_i^{\mathrm{t}} \\ 1 \end{pmatrix}, \tag{3.1}$$

where for each target coordinate $(x_i^{\mathrm{t}}, y_i^{\mathrm{t}})$ of the regular grid G in the output map, is determined from the corresponding source coordinate $(x_i^{\mathrm{s}}, y_i^{\mathrm{s}})$ sampled in the input map, both normalized between -1 and 1. To perform the spatial transformation of the input map $\boldsymbol{U}$, a sampling kernel is applied to get each particular pixel in the output map $\boldsymbol{V}$. This can be written as

$$\boldsymbol{V}_i^j = \sum_n^h \sum_m^w \boldsymbol{U}_{nm}^j \; \mathbf{k}(x_i^{\mathrm{s}} - m; \Phi_x) \; \mathbf{k}(y_i^{\mathrm{s}} - n; \Phi_y) \; \forall i \in [1 \ldots h'w'] \;\; \forall j \in [1 \ldots c], \tag{3.2}$$

where Φ_x and Φ_y are the parameters of a generic sampling kernel $\mathbf{k}$ (e.g. bilinear interpolation), $\boldsymbol{U}_{nm}^j$ is the value at location (n, m) in channel j of the input, and $\boldsymbol{V}_i^j$ is the value of the output pixel i at location $(x_i^{\mathrm{t}}, y_i^{\mathrm{t}})$ in channel j. In order to allow backpropagation of the loss function, it is possible to define the gradients with respect to $\boldsymbol{U}$ and G. For a bilinear sampling, the partial derivatives are

$$\frac{\partial \boldsymbol{V}_i^j}{\partial \boldsymbol{U}_{nm}^j} = \sum_n^h \sum_m^w \; \max(0, 1 - |x_i^{\mathrm{s}} - m|) \; \max(0, 1 - |y_i^{\mathrm{s}} - n|), \tag{3.3}$$

$$\frac{\partial \boldsymbol{V}_i^j}{\partial x_i^j} = \sum_n^h \sum_m^w \boldsymbol{U}_{nm}^j \; \max(0, 1 - |y_i^{\mathrm{s}} - n|) \begin{cases} 0 & \text{if } |m - x_i^{\mathrm{s}}| \geq 1, \\ 1 & \text{if } m \geq x_i^{\mathrm{s}}, \\ -1 & \text{if } m < x_i^{\mathrm{s}} \end{cases}, \tag{3.4}$$

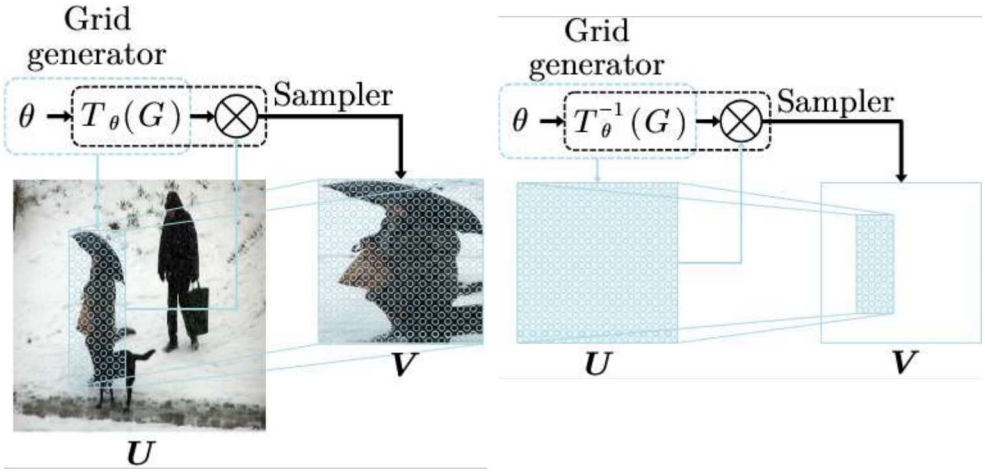
and similarly to Eq. 3.4 for $\frac{\partial \boldsymbol{V}_i^j}{\partial y_i^{\mathrm{s}}}$.

The effects of the affine transformations applied in the global and object pathways presented in the previous Subsection are shown in Figure 5. The composite transformation matrix

$$A_\theta = \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} w & 0 & 2((x_{min} + \frac{1}{2}w) - \frac{1}{2}) \\ 0 & h & 2((y_{min} + \frac{1}{2}h) - \frac{1}{2}) \\ 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

allows translation (t_x and t_y therms) and non-uniform scaling (s_x and s_y therms), as a function of the bounding box $\boldsymbol{b}_{box}$, defined in the introduction of this section. In the object pathway of the discriminator, the input map $\boldsymbol{U}$ represents a real or a synthetic image, and the output map $\boldsymbol{V}$ the transformed image of current object (Figure 5a). In the global and object pathways of the generator, the input feature map $\boldsymbol{U}$ represents the spatially replicated one-hot binary vector, and the output feature map $\boldsymbol{V}$ the partial layout of the scene (Figure 5b), obtained by the inverse matrix

$$A_\theta^{-1} = \begin{bmatrix} \frac{1}{s_x} & 0 & -\frac{t_x}{s_x} \\ 0 & \frac{1}{s_y} & -\frac{t_y}{s_y} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{w} & 0 & 2\frac{1}{w}((\frac{1}{2} - (x_{min} + \frac{1}{2}w)) \\ 0 & \frac{1}{h} & 2\frac{1}{h}((\frac{1}{2} - (y_{min} + \frac{1}{2}h)) \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.6}$$



(a) $T_\theta(G)$ (b) $T_\theta^{-1}(G)$

Figure 5 – Parameterised sampling grid G applied to an input map $\boldsymbol{U}$ producing the output map $\boldsymbol{V}$. Note that the transformations are inversible.

Source: Created by the author (2024).

3.2 BIGGAN

The modeling proposed by Goodfellow et al. (2014) involves training two adversarial networks, the generator (**G**) and the discriminator (**D**). While **D** is tasked with distinguishing its input data between real and fake, **G** learns to map random noise to data in order to maximize the probability of **D** making a mistake. Formally, the original objective function of Goodfellow et al. (2014) puts **D** and **G** in a Minimax problem

$$\min_{\mathbf{G}} \max_{\mathbf{D}} \; \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})}[\log \mathbf{D}(\mathbf{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - \mathbf{D}(\mathbf{G}(\boldsymbol{z})))], \tag{3.7}$$

where $\mathbf{D}(\mathbf{x})$ returns the probability of $\mathbf{x}$ belonging to the training set distribution p_{data} instead of the generator p_z. Both networks are trained simultaneously, however optimizing **D** every step is computationally prohibitive, and on limited datasets can lead to overfitting. Instead, **D** is optimized at each k steps, keeping the **G** optimization for each single step.

Proposed by Brock, Donahue and Simonyan (2019), the BigGAN belongs to a group of models known as conditional GANs (MIRZA; OSINDERO, 2014), which extend the objective function (Eq. 3.7):

$$\min_{\mathbf{G}} \max_{\mathbf{D}} \; \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log \mathbf{D}(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - \mathbf{D}(\mathbf{G}(\boldsymbol{z}|\mathbf{y})))], \tag{3.8}$$

by conditioning both $\mathbf{D}$ and $\mathbf{G}$ on some extra information $\mathbf{y}$, like sample class label. The BigGAN architecture for 128×128 resolution is shown in Figure 6. The embedding layer (yellow box), initialized from a $\mathcal{N}(0, 1)$ distribution, learns to represent each class label of the dataset in a weight vector $\boldsymbol{y}_{embedding}$. The noise vector $\boldsymbol{z}$ (blue box), drawn from distribution p_z such as $\mathcal{N}(0, 1)$ or $\mathcal{U}(-1, 1)$, is split into consecutive parts of equal size. The number of splits is determined by the tensors' resolutions along the generator, i.e., 4×4, 8×8, 16×16, 32×32, 64×64 and 128×128 (six splits).

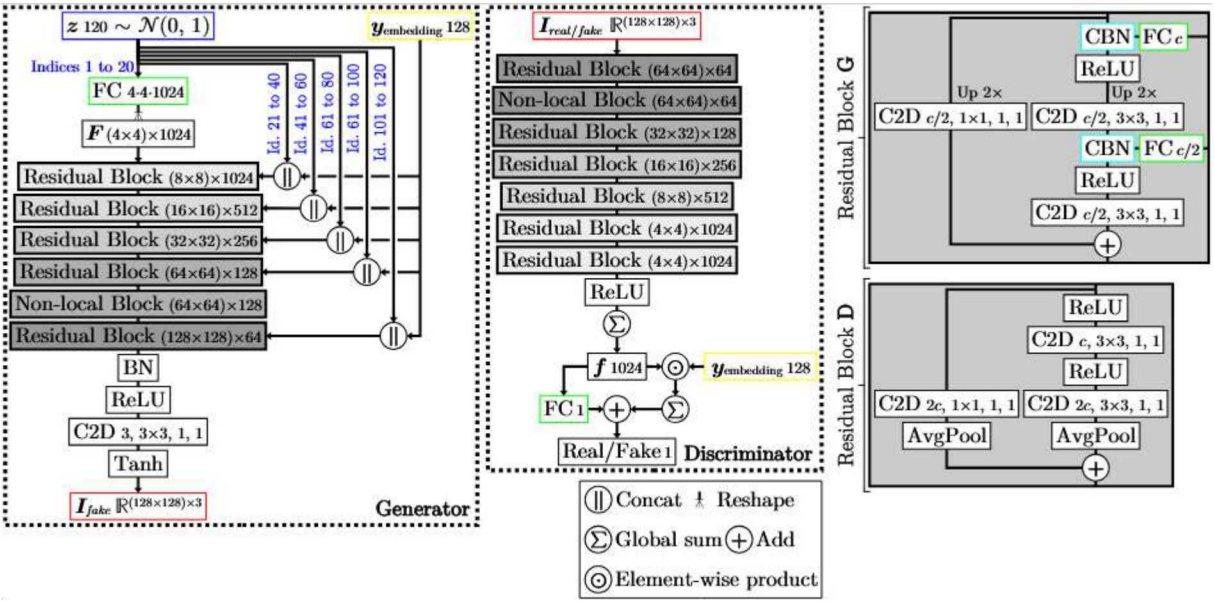


Figure 6 – BigGAN architecture for 128×128 resolution. Where it reads "Indices x to y" refers to the noise vector $\boldsymbol{z}$ partition that feeds that forward.

Source: Created by the author (2024).

The first part (indices 1 to 20) feeds the initial fully connected (FC) layer (green box). The other parts (indices 21 to 40, indices 41 to 60, etc.), after concatenated (denoted by "||") with the shared $\boldsymbol{y}_{embedding}$, feeds skip connections to deeper FC layers in the $\mathbf{G}$ residual blocks. The conditional batch normalization (CBN) layers (cyan box) allow its gain and bias parameters to establish specific patterns according to the FC layers outputs. This design enables the $\boldsymbol{z}$ and $\boldsymbol{y}_{embedding}$ vectors to directly influence features at different resolutions.

The reshaped feature map $\boldsymbol{F}$ moves forward through $\mathbf{G}$ via residual blocks containing 2-dimensional convolutional (C2D number of output channels, kernel size, stride, padding) layers with the Rectified Linear Unit (ReLU) activation function, as well as CBN layers.

Before the initial and skip convolutional layers, which are enabled to reduce the number of channels from c to $c/2$, there is an upsampling by a factor of 2 using an interpolation function. The architecture also includes Non-local (ZHANG et al., 2019) blocks, capable of capturing interactions between distant pixels. Finally, the resulting tensor with the desired resolution goes through another ReLU activation function, followed by a batch normalization (BN) layer, before undergoing a final reduction in the number of channels to $c = 3$ by a C2D layer, and normalization with the hyperbolic tangent (Tanh), generating the fake image $\boldsymbol{I}_{fake}$ (red box).

In turn, **D** has a more simplified architecture, moving forward a real or fake image $\boldsymbol{I}_{real/fake}$ (red box) through residual blocks containing C2D with ReLU. Within each residual block, after the final and skip convolutional layers, which are enabled to increase the number of channels from c to $2c$, there is a downsampling by a factor of 2 using average pooling (AvgPool) layers. In the last residual block, as the resolution of the output tensor is not reduced, there are no AvgPool layers. The resulting feature map undergoes a final downsampling using a global sum (denoted by "$\sum$"), obtaining a feature vector $\boldsymbol{f}$, which then feeds two flows. One extracts a single value from $\boldsymbol{f}$ using an FC layer, and the other performs the element-wise product (denoted by "$\odot$") between $\boldsymbol{f}$ and $\boldsymbol{y}_{embedding}$, using a global sum to also extracts a single value from the resulting vector. Both values are added (denoted by "+") together, giving the probability of $\boldsymbol{I}_{real/fake}$ being real (positive) or fake (negative).

Brock, Donahue and Simonyan (2019) explored the trade-off between sample fidelity and variety for a given **G**. Taking a model trained with $\boldsymbol{z} \sim \mathcal{N}(0, \sigma^2)$, where σ^2 is the variance, it is possible to use a different latent distribution for sampling. By truncating the $\boldsymbol{z}$ vector resampling the values with variance above a chosen threshold, there is an improvement in the individual sample quality at the cost of reducing the overall variety. This procedure is called Truncation Trick. However, this distribution shift in the latent space can produce saturation artifacts in the generated samples. Brock, Donahue and Simonyan (2019) increased the chances to avoid this effect by stimulating orthogonality in the **G** and **D** weights through the following regularization condition $\mathrm{R}_\beta(W)$ modified from Brock et al. (2016):

$$\mathrm{R}_\beta(W) = \beta \|WW^\top \odot (J - I)\|^2, \tag{3.9}$$

where W is the weight matrix, β a penalty hyperparameter, I the identity matrix and J a all-ones matrix.

A contribution of this dissertation is the adaptation of the BigGAN architecture into bounding box modules called global and object pathways, detailed in the Section 3.1. It is important to note that the peculiarities of this architecture, such as the skip connections supplied by the concatenations of the $\boldsymbol{z}$ chunks and the class embedding $\boldsymbol{y}$, allow new configurations of the global and object pathways.

3.3 DIFFERENTIABLE AUGMENTATION AND LECAM-DIVERGENCE LOSS

A limited training dataset can deteriorate the GAN performance, since the discriminator tends to memorize the training set. An overfitted discriminator penalizes excessively any generated sample that is not exact to the real data distribution, leading to poor generalization. Zhao et al. (2020) proposed a simple method that improves data efficiency through data augmentation applied not only on the real samples $\mathbf{x}$, but also on the fake $\mathbf{G}(\boldsymbol{z})$ ones, in both $\mathbf{D}$ and $\mathbf{G}$ updates (Figure 7), optimized by the discriminator $\mathbf{V_D}$ and generator $\mathbf{L_G}$ training objectives, given loss functions $\mathbf{f_D}$ and $\mathbf{f_G}$:

$$\max_{\mathbf{D}} \mathbf{V_D} = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\mathbf{f_D}(\mathbf{D}(\mathbf{T}(\mathbf{x})^{(\mathrm{i})}))] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\mathbf{f_D}(\mathbf{D}(\mathbf{T}(\mathbf{G}(\boldsymbol{z}))^{(\mathrm{ii})}))], \tag{3.10}$$

$$\min_{\mathbf{G}} \mathbf{L_G} = \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\mathbf{f_G}(\mathbf{D}(\mathbf{T}(\mathbf{G}(\boldsymbol{z}))^{(\mathrm{iii})}))], \tag{3.11}$$

where $\mathbf{T}$ represents the transformations in the data augmentation, such as translation, cutout and color. The $\mathbf{G}$ update (iii) requires $\mathbf{T}$ to be differentiable as gradients should be backpropagated through $\mathbf{T}$ to $\mathbf{G}$.

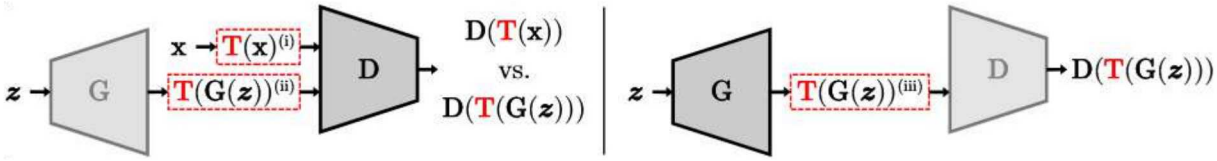


Figure 7 – Differentiable Augmentation for $\mathbf{D}$ (left) and $\mathbf{G}$ (right) updates.

Source: Created by the author (2024).

Tseng et al. (2021) proposed a regularized discriminator loss that improves the performance of GANs learning dynamics especially under limited training data. They introduced a $\ell 2$ norm between the current real and fake image predictions and moving average variables called anchors that tracks the historical discriminator predictions. Based on Eq. 3.10:

$$\min_{\mathbf{D}} \mathbf{L_D} = -\mathbf{V_D} + \lambda \mathrm{R_{LC}}, \tag{3.12}$$

the regularization term $\mathrm{R_{LC}}$ is written as:

$$\mathrm{R_{LC}} = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\|(\mathbf{D}(\mathbf{x}) - \boldsymbol{\alpha}_{\boldsymbol{I}_f}\|^2] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\|(\mathbf{D}(\mathbf{G}(\boldsymbol{z})) - \boldsymbol{\alpha}_{\boldsymbol{I}_r}\|^2], \tag{3.13}$$

where $\boldsymbol{\alpha}_{\boldsymbol{I}_r}$ and $\boldsymbol{\alpha}_{\boldsymbol{I}_f}$ are the anchors for the real and fake images, respectively, and λ an empirically adjustable hyperparameter. At first glance, although it appears counterintuitive, the regularization term $\mathrm{R_{LC}}$ encourages $\mathbf{D}$ to mix the predictions between real and generated images. Tseng et al. (2021) demonstrated that $\mathrm{R_{LC}}$ can enforce the minimization of the weighted divergence called LeCam-divergence, which plays a crucial role in GANs since it is an underlying measure to align the data distribution p_{data} and the generated distribution p_z.

The simultaneous application of the Differentiable Augmentation and the LeCam-Divergence Loss is crucial for the convergence of the original BigGAN architecture trained on the dataset used in this work. An important contribution of this dissertation is the non-dependence of both methods for the convergence of the adapted BigGAN architecture into global and object pathways, even though they contribute to improving data efficiency.

4 PROPOSED METHOD

This chapter introduces the main contribution of this dissertation, which consists of the integration between the bounding box modules, called global and object pathways, and the BigGAN architecture, both presented in the Chapter 3. First, Section 4.1 presents a multiscale boxes proposal to represent the objects. Next, Section 4.2 describes our proposal for the BigGAN generator architecture, while Section 4.3 describes our proposal for the BigGAN discriminator architecture.

4.1 MULTISCALE BOXES

Instead of spatially locating an object by a single bounding box $\boldsymbol{b}_{box}$, according to the Section 3.1, this work proposes the use of multiscale boxes. Figure 8 shows an example of a scene from MS-COCO dataset and its input configuration with multiscale boxes. A scene stores a dictionary $\boldsymbol{d}$ of fixed size o_{max} (maximum number of objects) where each element, indexed by a label l, becomes a set of b concentric bounding boxes with the same aspect ratio. Given a scale increase factor s, a bounding box is defined by $\boldsymbol{b}_{box} = [x_{min} - (j-1)\frac{s}{2}w, y_{min} - (j-1)\frac{s}{2}h, w + (j-1)sw, h + (j-1)sh]$, where $j = 1, 2, ..., b$. When $j = 1$, we get the original bounding box with coordinates (x_{min}, y_{min}, w, h).

In this way, the transformations applied to create the layouts in the global and the object pathways of both the generator and the discriminator are iterated through a nested loop $o_{max} \times b$, where $i = 1, 2, ..., o_{max}$ and $j = 1, 2, ..., b$. The intuition behind the use of bounding boxes with multiple scales is to smooth out the transition regions between the objects and the background. Furthermore, as the overlapping areas are added together, the objects' features are reinforced.

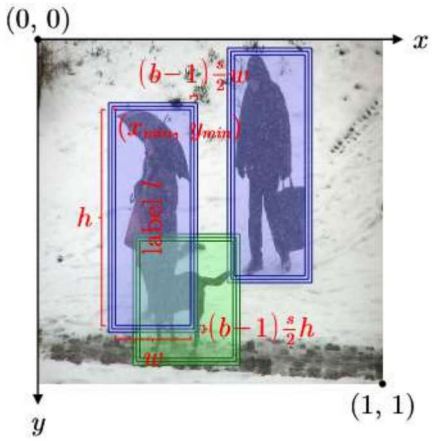


Figure 8 – Example of a scene from MS-COCO dataset and its input configuration with multiscale boxes.

Source: Created by the author (2024).

4.2 PROPOSED GENERATOR

Our customization proposal for the BigGAN generator architecture is shown in Figure 9. The integrated modules are highlighted in dotted boxes with specific colors. The processing pipeline for these modules are maintained as proposed by Hinz, Heinrich and Wermter (2019). In this proposal, a sampling grid G is applied to an input map represented by a spatially replicated one-hot binary vector $\boldsymbol{L}_{one\text{-}hot}$ or a local feature map $\boldsymbol{F}_{local}$. The sampler (denoted by "⊗") performs an inverse affine transformation T_θ^{-1} parameterized by a bounding box $\boldsymbol{b}_{box}$, generating an output map. The iterative sum of each output map with an empty canvas creates the layout. More details are described in Subsections **3.1.1** and **3.1.2**. Our proposed modification comes from the use of multiscale boxes (described in the previous Section), which inserts an extra loop (dotted rounded arrow) into the layout creation.

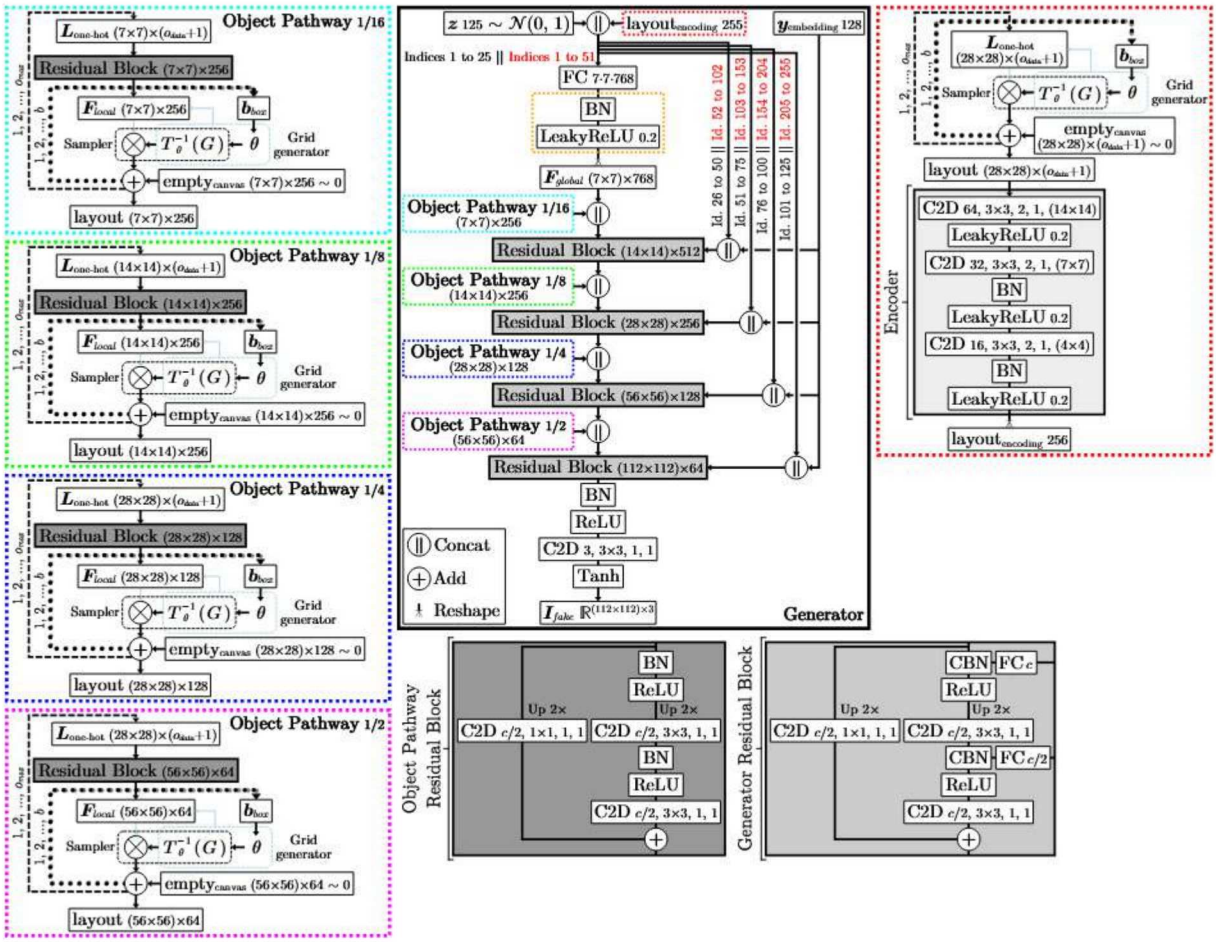


Figure 9 – Customization proposal for the BigGAN generator architecture. Where it reads "Indices x to y" refers to the vector partition (in black the noise vector $\boldsymbol{z}$, and in red the layout encoding) that feeds that forward.

Source: Created by the author (2024).

For this work, the encoder architecture adopted by Hinz, Heinrich and Wermter (2019) is also maintained. Three C2D layers of stride 2 and padding 1 perform consecutive

downsamplings, while obtaining a high-level latent representation of the layout. Leaky Rectified Linear Unit (LeakyReLU) activation functions amplify the range of convolutional layers weights to negative values normalized by a slope coefficient of 0.2, and BN layers follow the last two C2D layers.

The layout encoding (red dotted box) is inserted into the network input. We propose to split it into consecutive parts of equal size as Brock, Donahue and Simonyan (2019) have proposed for the noise vector $\boldsymbol{z}$. The number of splits is determined by the tensors' resolutions along the generator, i.e., 7×7, 14×14, 28×28, 56×56 and 112×112 (five splits). The encoder produces a 256-element layout. In turn, we opted to initialize the noise vector $\boldsymbol{z}$ with 128-elements, half of the layout encoding. In order to obtain five parts of equal size, the last layout encoding index and the last three $\boldsymbol{z}$ indexes are discarded.

The initial FC layer receives the first part of the noise vector $\boldsymbol{z}$ (indices 1 to 25) concatenated with the first part of the layout encoding (indices 1 to 51). The remaining parts of $\boldsymbol{z}$ and the layout encoding are also respectively concatenated with each other (indices 26 to 50 || indices 52 to 102, indices 51 to 75 || indices 103 to 153, etc.) and with the shared class weight vector $\boldsymbol{y}_{embedding}$. The resulting vector with 208 elements feeds deeper FC layers into residual blocks. We add to the initial FC layer the same normalization of the last two C2D layers of the encoder, i.e, BN followed by LeakyReLU (orange dotted box). The reshaped output then establishes the global feature map $\boldsymbol{F}_{global}$. Due to the skip connections, the global network inputs also directly influence the features of different resolutions along the generator.

The approach proposed by Hinz, Heinrich and Wermter (2019) establishes a single object pathway. Its output layout has a resolution of 1/4 of the fake image $\boldsymbol{I}_{fake}$ resolution, and is generated from spatially replicated one-hot binary vectors $\boldsymbol{L}_{one\text{-}hot}$ with 1/16 of the $\boldsymbol{I}_{fake}$ resolution. This configuration can lead to objects being underrepresented, especially those at smaller scales, depending on the network final resolution. In this work, we propose multiple object pathways, whose output layouts form a resolution pyramid (1/2, 1/4, 1/8 and 1/16 of the $\boldsymbol{I}_{fake}$ resolution). In this way, the objects' features are reinforced from the input resolution up to highest resolutions capable to capture more information. Figure 10 compares the inputs and outputs resolutions of the object pathways proposed by this work with the object pathway proposed by Hinz, Heinrich and Wermter (2019) (shaded layout).

The output layout of the first object pathway (cyan dotted box) is concatenated with the global feature map $\boldsymbol{F}_{global}$. The output layouts of the two middle object pathways (green and blue dotted boxes) are concatenated with the output tensors of the first two generator residual blocks. The output layout of the last object pathway (magenta dotted box) is concatenated with the output tensor of the penultimate generator residual block. Note that the first three object pathways preserve their $\boldsymbol{L}_{one\text{-}hot}$ resolutions. On the other hand, the last object pathway obtains local feature maps $\boldsymbol{F}_{local}$ with twice of their $\boldsymbol{L}_{one\text{-}hot}$

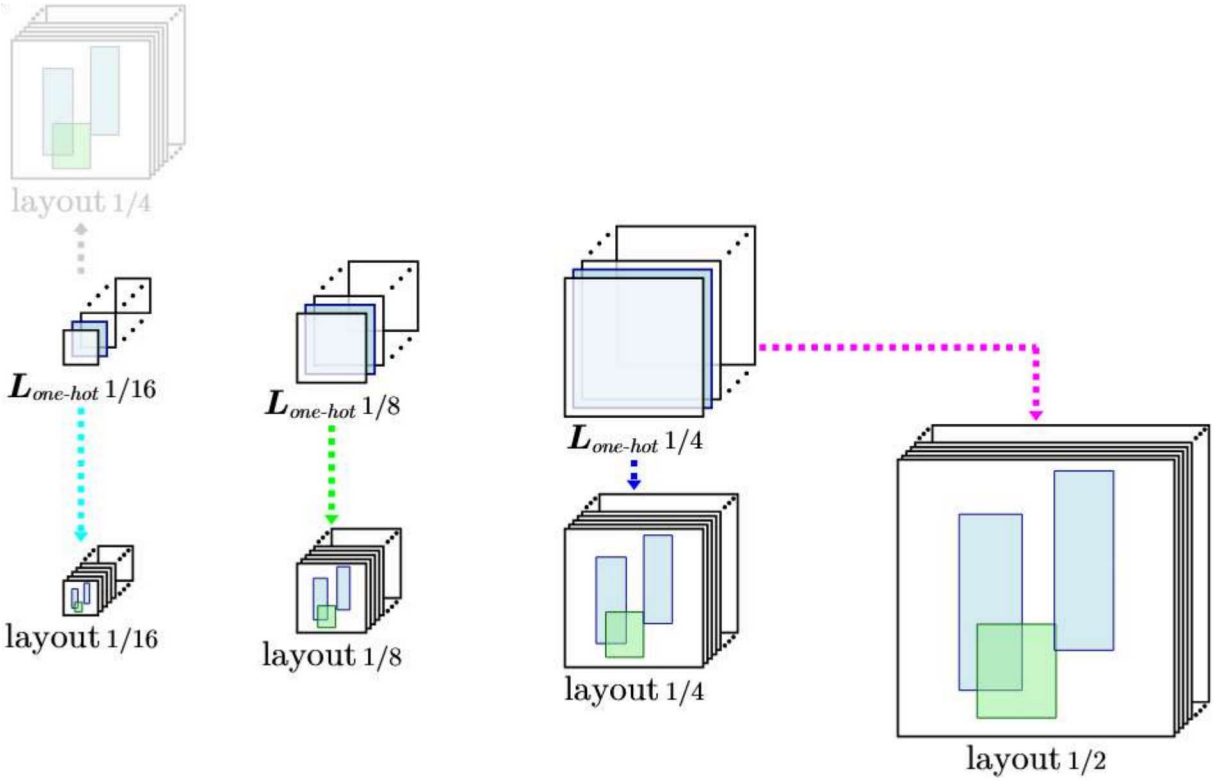


Figure 10 – Comparison of the input and output resolutions of the object pathways proposed by this work and the one proposed by Hinz, Heinrich and Wermter (2019) (shaded layout).

Source: Created by the author (2024).

resolution. The idea is to prevent that the upsampling occurs for the lowest network resolutions and, when applied, occurs by a single factor of 2×. Unlike the generator's residual blocks, the object pathways' residual blocks are not conditioned to the $\boldsymbol{y}_{embedding}$. Consequently, they have no FC layers, and the CBN layers are replaced by BN layers. Finally, after the last residual block, the generator follows its original configuration, generating the fake image $\boldsymbol{I}_{fake}$.

Brock, Donahue and Simonyan (2019) proposed different depth – number of residual blocks – and width – number of output channels of the residual blocks – configurations for 128 × 128, 256 × 256 and 512 × 512 resolutions. Due to the particularities of the dataset explored in this work (detailed in the Section 5.1), and the computational resources available, we opted to fix the resolution at 112 × 112. In this sense, based on the original BigGAN architecture for 128 × 128 resolution (Figure 6), we propose a configuration with one less residual block in both the generator and the discriminator.

4.3 PROPOSED DISCRIMINATOR

Our customization proposal for the BigGAN discriminator architecture, shown in Figure 11, differs in two main aspects from the customization proposal for the BigGAN

generator architecture, described in the previous section. The first one is the maintenance of a single object pathway. The second one is the global feature map $\boldsymbol{F}_{global}$ resolution, i.e., from 1/16 to 1/4 of the input image $\boldsymbol{I}_{real/fake}$ resolution. Both aspects preserve the structure proposed by Hinz, Heinrich and Wermter (2019) for the discriminator. The aim is to keep the discriminator architecture simpler than the generator. Adversarial training involves the progressive adaptation of one network to another. So, if a discriminator is too difficult to be deceived, the generator won't know how to update its weights.

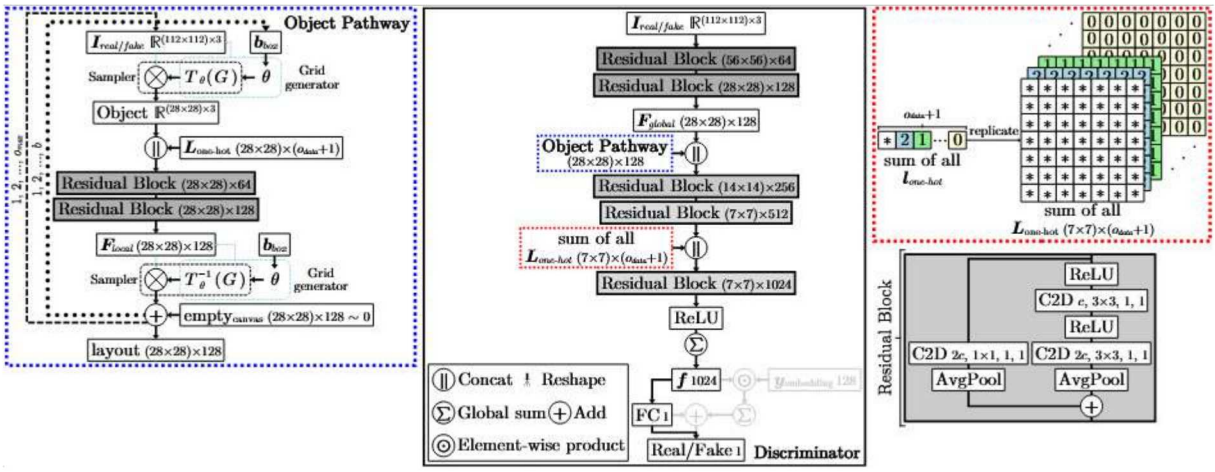


Figure 11 – Customization proposal for the BigGAN discriminator architecture. o_{data} is the number of object classes of the dataset. The initial position of the vector resulting from the sum of all one-hot binary vectors $\boldsymbol{l}_{one\text{-}hot}$ receives a value (denoted by "$*$") equal to the difference between the maximum number of objects o_{max} and the number of objects in the scene o_{scene}. The class weights vector $\boldsymbol{y}_{embedding}$ incorporation flow was excluded (shaded forward).

Source: Created by the author (2024).

The object pathway processing pipeline (blue dotted box) is maintained as proposed by Hinz, Heinrich and Wermter (2019). In this proposal, there are two stages. In the first one, a sampling grid G is applied to an input map represented by a real or a fake image $\boldsymbol{I}_{real/fake}$. The sampler (denoted by "$\otimes$") then performs an affine transformation T_θ parameterized by a bounding box $\boldsymbol{b}_{box}$, extracting the interest object from the scene with 1/4 of the input image resolution.

In the second stage, the extracted object is concatenated with its respective spatially replicated one-hot binary vector $\boldsymbol{l}_{one\text{-}hot}$. From the resulting tensor, two residual blocks extract a local feature map $\boldsymbol{F}_{local}$, preserving the resolution (no AvgPool layers). The number of channels of those two residual blocks corresponds to the two residual blocks that extract the global feature map $\boldsymbol{F}_{global}$ in the generator. The output layout is created just like in the generator object pathways. More details are described in Subsections **3.1.1** and **3.1.2**. Our proposed modification comes from the use of multiscale boxes (described in the previous section), which inserts an extra loop (dotted rounded arrow) into the whole process.

After the concatenation between the global feature map $\boldsymbol{F}_{global}$ and the object pathway output layout, two residual blocks are applied in order to obtain a merged feature representation. The resulting tensor is concatenated with the spatially replicated sum of all one-hot binary vectors $\boldsymbol{l}_{one\text{-}hot}$. A new residual block is performed, followed by a ReLU activation function. A global sum (denoted by "$\sum$") of all the elements from the same channel obtains a feature vector $\boldsymbol{f}$. Finally, a FC layer extracts from $\boldsymbol{f}$ the probability of $\boldsymbol{I}_{real/fake}$ being real or fake.

In the original BigGAN discriminator architecture, proposed by Brock, Donahue and Simonyan (2019), $\boldsymbol{f}$ feeds a second flow, responsible to incorporate the class weights vector $\boldsymbol{y}_{embedding}$. In our work, the sample class annotation is obtained through a conversion of the sum of all $\boldsymbol{l}_{one\text{-}hot}$, described in the Section 5.1. Experimentally, the spatial replication of this sum, concatenated with the penultimate residual block output, proved to be more efficient. Therefore, $\boldsymbol{y}_{embedding}$ flow is excluded (shaded forward in Figure 11).

5 EXPERIMENTAL RESULTS

This chapter presents the experiments to evaluate the proposed method both quantitatively and qualitatively. Section 5.1 describes the dataset used in this work. Section 5.2 describes the hyperparameters' setting and the training protocol. Section 5.3 presents the quantitative analysis and Section 5.4 the qualitative analysis of the results. Finally, in the Section 5.5, there is a general discussion about the results.

5.1 DATASET

This work uses a dataset created in the context of the project entitled Happy Cow ID as part of the Digital Zootechnical Residency program, a Brazilian Agricultural Research Corporation (Embrapa – Dairy Cattle) initiative in collaboration with the Federal University of Juiz de Fora (UFJF) and other Higher Education Federal Institutions. The dataset contains bovine images, where each face and interest subregion is spatially localized by a bounding box and annotated by a label. The labels are: cow (face), muzzle, left ear, right ear, left eye and right eye (subregions). The images have different aspect ratios, multiple individuals, wide scale, rotation, background, illumination and sharpness variation, as well as occlusion situations. Originally designed for the object detection task, we adapted the dataset for the facial generation task.

Figure 12 describes the pre-processing steps (gray boxes) applied to adapt the dataset for the facial generation task. The aim is to extract the faces, preserving only the annotations of their respective interest subregions. The figure shows an example selected from the original dataset ("cow_3061") that involves bounding boxes distributions that are important to point out. Notice that some face bounding boxes (drawn in black) do not fully encompass the bounding boxes of their respective interest subregions (drawn in the other colors). Another perceptible condition is the intersection of bounding boxes of interest subregions with more than one face bounding box. In addition, some interest subregions are not linked to any face region. Those situations, often found along the original dataset, combined with the wide pose variation, make it difficult to build an algorithm able to automatically determine which interest subregions belong to which faces, if they do. For this reason, the pre-processing is carried out semi-automatically.

In the facial selection step, each facial region is detached from the source image along with the interest subregions whose bounding boxes intersect the respective face bounding box. A manual analysis then checks whether each particular interest subregion matches the current face, in order to define the facial layout. If so, the respective interest subregion is allocated to the face in question. Otherwise, it is disregarded from the current check. With the facial layout defined, each facial region is extended following three criteria:

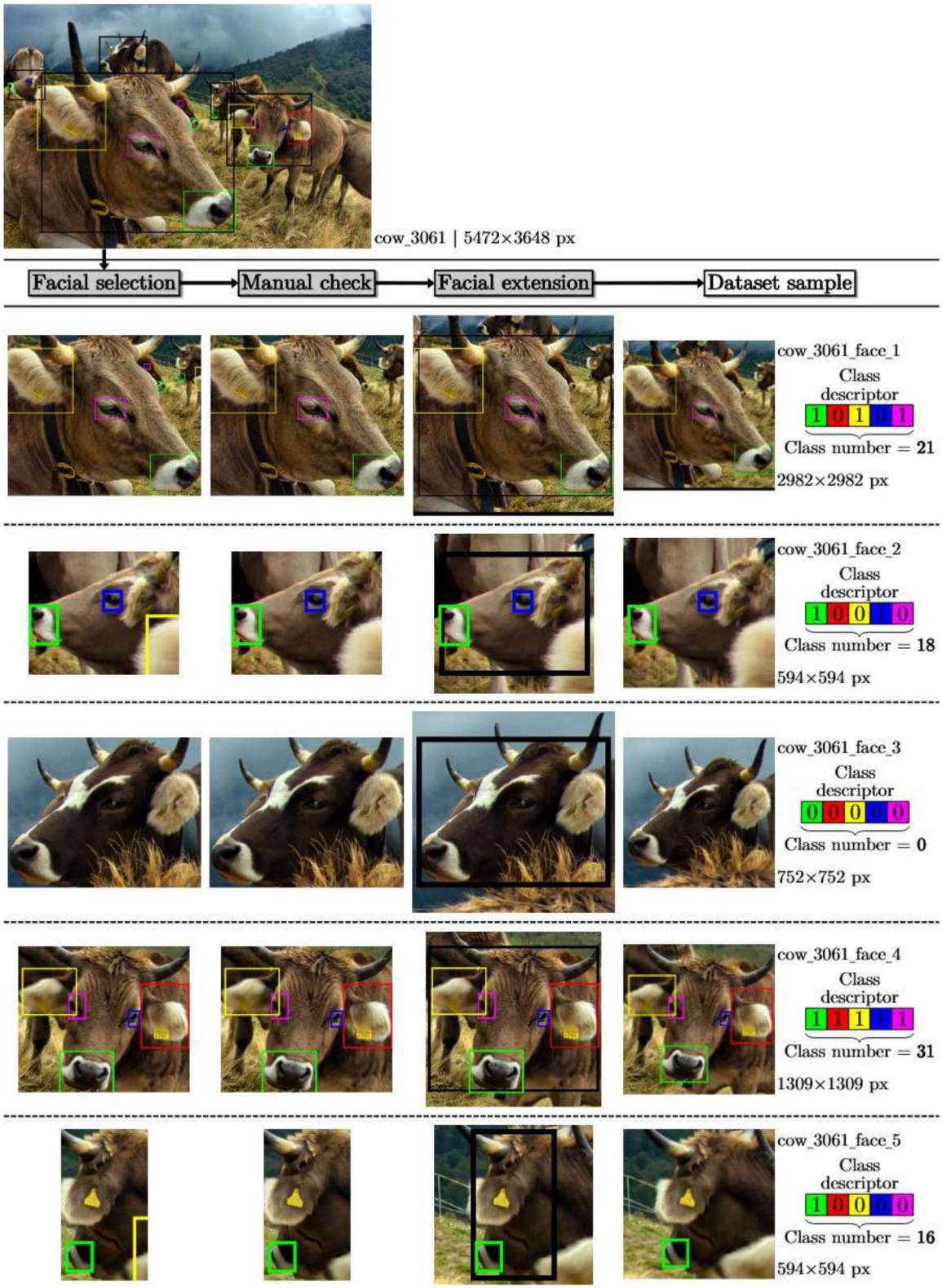


Figure 12 – Pre-processing steps (gray boxes) applied to adapt the dataset for the facial generation task. The outputs of each step are displayed in columns, and the rows show the progress of the steps for each face detached from the source image. The facial layout provides a binary descriptor that, when converted to decimal, represented the image class.

Source: Created by the author (2024).

1. Fully encompass the interest subregions. One or more coordinates of the face bounding box $(x_{min}, y_{min}, x_{max}, y_{max})$ can be adjusted to the coordinates of a interest subregion bounding box $(x'_{min}, y'_{min}, x'_{max}, y'_{max})$, if $x'_{min} < x_{min}$, $y'_{min} < y_{min}$, $x'_{max} > x_{max}$ or $y'_{max} > y_{max}$.

2. Stretch the face bounding box. The intention is to accentuate the face borders where they could be over-adjusted, without drastically interfering on the scale. Each direction is extended by up to 16 pixels, i.e., $(x_{min}-16, y_{min}-16, x_{max}+16, y_{max}+16)$, respecting the image boundaries.

3. Set the aspect ratio of the face region to 1. With the face bounding box centered, the lowest dimension is filled with background regions. If it exceeds the image boundaries, a zero padding is applied.

Following the same representation described in Subsection **3.1.1**, the label of each bounding box allocated to a facial layout is encoded as a one-hot binary vector $\boldsymbol{l}_{one\text{-}hot}$, where the bit 1 is assigned to the index of the respective interest subregion label l, while the other indices are set to bit 0. The bit-by-bit sum of all $\boldsymbol{l}_{one\text{-}hot}$ of a facial layout provides a number on the binary base that, when converted to the decimal base, represents the image class. For example, in the Figure 12, the facial layout configuration of the sample "cow_3061_face_1" corresponds to the number 10101 on the binary base, 21 to the decimal base. In this way, we can insert the class information into the BigGAN, via weights vector $\boldsymbol{y}_{embedding}$, as described in the Section 3.2.

In total, two datasets were pre-processed. The first one contains $6,182$ images collected through web searching, resulting in $9,495$ face images distributed in 32 classes. The second one contains 500 images collected manually on farms, resulting in 939 face images also distributed in 32 classes. The first dataset was used to train the models and quantitatively evaluate their performance. In turn, the facial layouts of the second dataset were used in the qualitative analyses.

Table 1 presents the number of matched and mismatched instances of the interest subregions after pre-processing, as well as their frequencies in the new dataset (percentage of face images containing that subregion), and their average resolutions. Proportionally, the mismatched instances are minimal compared to the matched ones. The ears and eyes have lower frequencies than the muzzle in the dataset, mainly due to pose variations. In addition to smaller frequencies, the eyes have the lowest average resolutions. The average resolution of the web searching and the manual collection datasets face images is 729.61×729.61 pixels and 744.04×744.04 pixels, respectively.

Figure 13 shows the number of images per class. The classes are arranged according to the number of interest subregions. The legend displays the proportion of images per number of interest subregions. Analyzing the web searching dataset, with 40.51%, the

Table 1 – Information about the interest subregions after pre-processing.

	Label	**Matched instances**	**Mismatched instances**	**Dataset frequency (%)**	**Average width (px)**	**Average height (px)**
Web searching	muzzle	8622	71	90.81	216.15	168.73
	left ear	7386	197	77.79	205.76	184.22
	right ear	7359	168	77.50	218.56	191.96
	left eye	6145	34	64.72	76.69	88.19
	right eye	6285	51	66.19	82.20	93.66
Manual collection	muzzle	791	8	84.24	224.88	164.06
	left ear	652	22	69.44	245.09	206.84
	right ear	620	15	66.03	267.70	224.47
	left eye	485	4	51.65	89.65	102.83
	right eye	473	4	50.37	100.88	115.91

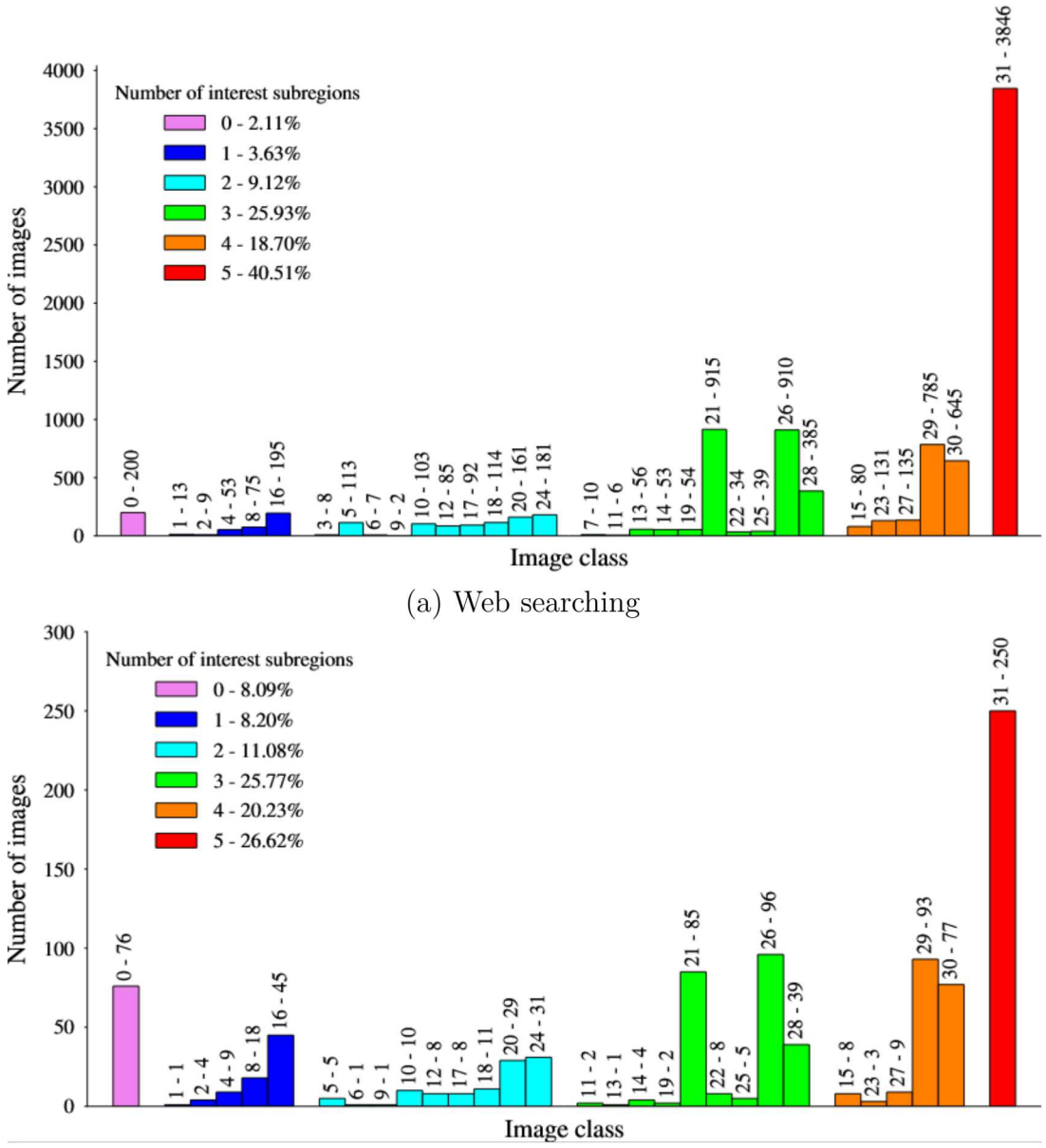
Source: Created by the author (2024).

image class 31, which contains all five interest subregions (frontal pose), is the most significant, even though most of the dataset contains four or less interest subregions. Next, the most representative image classes are 21 and 26, typified by the visibility of one side of the face (lateral poses), including the muzzle. Two other notable image classes are 29 and 30, characterized by a partial horizontal inclination that allows the annotation of both ears, but obstructs the simultaneous visibility of both eyes, also including the muzzle. Those five image classes (31, 30, 29, 26 and 21) concentrate 74.79% of the dataset. Note that the manual collection dataset follows a similar distribution, although with a smaller deviation.

Table 2 presents a statistical analysis of the upsizing conditions of the web searching dataset images for some specific resolutions. Upsizing should be avoided as it generates artifacts in the image. Note that higher target resolutions require considerable proportions of upsized images. In addition, the difference between the average resolution of the images to be upsized and the target resolution increases significantly. Under those conditions, we decided to apply a reasonable resolution size, where both the proportion and average resolution of upsized images are tolerable (line highlighted in gray).

5.2 TRAINING PROTOCOL AND HYPERPARAMETERS' SETTING

The proposed method was implemented using the machine learning library PyTorch, in the Python language. The use of this library is motivated by the implementation of the original BigGAN architecture, the baseline model of our proposal. All the models were run using a 32 GB NVIDIA Quadro GV100 GPU and an Intel Core i7 870 with 12 threads and 16 GB of RAM. The models were trained from scratch for 1000 epochs, a number observed along the research as enough for both model's convergence and the discussion of the different divergence behaviors.



(a) Web searching

(b) Manual collection

Figure 13 – Number of images per class in the two datasets.

Source: Created by the author (2024).

Inspired by the results achieved by Zhao et al. (2020) for the BigGAN, in the present work, the transformations applied to the Differentiable Augmentation – during training – include translation (within $[-1/8, 1/8]$ of the image size, padded with zeros) and cutout (masking with a random square of half the image size). A random horizontal flip is applied – before training – to all experiments, without creating new samples (oversampling). The translation and horizontal flip transformations are adapted for the bounding boxes annotations. After a translation, a bounding box is discarded if all its coordinates exceed the image boundaries. Otherwise, the bounding box coordinates are adjusted and just the region outside of the image boundaries is discarded. On the horizontal flip, the orientation of the ears and eyes is also flipped.

Table 2 – Information about the resolutions of the web searching dataset images. The line highlighted in gray indicates the resolution used in this work.

Target resolution (px)	**Upsized images (%)**	**Average resolution of upsized images (px)**
56 × 56	0.09	50 × 50
64 × 64	0.36	57 × 57
112 × 112	15.22	87 × 87
128 × 128	20.09	95 × 95
224 × 224	37.63	130 × 130
256 × 256	41.34	140 × 140

Source: Created by the author (2024).

Table 3 lists the standard training hyperparameters, inspired by the results achieved by Tseng et al. (2021) for the BigGAN. The discriminator is optimized every $k = 4$ steps of the generator. Adam is adopted as optimizer on both the discriminator and generator, setting the learning rate to $2 \cdot 10^{-4}$, first and second-order momentum to 0 and 0.999, respectively, and weight decay to 0. The weights are initialized from a $\mathcal{N}(0, 0.02)$ distribution. During training, the weights undergo an orthogonal regularization described by Equation 3.9, with a penalty hyperparameter $\beta = 1 \cdot 10^{-4}$. To sample the input pixels for the spatial transformations carried out on the modules integrated into the BigGAN architecture, the bilinear interpolation method is used (Equation 3.3). Similarly, the bilinear interpolation is used to upsample the tensors by a factor of 2× in the generator's residual blocks.

Table 3 – Standard training hyperparameters.

Parameters	**Values**
Discriminator optimization steps (k)	4
Optimizer	Adam
First-order momentum	0
Second-order momentum	0.999
Weight decay	0
Generator learning rate	$2 \cdot 10^{-4}$
Discriminator learning rate	$2 \cdot 10^{-4}$
Weights initialization	$\mathcal{N}(0, 0.02)$
Orthogonal regularization penalty (β)	$1 \cdot 10^{-4}$

Source: Created by the author (2024).

In the LeCam-Divergence Loss, described by Equation 3.12, λ is an empirically adjustable hyperparameter applied to the regularization term $\mathrm{R_{LC}}$ (described in Equation 3.13). Two hyperparameters related to the multiscale boxes (Section 4.1) require empirical analysis. The first one is the number of concentric bounding boxes b for each interest

region. The second hyperparameter is the scale increase factor s. Finally, we decided to vary the batch size and investigate its impact on the results. We performed ablations to define some hyperparameters, which are discussed in Section 5.3.

Some metrics are used to evaluate the performance of GANs. The Fréchet Inception Distance (FID), proposed by Heusel et al. (2017), quantifies the fidelity and variability of the generated samples compared to real ones. The first step of the FID calculation consists of inputting both the fake and real samples into the Inception-v3 (SZEGEDY et al., 2016) architecture. The aim is to extract feature vectors at a specific network level. In this work, the real images – represented by the web searching dataset ($9,495$ samples) – and the fake images – $10,000$ generated samples – are input with a 112×112 resolution. The feature vectors are extracted at the output of the final AvgPool layer with 2048 elements.

In the second step, the FID is calculated by:

$$\text{FID} = \|\boldsymbol{\mu}_{real} - \boldsymbol{\mu}_{fake}\|^2 + \text{Tr}\left(K_{real} + K_{fake} - 2\sqrt{K_{real}K_{fake}}\right), \tag{5.1}$$

where $\boldsymbol{\mu}_{real}$ and $\boldsymbol{\mu}_{fake}$ are the mean vectors and K_{real} and K_{fake} are the covariance matrices for the real and fake feature vectors. Lower FID values indicate the real and fake images are similar. The whole process is computationally expensive and slow to run. For this reason, we set fixed intervals of 40 epochs to calculate the FID. For the experiments run with $1,000$ and $1,200$ epochs, 26 points (the initial step and more 25) and 31 points (the initial step and more 30) are sampled, respectively. Checkpoint models are saved for the last and lowest FID points. Other auxiliary metrics are used in this work to analyze the model's training, such as the discriminator predictions and the generator and discriminator losses for the real and fake images.

5.3 QUANTITATIVE RESULTS

In order to establish a comparison baseline model with our method, we run experiments with an architecture derived from Figures 9 and 11. This architecture disconnects all the bounding box modules, i.e., the layout encoding and the object pathways. Since the image class is derived from the bounding boxes labels, the weights vector $\boldsymbol{y}_{embedding}$ is disconnected. The batch normalization layer and the LeakyReLu activation function, after the initial fully connected layer of the generator, are also removed. As a result, we have an architecture similar to the original BigGAN (Figure 6), but for 112×112 resolution and without the Non-local blocks.

Table 4 shows the FID results and the training times for the experiments with the baseline model. Initially, we compared the BigGAN with and without the Non-local blocks. The BigGAN without the Non-local blocks achieved a lower FID (181.41 against 211.14). The Non-local block influences the capture of interactions between distant pixels on the image. The improvement with the BigGAN without the Non-local blocks can be

explained by the nature of the dataset, where the images are largely textural, i.e., the interactions between pixels are predominantly local. In addition, the training time was reduced by approximately 4 hours due to the parameters reduction.

Table 4 – FID and training time for the baseline model (BigGAN), combinations with LeCam Divergence Loss (LC) and Differentiable Augmentation (DA) and batch size variation. BigGAN* indicates that Non-local blocks are kept. Lowest FID for experiments with and without LC and DA are highlighted in gray.

Model	**Epochs**	**Batch size**	**FID (↓)**	**Training time (hours)**
BigGAN*	1000	50	211.14	∼18
BigGAN			181.41	∼14
BigGAN + $LC_{\lambda=0.01}$	1000	50	192.80	∼14
BigGAN + $LC_{\lambda=0.3}$			62.27	∼14
BigGAN + DA	1000	50	13.44	∼14
BigGAN + $LC_{\lambda=0.3}$ + DA	1200	40	33.33	∼17
		50	9.58	∼17
		60	11.57	∼17

Source: Created by the author (2024).

The next step was to apply the LeCam Divergence Loss (LC) to the BigGAN without the Non-local blocks. Tseng et al. (2021) set the regularization weight λ to 0.3 and 0.01 for the experiments on the CIFAR (full, 10% and 20% data) and the ImageNet (full, 10%, 25% and 50% data) datasets, respectively. In the present work, both values were evaluated and the best one selected. While there is a deterioration of the FID using $\lambda = 0.01$ (181.41 to 192.80), there is a significant reduction of the FID for $\lambda = 0.3$ (181.41 to 62.27). This might be related to the fact that the CIFAR dataset with 20% data includes $12,000$ samples, which is close to the $9,495$ samples in our dataset.

To analyze the impacts of the Differentiable Augmentation (DA) separately, we firstly applied it to the BigGAN without the LeCam Divergence Loss. The FID decreases considerably, from 181.41 to 13.44. Then, we applied the Differentiable Augmentation and the LeCam Divergence Loss ($\lambda = 0.3$) together, and evaluated the batch size variation to 40, 50 and 60. We observed that the model did not finish to converge after $1,000$ epochs, so we extended the training for 200 extra epochs. The combination of both methods achieved the best result for the baseline model (FID 9.58 for the batch size 50). Notice that the batch size is a sensitive variable for the baseline model, especially for batch size 40.

Table 5 shows the FID results and the training times for the experiments with our model ($BigGAN_{BMs}$), which consists of coupling the bounding box modules (BMs) to the BigGAN architecture, as described in the Chapter 4. Initially, we experimented

our model keeping the Non-local block on the architecture ($\text{BigGAN}^{*}_{\text{BMs}}$). This does not compromise the structure of our method, and the presence of the Non-local blocks should be evaluated according to the dataset under study. Our model achieved a lower FID without the Non-local blocks (15.27 against 22.15), as seen for the baseline model.

Table 5 – FID and training time for our model ($\text{BigGAN}_{\text{BMs}}$), combinations with multiscale boxes, LeCam Divergence Loss (LC), Differentiable Augmentation (DA) and batch size variation. BMs* denotes the proposed bounding box modules by Hinz, Heinrich and Wermter (2019). $\text{BigGAN}^{*}_{\text{BMs}}$ indicates that Non-local blocks are kept. Lowest FID for experiments with and without LC and DA are highlighted in gray.

Model	**Multiscale boxes**	**Epochs**	**Batch size**	**FID**	**Training time (hours)**
$\text{BigGAN}^{*}_{\text{BMs}}$	$(b = 1, s = 0)$	1000	50	22.15	∼25
$\text{BigGAN}_{\text{BMs}^{*}}$				20.48	∼25
$\text{BigGAN}_{\text{BMs}}$				15.27	∼24
$\text{BigGAN}_{\text{BMs}}$ (empty inputs)				206.13	∼23
$\text{BigGAN}_{\text{BMs}}$	$(b = 2, s = 0.1)$	1000	50	15.09	∼36
	$(b = 2, s = 0.2)$			14.67	∼36
	$(b = 2, s = 0.4)$			16.35	∼36
$\text{BigGAN}_{\text{BMs}}$	$(b = 1, s = 0.2)$	1000	50	15.32	∼32
	$(b = 3, s = 0.1)$			19.84	∼48
$\text{BigGAN}_{\text{BMs}}$ + $\text{LC}_{\lambda=0.01}$	$(b = 2, s = 0.2)$	1000	50	32.46	∼36
$\text{BigGAN}_{\text{BMs}}$ + $\text{LC}_{\lambda=0.3}$				17.58	∼36
$\text{BigGAN}_{\text{BMs}}$ + DA	$(b = 2, s = 0.2)$	1000	50	10.30	∼64
$\text{BigGAN}_{\text{BMs}}$ + $\text{LC}_{\lambda=0.3}$ + DA	$(b = 2, s = 0.2)$	1200	40	8.67	∼77
			50	8.20	∼77
			60	9.67	∼77

Source: Created by the author (2024).

We compared our model with a version ($\text{BigGAN}_{\text{BMs}^{*}}$) that maintains the specifications proposed by Hinz, Heinrich and Wermter (2019). This model disconnects all the generator's object pathways, except the Object Pathway 1/4 (blue dotted box in Figure 9), which now receives as input a spatially replicated one-hot binary vector $\boldsymbol{L}_{one\text{-}hot}$ with 7×7 resolution, in addition to an extra residual block. Analogous to the discriminator object pathway, the two residual blocks have the same number of output channels (512 and 256) as the residual blocks prior to the concatenation of the 28×28 resolution output layout to the generator. Our model ($\text{BigGAN}_{\text{BMs}}$) achieved a better FID (15.27 against 20.48) and with an equivalent training time (∼25h).

Finally, we tried out our model by setting the bounding boxes and labels inputs to zero. Its FID is comparable to the baseline model without the LeCam Divergence Loss and the Differentiable Augmentation (206.13 against 181.41), although it takes more time

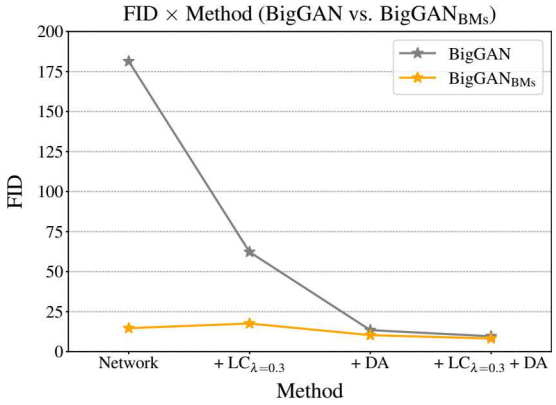
to train. This contributes to the idea that, without the bounding box annotations, it is more appropriate to use the model without the bounding box modules.

Next, we explored the number of concentric bounding boxes b and the scale factor s for the proposed multiscale boxes. First, for $b = 2$, we range s to 0.1, 0.2 and 0.4. The FID decreased for $s = 0.1$ and $s = 0.2$ compared to the model without multiscale boxes, while for $s = 0.4$ the FID increased (15.09, 14.67 and 16.35, respectively, against 15.27). With the best value for s set to 0.2, we applied this same scaling factor to the original bounding boxes ($b = 1$), and a scaling factor of 0.1 (half of 0.2) for $b = 3$. The idea is to find evidence that the improvement comes from covering a larger area with one or more extra bounding boxes. The hypothesis was supported for $b = 2$ compared to $b = 1$ (14.67 against 15.32). However, the FID increased for $b = 3$ (14.67 to 19.84). The multiscale boxes significantly impacted the training time, which is expected due to the nested loop in the bounding box modules and the scaling factor applied.

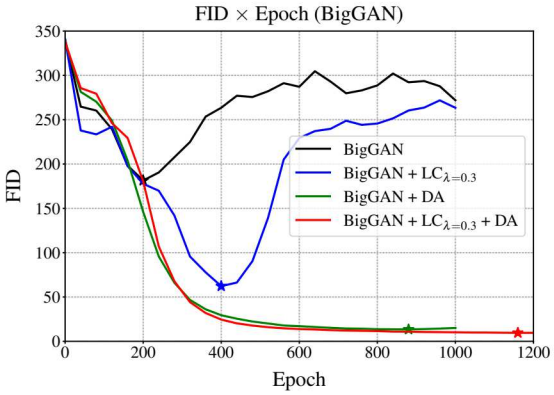
For the LeCam Divergence Loss (LC), the Differentiable Augmentation (DA) and the batch size variation experiments, we follow the same logic used for the baseline model. In the experiments exclusively with the LeCam Divergence Loss, both values for λ (0.01 and 0.3) increased the FID from 14.67 to 32.46 and 17.58, respectively. For the experiment solely with the Differentiable Augmentation, the FID decreased from 14.67 to 10.30. Both methods combined yielded the best FID (8.20) for a batch size 50. Our model appeared less sensitive to batch size variation than the baseline model. In this way, it is possible to balance the computational cost with a reduced impact on the FID score.

Figure 14a shows a comparison of the main FID results between the baseline model (BigGAN) and our model (BigGAN$_{\text{BMs}}$), highlighted in gray in Tables 4 and 5, respectively. Our model (orange line) achieved better FIDs in all experiments with and without the LeCam Divergence Loss and the Differentiable Augmentation. The baseline model (gray line) becomes competitive with our model in the experiments with the Differentiable Augmentation. According to the FID graphics over the epochs (Figures 14b and 14c), without Differentiable Augmentation (black and blue lines), our model converges to lower FID scores (14.67 and 17.58, respectively), and presents a more controlled divergence behavior. With Differentiable Augmentation (green lines) and Differentiable Augmentation plus LeCam Divergence Loss (red lines), our model converges to lower FID scores (10.30 and 8.20), but nearer to the baseline (13.44 and 9.58). In other words, we can argue that our model reveals better stability compared to the baseline, at the cost of spending considerably higher training times, reported in the "Training time (hours)" column of Tables 4 and 5.

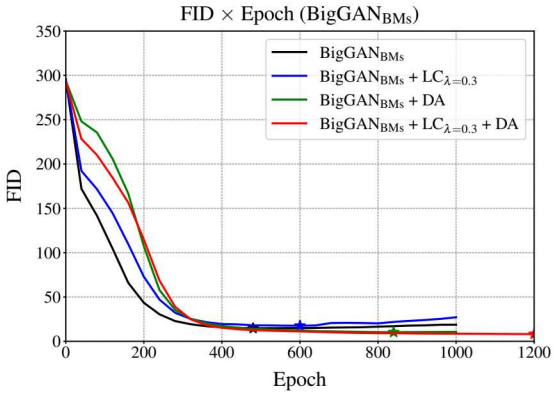
Figure 15 depicts the discriminator predictions for both the baseline (BigGAN) and our model (BigGAN$_{\text{BMs}}$). The discriminator assigns greater positive prediction values for samples that it considers to be real, and lower negative prediction values for samples



(a) BigGAN vs. BigGAN$_{BMs}$ (Ours)



(b) BigGAN



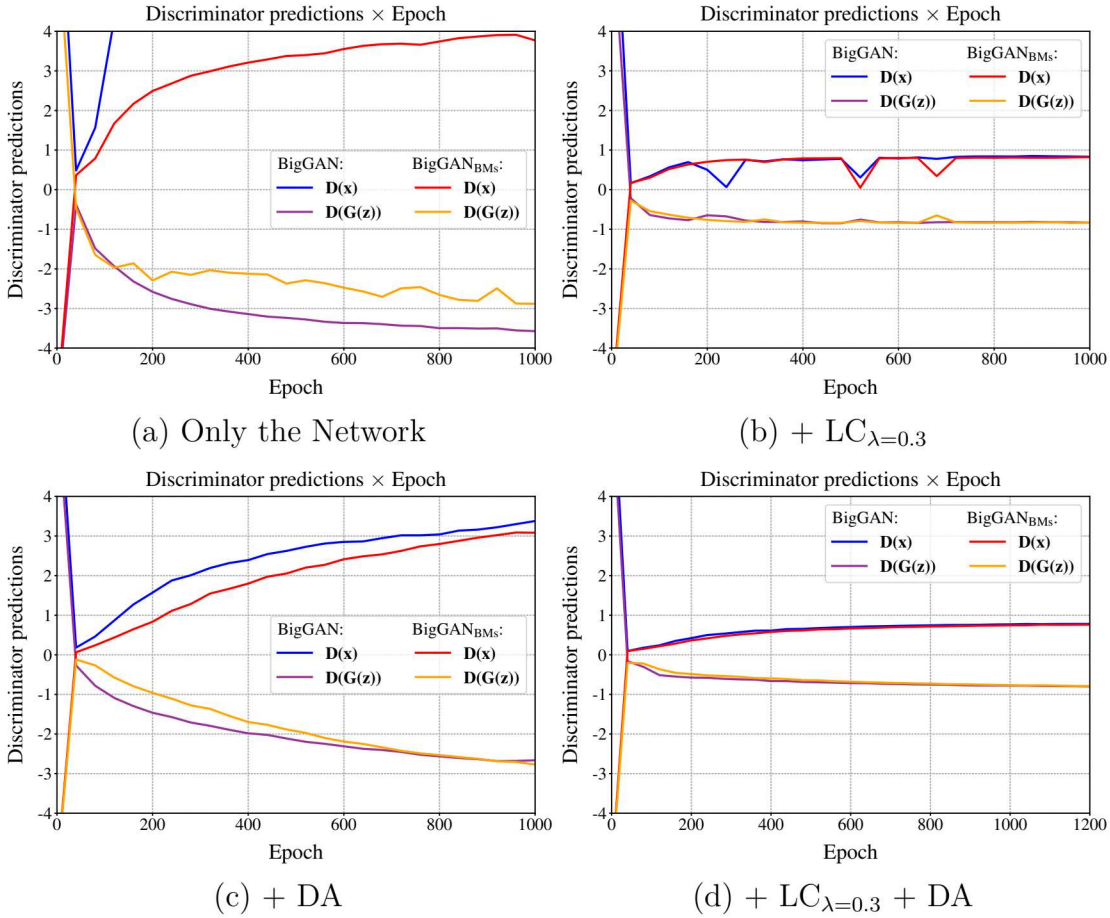
(c) BigGAN$_{BMs}$ (Ours)

Figure 14 – Comparison of the main FID results between the baseline model (BigGAN) and our model (BigGAN$_{BMs}$), highlighted in gray in Tables 4 and 5, respectively. (a) compares both models for the experiments combinations with LeCam Divergence Loss and Differentiable Augmentation. (b) and (c) present the FID curves over the epochs of the corresponding experiments plotted in (a). "$\star$" denotes the lowest FID score for each curve.

Source: Created by the author (2024).

that it deems to be fake. The $\mathbf{D}(\mathbf{x})$ curve points (blue and red lines) were acquired by the prediction's average of all the dataset samples, and the $\mathbf{D}(\mathbf{G}(\boldsymbol{z}))$ curve points (purple and orange lines) by the prediction's average of the exact $10,000$ images generated for the FID calculation, sampled at intervals of 40 epochs. The proximity of $\mathbf{D}(\mathbf{x})$ and $\mathbf{D}(\mathbf{G}(\mathbf{z}))$ is indicative of the confusion degree between the two distributions. Note that the $\mathbf{D}(\mathbf{x})$ and $\mathbf{D}(\mathbf{G}(\boldsymbol{z}))$ curves tend to lower absolute values with the LeCam Divergence Loss (graphics (b) and (d)), for both the baseline and our model. Without the LeCam Divergence Loss (graphics (a) and (c)), our model reduces the absolute values for the $\mathbf{D}(\mathbf{x})$ and $\mathbf{D}(\mathbf{G}(\boldsymbol{z}))$ predictions (red and orange curves). However, it maintains a divergent behavior over the training, as well as the baseline.

Figure 16 depicts the generator and the discriminator losses ($\mathbf{L_G}$ and $\mathbf{L_D}$ real/fake, respectively) curves for both the baseline (BigGAN) and our model (BigGAN$_{BMs}$). The $\mathbf{L_D}$ real/fake curve points are acquired by the loss's average of all the dataset samples,



(a) Only the Network

(b) + LC$_{\lambda=0.3}$

(c) + DA

(d) + LC$_{\lambda=0.3}$ + DA

Figure 15 – Discriminator predictions for both the baseline (BigGAN) and our model BigGAN$_{BMs}$. The graphics from (a) to (d) are grouped according to the LeCam Divergence Loss and the Differentiable Augmentation applications.

Source: Created by the author (2024).

and the $\mathbf{L_G}$ curve points by the loss's average of the exact 10,000 images generated for the FID calculation, sampled at intervals of 40 epochs. Without the LeCam Divergence Loss (graphics (a) and (c)), the $\mathbf{L_D}$ curves (green and orange lines) stabilize at too low value, providing insufficient feedback to the generator. Consequently, the $\mathbf{L_G}$ curves (blue lines) deteriorate during training. With the LeCam Divergence Loss (graphics (b) and (d)), the $\mathbf{L_D}$ curves stabilize at higher values, preventing the $\mathbf{L_G}$ curves from diverging. Both the baseline (continuous lines) and our model (dashed lines) behaved similarly.

5.4 QUALITATIVE RESULTS

In this section, we present a qualitative analysis of the cattle faces generated by the best model – in terms of FID – for the baseline and for our method, both with LeCam Divergence Loss plus Differentiable Augmentation. Our model's images were generated using the 939 facial layouts derived from the manual collection dataset. The same number of images were generated for the baseline. To establish a selection criterion, we initially

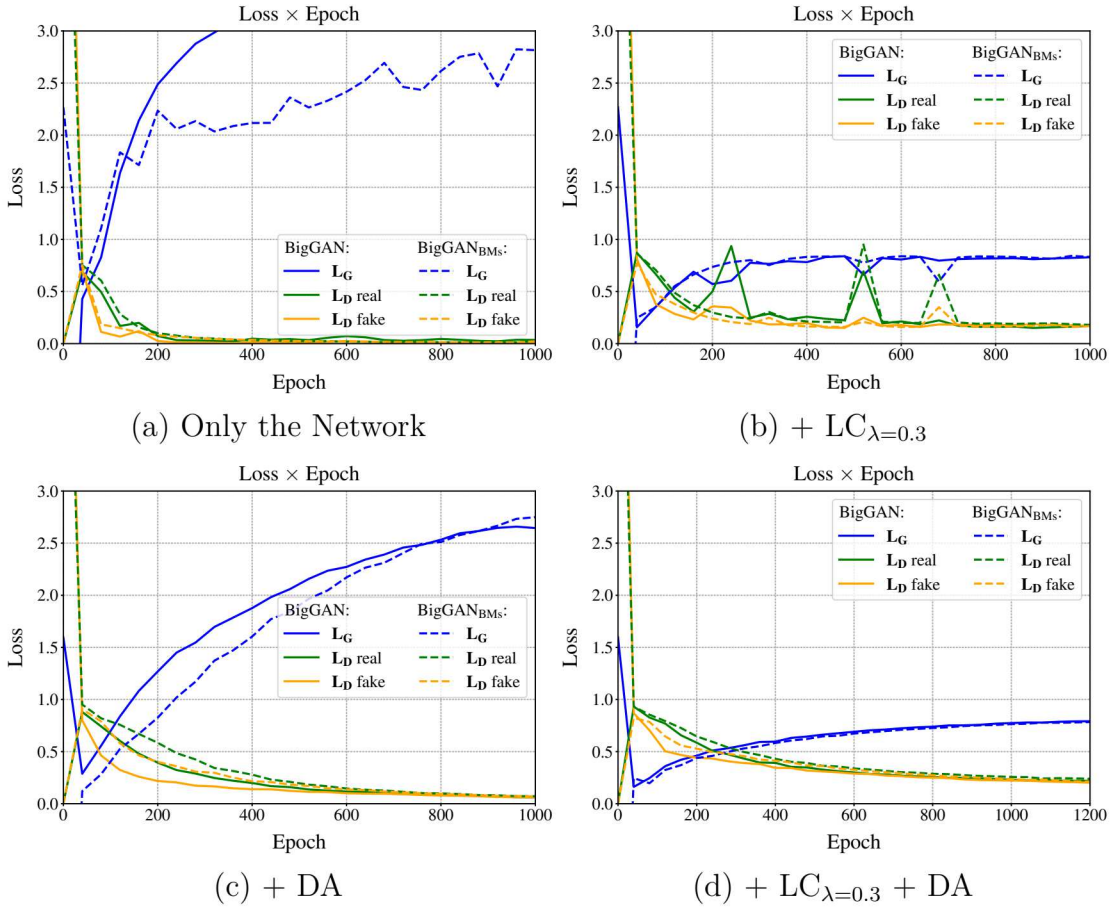


Figure 16 – Generator and discriminator losses for both the baseline (BigGAN) and our model (BigGAN$_{\text{BMs}}$). The graphics from (a) to (d) are grouped according to the LeCam Divergence Loss and the Differentiable Augmentation applications.

Source: Created by the author (2024).

analyzed the output values of the models' discriminators. Figure 17 depicts the boxplot graphic of the discriminator predictions by model. The greater the positive prediction value, the more the discriminator considers the image to be real. Conversely, the lower the negative prediction value, the more the discriminator deems the image to be fake. Our model's discriminator assigned more positive prediction values, with an upper limit, third quartile and median above zero. In contrast, the baseline discriminator had more negative predictions, with few outliers above zero. It is essential to acknowledge that the discriminators have learned to distribute their predictions in distinct manners, linked to their respective generators. Consequently, the distributions cannot be employed for a qualitative comparison between the models.

For each model we selected the images nearest to the following boxplot elements: minimum value (position at 0%), first quartile (position at 25%), median (position at 50%), third quartile (position at 75%) and maximum value (position at 100%). Near the minimum, we expect to see images with less representative features. From the first quartile to the third quartile, we expect to gradually get more representative images

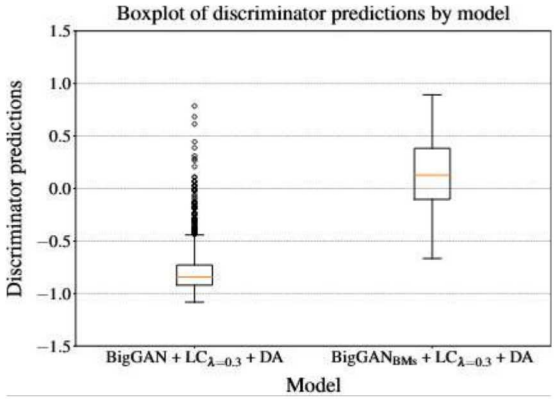


Figure 17 – Boxplot of the discriminator predictions by model.

Source: Created by the author (2024).

with different poses and backgrounds. Near the maximum, we expect to see images with predominant dataset characteristics. Figures 18 and 19 display the 36 images nearest to each boxplot element for the baseline and our model, respectively. The baseline model produces images with a greater degree of heterogeneity in background content compared to those generated by our proposed model. Because our model prioritizes the facial subregions, the background is treated in a secondary manner. Consequently, the faces generated by our model exhibit a greater sense of realism compared to the baseline.

Next, we analyzed the effects of the truncation technique, described in Section 3.2. This procedure explores the trade-off between the fidelity and the variability of the samples generated by increasing the truncation of the noise vector $\boldsymbol{z}$. To perform this analysis, we initialized the noise vector $\boldsymbol{z}$ to variance values $\sigma^2 \in \{1, 0.5, 0.05\}$, where $\sigma^2 = 1$ is the value used during training, as previously demonstrated in Figures 18 and 19. As σ^2 is reduced, and the elements of $\boldsymbol{z}$ truncated towards zero, the samples approach the mode of the generator's output distribution, at the cost of the features' variability.

Following the same protocol as for the images generated with $\sigma^2 = 1$, Figures 20 and 21 display the 12 images nearest to the 50% (median) and 100% (maximum) percentiles for the baseline and our model, respectively. The $\sigma^2 = 0.5$ achieved an interesting trade-off between fidelity and variability. Looking at the baseline, this balance was pronounced in the background and pose features, particularly for the maximum percentile. In turn, our model reflected this balance in the animal's face colors. For $\sigma^2 = 0.05$, the background of the samples generated by the baseline begins to merge with the animal's body and face. In contrast, our model preserves better the facial features. One possible explanation for this behavior stems from the fact that our model treats and reinforces the subregions independently of the background and the animal's body. The extra information given by the subregions could have provided more facial features along the training.

In addition, we evaluated the ability of our method to generate images from facial layouts not extracted directly from real samples. For each of three predominant classes

(a) Minimum

(b) First quartile

(c) Median

(d) Third quartile

(e) Maximum

Figure 18 – Images generated by the baseline (BigGAN + $LC_{\lambda=0.3}$ + DA). From (a) to (e), there are the 36 images nearest to the percentiles 0% (minimum), 25% (first quartile), 50% (median), 75% (third quartile) and 100% (maximum).

Source: Created by the author (2024).

(a) Minimum (b) First quartile

(c) Median (d) Third quartile

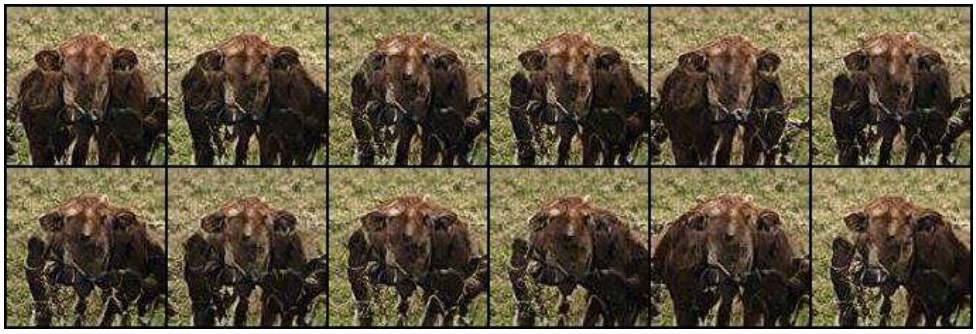
(e) Maximum

Figure 19 – Images generated by the proposed model ($\text{BigGAN}_{\text{BMs}}$ + $\text{LC}_{\lambda=0.3}$ + DA). From (a) to (e), there are the 36 images nearest to the percentiles 0% (Minimum), 25% (first quartile), 50% (median), 75% (third quartile) and 100% (maximum).

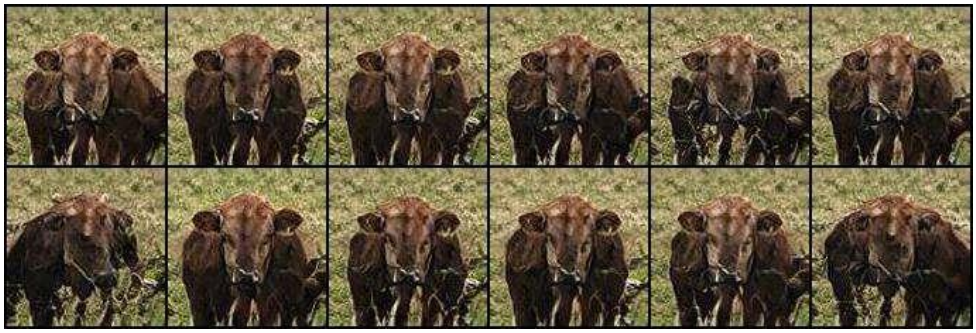
Source: Created by the author (2024).

(a) $\sigma^2 = 0.5$ and median

(b) $\sigma^2 = 0.5$ and maximum



(c) $\sigma^2 = 0.05$ and median

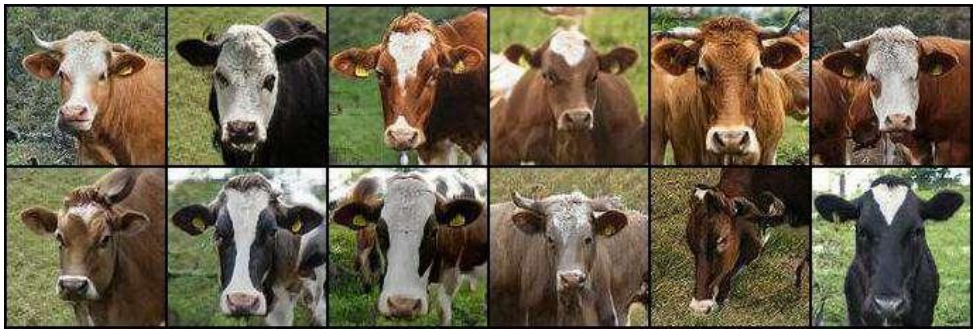


(d) $\sigma^2 = 0.05$ and maximum

Figure 20 – The effects of increasing the noise vector $\boldsymbol{z}$ truncation. Images generated by the baseline (BigGAN + $\text{LC}_{\lambda=0.3}$ + DA). (a)-(c) and (b)-(d) display the 12 images nearest to the 50% (median) and 100% (maximum) percentiles of the discriminator's predictions, respectively.

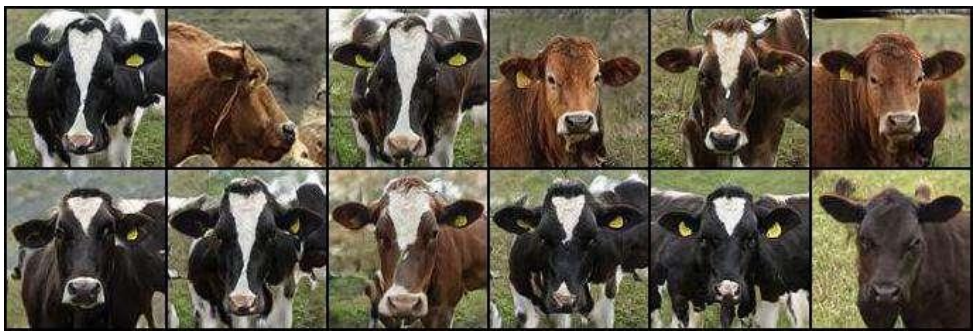
Source: Created by the author (2024).

(a) $\sigma^2 = 0.5$ and median



(b) $\sigma^2 = 0.5$ and maximum

(c) $\sigma^2 = 0.05$ and median



(d) $\sigma^2 = 0.05$ and maximum

Figure 21 – The effects of increasing the noise vector $\boldsymbol{z}$ truncation. Images generated by the proposed model ($\text{BigGAN}_{\text{BMs}}$ + $\text{LC}_{\lambda=0.3}$ + DA). (a)-(c) and (b)-(d) display the 12 images nearest to the 50% (median) and 100% (maximum) percentiles of the discriminator's predictions, respectively.

Source: Created by the author (2024).

in the manual collection dataset – 31 (frontal pose), 29 (partial lateral pose) and 26 (lateral pose) – we obtained a mean facial layout by calculating the average of all bounding boxes' coordinates in dataset for each interest subregion. Subsequently, we applied two distinct transformations. The first one translates each bounding box by a random shift factor generated by a normal distribution ($\mathcal{N}(0, 0.025)$ and $\mathcal{N}(0.025, 0.05)$), in a random direction (x or y axis). The second one scales the bounding boxes relative to the canvas' center (zoom-in and zoom-out of 25%). The objective is to subject the generator to unconventional facial layouts.

For each configuration, we generated a number of samples corresponding to the total of the class in question in the manual collection dataset (see Figure 13). The four highest predictions of the discriminator were selected and displayed in Figures 22, 23 and 24. Class 31 demonstrated a superior adaptability to shifted layouts in comparison to classes 29 and 26, which is to be expected given the prevalence of frontal faces within the dataset. Class 29 yielded samples with more unusual features. An apparent restricting factor was the left ear (left bounding box), which was poorly generated or not present at all. Class 26 showed a satisfactory adaptability, although the generator struggled to process zoom effects.

5.5 DISCUSSION

Based on the analyses presented throughout this chapter, it is possible to conclude that the main objective of this work was achieved. According to the quantitative results described in Section 5.3, our method yielded lowest FID scores in all scenarios with and without the LeCam Divergence Loss and the Differentiable Augmentation. The FID curves over the epochs show that our method converges smoothly, even without such methods to deal with data limitations. Those results provide an evidence that the bounding boxes' spatial information coupled to the original BigGAN tends to improve cattle generation. That is, the new configuration of the proposed bounding box modules is suitable for adding extra information about the cattle main subregions to the proposition of Hinz, Heinrich and Wermter (2019).

As seen in the discriminator predictions and the generator and discriminator losses in Section 5.3, the LeCam Divergence Loss contributed significantly to improving the adversarial dynamic between generator and discriminator. The fact that this behavior occurs for both the baseline and our method suggests that the data limitation is probably a major drawback. However, we cannot discard other possible hyperparameter ablations – such as learning rates and weight decay – that were fixed in this work. The extensive time required to train the models, especially for our method, made a more detailed hyperparameter investigation prohibitive.

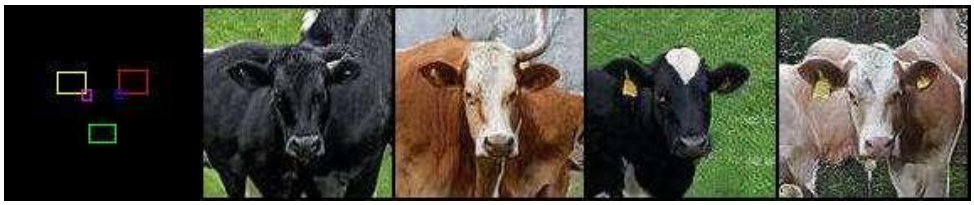
Qualitatively, the fidelity and the variability of the faces generated by our method

(a) Average facial layout

(b) Random translation with random shift factors $\mathcal{N}(0, 0.025)$

(c) Random translation with a random shift factors $\mathcal{N}(0.025, 0.05)$

(d) Scale factor of 1.25 (zoom-in of 25%)



(e) Scale factor of 0.75. (zoom-out of 25%)

Figure 22 – Images generated from shifted facial layouts relative to the average facial layout of class 31 (frontal pose). (a) are the average samples. (b) and (c) are samples shifted by random translation. (d) and (e) samples shifted by centralized zoom scale.

Source: Created by the author (2024).

(a) Average facial layout

(b) Random translation with random shift factors $\mathcal{N}(0, 0.025)$

(c) Random translation with random shift factors $\mathcal{N}(0.025, 0.05)$

(d) Scale factor of 1.25 (zoom-in of 25%)

(e) Scale factor of 0.75. (zoom-out of 25%)

Figure 23 – Images generated from shifted facial layouts relative to the average facial layout of class 31 (frontal pose). (a) are the average samples. (b) and (c) are samples shifted by random translation. (d) and (e) samples shifted by centralized zoom scale.

Source: Created by the author (2024).

(a) Average facial layout

(b) Random translation with random shift factors $\mathcal{N}(0, 0.025)$

(c) Random translation with random shift factors $\mathcal{N}(0.025, 0.05)$

(d) Scale factor of 1.25 (zoom-in of 25%)

(e) Scale factor of 0.75 (zoom-out of 25%)

Figure 24 – Images generated from shifted facial layouts relative to the average facial layout of class 31 (frontal pose). (a) are the average samples. (b) and (c) are samples shifted by random translation. (d) and (e) samples shifted by centralized zoom scale.

Source: Created by the author (2024).

are at least competitive with the baseline. The frontal pose – which contains all the interest subregions – achieved a better level of realism compared to the lateral poses, both for facial layouts extracted from the dataset itself and for shifted facial layouts unknown to the generator. This could be associated to the higher presence of frontal poses in the dataset, and the fact that they provide more subregions to the network. Our method responded more satisfactorily to the noise vector $\boldsymbol{z}$ truncation, which allows a more precise control of the trade-off between sample fidelity and variability. Overall, our method has demonstrated potential to assist phenotype based approaches, such as breed, age and pose classifications and health status inference. Analyzing the main aspects of our method, we can point out some advantages and disadvantages compared to the baseline. The advantages are:

- Greater control over the face generation, allowing to specify the localization and scale of the subregions of interest, as well as which subregions are visible;
- Lower dependence on methods to deal with data limitations, since the insertion of bounding boxes via bounding box modules provides significant geometric information for the network.

As disadvantages, we can mention:

- The necessity to provide a facial layout for the network. This requires a greater expertise on the user's part, especially without an auxiliary graphic tool;
- Longer time to train the model.

6 CONCLUSION

In this work, we seek to answer the following research hypothesis: "It is possible to generate cattle faces where the spatial location information provided by bounding boxes of interest subregions, such as the muzzle, ears and eyes, can be incorporated into a GAN, in order to quantitatively and qualitatively improve the generation of synthetic images." To this purpose, we developed a novel facial generation approach, reinforcing the spatial localization of those subregions of interest to induce the generation of the face as a whole. The proposed method is inspired by the generation of scenes with multiple objects. Based on this concept, we customized the architecture of a GAN known in the literature, named BigGAN, by inserting new bounding box modules configuration. An important restriction of this work is the use of a limited dataset. To mitigate the effects of data constraints, we applied two existing methods in the literature, the LeCam Divergence Loss and the Differentiable Augmentation. We conducted various experiments, quantitative and qualitative analyses in order to evaluate the proposed method. The experimental results indicated gains provided by the insertion of extra information for the interest subregions via bounding box modules. The additional geometric information reduced the dependence on methods to deal with limited data. On the other hand, the integration of the bounding box modules required more time to train the model. In addition, the manual insertion of subregions can allow an user to design the facial generation.

We adapted an existing annotated dataset for the image generation task. We applied a semi-automatic pre-processing to filter and adjust bounding boxes. This dataset is another contribution of this work, with information about interest subregions, such as the muzzle, ears and eyes, forming a layout of the animal's face. Once available to the computer vision community, this dataset has potential to be exploited for cattle facial images analysis and synthesis. In addition, it is possible that the dataset can also be used in other applications, such as detection and identification tasks.

As future works, we intend to explore new architectures, since the bounding box modules can be adapted to other GANs. Different generative and feature extraction approaches can be interesting, such as methods based on transformers or diffusion models. Other lines to be followed include new data annotation formats. Instead of using bounding boxes, more complex polygons can better fit into the subregions of interest. Further interest subregions can be annotated, such as horns and identification tags. Multilabel information with phenotype based annotations, such as breed, sex and age, and textual information, such as a natural language description of the image, can also contribute to refining the methodology. Since the subregions can be inserted at arbitrary positions, we intend to develop a graphical tool, allowing the user to design the facial generation.

REFERENCES

BASHIR, S. M. A. et al. A comprehensive review of deep learning-based single image super-resolution. **PeerJ Computer Science**, PeerJ Inc., v. 7, p. e621, 2021.

BERCKMANS, D. General introduction to precision livestock farming. **Animal Frontiers**, Oxford University Press, v. 7, n. 1, p. 6–11, 2017.

BROCK, A.; DONAHUE, J.; SIMONYAN, K. Large ccale gan training for high fidelity natural image synthesis. **ICLR**, 2019.

BROCK, A. et al. Neural photo editing with introspective adversarial networks. **arXiv preprint arXiv:1609.07093**, 2016.

CHOI, Y. et al. Stargan v2: Diverse image synthesis for multiple domains. In: IEEE. **CVPR**. Seattle, WA, USA, 2020. p. 8188–8197.

DIWAN, T.; ANIRUDH, G.; TEMBHURNE, J. V. Object detection using yolo: challenges, architectural successors, datasets and applications. **Multimedia Tools and Applications**, Springer, v. 82, n. 6, p. 9243–9275, 2023.

GLEERUP, K. B. et al. Pain evaluation in dairy cattle. **Applied Animal Behaviour Science**, Elsevier, v. 171, p. 25–32, 2015.

GOODFELLOW, I. et al. Generative adversarial nets. **NeurIPS**, v. 27, 2014.

HEUSEL, M. et al. Gans trained by a two time-scale update rule converge to a local nash equilibrium. **NeurIPS**, v. 30, 2017.

HINZ, T.; HEINRICH, S.; WERMTER, S. Generating multiple objects at spatially distinct locations. **arXiv preprint arXiv:1901.00686**, 2019.

HO, J.; JAIN, A.; ABBEEL, P. Denoising diffusion probabilistic models. **NeurIPS**, v. 33, p. 6840–6851, 2020.

JADERBERG, M. et al. Spatial transformer networks. **NeurIPS**, v. 28, 2015.

KARRAS, T. A style-based generator architecture for generative adversarial networks. **arXiv preprint arXiv:1812.04948**, 2019.

KARRAS, T. et al. Training generative adversarial networks with limited data. **NeurIPS**, v. 33, p. 12104–12114, 2020.

KIM, D. et al. Maximum likelihood training of implicit nonlinear diffusion models. **NeurIPS**, v. 35, p. 32270–32284, 2022.

KRISHNA, R. et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. **IJCV**, Springer, v. 123, p. 32–73, 2017.

KUMARI, N. et al. Ensembling off-the-shelf models for gan training. In: IEEE. **CVPR**. New Orleans, LA, USA, 2022. p. 10651–10662.

LIN, T.-Y. et al. Microsoft coco: Common objects in context. In: SPRINGER. **ECCV**. Zurich, Switzerland, 2014. p. 740–755.

LIU, Z. et al. Deep learning face attributes in the wild. In: IEEE. **ICCV**. Santiago, Chile, 2015. p. 3730–3738.

LIU, Z. et al. Large-scale celebfaces attributes (celeba) dataset. **Retrieved August**, v. 15, n. 2018, p. 11, 2018.

LU, Y. et al. Generative adversarial networks (gans) for image augmentation in agriculture: A systematic review. **Computers and Electronics in Agriculture**, Elsevier, v. 200, p. 107208, 2022.

MINAEE, S. et al. Image segmentation using deep learning: A survey. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, v. 44, n. 7, p. 3523–3542, 2021.

MIRZA, M.; OSINDERO, S. Conditional generative adversarial nets. **arXiv preprint arXiv:1411.1784**, 2014.

OLIVEIRA, D. A. B. et al. A review of deep learning algorithms for computer vision systems in livestock. **Livestock Science**, Elsevier, v. 253, p. 104700, 2021.

PAPAKONSTANTINOU, G. I. et al. Precision livestock farming technology: Applications and challenges of animal welfare and climate change. **Agriculture**, Multidisciplinary Digital Publishing Institute, v. 14, n. 4, p. 620, 2024.

REED, S. et al. Generative adversarial text to image synthesis. In: PMLR. **ICML**. New York, NY, USA, 2016. p. 1060–1069.

SINGH, P.; DEVI, K. J.; VARISH, N. Muzzle pattern based cattle identification using generative adversarial networks. In: SPRINGER. **Soft Computing for Problem Solving: Proceedings of SocProS 2020, Volume 1**. Singapore, 2021. p. 13–23.

STYGAR, A. H. et al. A systematic review on commercially available and validated sensor technologies for welfare assessment of dairy cattle. **Frontiers in Veterinary Science**, Frontiers, v. 8, p. 177, 2021.

SYLVAIN, T. et al. Object-centric image generation from layouts. In: AAAI. **Proceedings of the AAAI Conference on Artificial Intelligence**. Online, 2021. v. 35, n. 3, p. 2647–2655.

SZEGEDY, C. et al. Rethinking the inception architecture for computer vision. In: IEEE. **CVPR**. Las Vegas, NV, USA, 2016. p. 2818–2826.

SZELISKI, R. **Computer Vision: Algorithms and Applications**. Switzerland: Springer Nature, 2022.

TAN, M.; LE, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In: PMLR. **ICML**. Long Beach, California, USA, 2019. p. 6105–6114.

TSENG, H.-Y. et al. Regularizing generative adversarial networks under limited data. In: IEEE. **CVPR**. Nashville, TN, USA, 2021. p. 7921–7931.

VASWANI, A. Attention is all you need. **NeurIPS**, 2017.

XU, T. et al. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In: IEEE. **CVPR**. Salt Lake City, UT, USA, 2018. p. 1316–1324.

YU, F. et al. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. **arXiv preprint arXiv:1506.03365**, 2015.

ZHANG, B. et al. Styleswin: Transformer-based gan for high-resolution image generation. In: IEEE. **CVPR**. New Orleans, LA, USA, 2022. p. 11304–11314.

ZHANG, H. et al. Self-attention generative adversarial networks. In: PMLR. **ICML**. Long Beach, California, USA, 2019. p. 7354–7363.

ZHANG, H. et al. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In: IEEE. **ICCV**. Venice, Italy, 2017. p. 5907–5915.

ZHANG, H. et al. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, v. 41, n. 8, p. 1947–1962, 2018.

ZHAO, S. et al. Differentiable augmentation for data-efficient gan training. **NeurIPS**, v. 33, p. 7559–7570, 2020.