



UNIVERSITÀ DI PARMA

ARCHIVIO DELLA RICERCA

University of Parma Research Repository

MetalGAN: Multi-domain label-less image synthesis using cGANs and meta-learning

This is the peer reviewed version of the following article:

Original

MetalGAN: Multi-domain label-less image synthesis using cGANs and meta-learning / Fontanini, T.; Iotti, E.; Donati, L.; Prati, A.. - In: NEURAL NETWORKS. - ISSN 0893-6080. - 131:(2020), pp. 185-200. [10.1016/j.neunet.2020.07.031]

Availability:

This version is available at: 11381/2885017 since: 2022-01-18T17:00:25Z

Publisher:

Elsevier Ltd

Published

DOI:10.1016/j.neunet.2020.07.031

Terms of use:

Anyone can freely access the full text of works made available as "Open Access". Works made available

Publisher copyright

note finali coverpage

(Article begins on next page)

26 December 2024

MetalGAN: Multi-Domain Label-Less Image Synthesis Using cGANs and Meta-Learning

Tomaso Fontanini, Eleonora Iotti, Luca Donati, Andrea Prati

*IMP Lab, Department of Engineering and Architecture, University of Parma, Parco Area
delle Scienze, 181/A, 43124 Parma*

Abstract

Image synthesis is currently one of the most addressed image processing topic in computer vision and deep learning fields of study. Researchers have tackled this problem focusing their efforts on its several challenging problems, e.g. image quality and size, domain and pose changing, architecture of the networks, and so on. Above all, producing images belonging to different domains by using a single architecture is a very relevant goal for image generation. In fact, a single multi-domain network would allow greater flexibility and robustness in the image synthesis task than other approaches. This paper proposes a novel architecture and a training algorithm, which are able to produce multi-domain outputs using a single network. A small portion of a dataset is intentionally used, and there are no hard-coded labels (or classes). This is achieved by combining a conditional Generative Adversarial Network (cGAN) for image generation and a Meta-Learning algorithm for domain switch, and we called our approach MetalGAN. The approach has proved to be appropriate for solving the multi-domain label-less problem and it is validated on facial attribute transfer, using CelebA dataset.

Keywords:

Generative Adversarial Networks, Meta-Learning, Image-to-Image translation, Few-shots, Multi-Domain

Email address: `tomaso.fontanini@studenti.unipr.it` (Tomaso Fontanini)

10 1. Introduction

11 Image generation or synthesis consists in the act of producing a novel
12 image—representing a subject of interest or whatever else—from an input
13 that could be a random noise matrix, another (real) image, or a combination
14 of these two possibilities, eventually put beside a label or a condition that
15 somehow controls the output. The required output should belong to a specific
16 domain, or it should have been obtained following a precise style. Conversely,
17 in some cases the image domain or style could be not decided *a-priori*, and
18 the developed system should perform multi-domain image generation.

19 Since the recent advances of deep learning techniques and architectures
20 on image generation, the image synthesis task has become more and more ac-
21 cessible and understood. Examples of such techniques are the use of *Genera-*
22 *tive Adversarial Networks (GANs)* (in particular *Deep Convolutional* GANs,
23 called DCGANs) and *conditional GANs (cGANs)*, which should have dif-
24 ferent architectures, such as *U-Nets* and *StyleGAN*, and different training
25 methods, like *pix2pix*, *cycleGAN*, and so on. A high number of specific
26 approaches were developed to face the aforementioned variety of image gen-
27 eration tasks, leading to a vast literature for each specific sub-problem. A
28 brief outline of the major methodologies are reported in the next section.

29 This paper focuses on the specific problem of image-to-image translation.
30 Image-to-image translation is the act of transforming an arbitrary image in
31 another, more useful, representation of the same data. Image colorization,
32 semantic segmentation, style transfer are examples of image-to-image transla-
33 tions. In particular, this work approaches the task of transforming the input
34 image over a range of so-called *domains*, i.e. recognizable sets of similar im-
35 ages which share common characteristics. One of the scope of this work is to
36 develop a single architecture that is able to handle many different domains,
37 i.e. a multi-domain image-to-image generator system. More specifically, in
38 a system like that, the architecture is required to learn multiple mapping
39 functions, that is one for each domain. In our case, the multiple domains
40 are represented by different facial attributes like “blond hair” or “pale skin”
41 and the mapping functions have the objective to apply these attributes on
42 any face passing through the architecture. Moreover, the idea is to make the
43 model capable to learn new, unseen, domains by using few images for each
44 new domain, in order to gain a great flexibility and generalization capability
45 of the proposed architecture and to tackle also those applications or domains
46 where there is scarcity of available data. Hence, the topic covered in this

47 work is *multi-domain image-to-image translation*, deeply entranced with the
48 concept of domain adaptation.

49 One interesting take on this topic is that many of these image-to-image
50 transformations are linked by a common way of working. For example, chang-
51 ing hair color, e.g. switching the domain to the new one of “people with blond
52 hair”, needs to correctly segment hair in the same way of changing the do-
53 main in “people with black hair”. Similarly, changing a face into its older
54 version and add glasses to a face both need to correctly locate the subject’s
55 eyes. Yet, for long time, a single neural network for each of these tasks had
56 to be created, even if the tasks were quite similar. A solution to this kind
57 of problem has been proposed with StarGAN (Choi et al., 2018), which had
58 the intuition of bringing together multiple image-to-image transformations
59 in the same network architecture.

60 Another observation is that most of the existing approaches to image-
61 to-image translations perform a full training with input-output examples of
62 images, to achieve high quality results. An evident drawback of this approach
63 is that they need very large datasets to be trained. Dataset could be labeled
64 or unlabeled, but usually the domain switch is controlled by a conditioning
65 label, which indicates the target domain to transform the image to. Hence,
66 image-to-image translation often requires a lot of labeled images, where the
67 label denotes the domain(s). It is worth noting that an image could have
68 more than one label: this is the reason for not addressing labels as classes.

69 Regarding the adaptation of the model to new domains with few images,
70 a similar issue is the few-shot learning problem. Few-shot problems are often
71 addressed by using *meta-learning* techniques, thanks to their ability to switch
72 among a distribution of tasks during training. Training in a meta-learning
73 settings means creating a learning system that includes another learning sub-
74 system: the sub-system trains a model (such as a neural network) on a single
75 task sampled from the distribution, and the meta-learning system trains the
76 sub-system, thus adapting the model to all tasks. Meta-learning methods
77 have proved to be successful in classification and regression scenarios, but
78 there are still few papers (Liu et al., 2019; Zhang et al., 2018) in the field of
79 image generation.

80 Linked to domain adaptation, another known problem of traditional train-
81 ing settings is that once a new set of tasks emerges, e.g. a new domain is
82 added to the target (or desired) outputs, a full retraining of the whole sys-
83 tem is needed. This happens even if the new task is similar to tasks that
84 the network has already learned. The full re-training includes incorporating

85 the new domain in the input examples and also it often needs architecture
86 changes.

87 The main proposal of this article, and its principal contributions are:

- 88 • a system that consists in a single cGAN (i.e., two networks, a generator
89 and a discriminator) performing image-to-image translation, trained on
90 multiple domains;
- 91 • both networks do not contain any reference to the label or domain of
92 the input or output (*label-less*) therefore allowing a much more flexible
93 architecture;
- 94 • the system is able to switch task with just few examples of a new, un-
95 seen domain, by means of a meta-learning training algorithm. This was
96 impossible in previous architectures, representing a great limitation;
- 97 • the system uses knowledge accumulated at training time with well-
98 known, largely represented classes, to easily learn new, unknown tasks
99 in few iterations.

100 Taking into account all these contributions and the main proposed idea to
101 fuse together meta-learning and GAN, we named our approach *MetalGAN*.

102 The paper is organized as follows. Section 2 presents an extensive eval-
103 uation of the state-of-art. Section 3 introduces a complete overview of the
104 system: main idea and notations, architecture of the network and algorithm.
105 Section 4 describes the experimental results. Finally, Section 5 presents our
106 conclusions.

107 2. Related Work

108 **Image-to-image translation.** The main topic of this paper, i.e. image-
109 to-image translation, has become a hot topic in machine learning researcher
110 community after the introduction of encoder-decoder networks like U-Nets
111 (Ronneberger et al., 2015), *Fully Convolutional Neural networks (FCN)* (Long
112 et al., 2015) and conditional GANs (Mirza and Osindero, 2014). The GAN
113 approach, that can be considered a form of Artificial Curiosity as inter-
114 estingly stated in (Schmidhuber, 2020), to image synthesis has proven an
115 unprecedented quality of output results, reaching photorealism in many do-
116 mains, such as face synthesis. While traditional GANs (Goodfellow et al.,
117 2014) generate images from noise, conditional GANs (cGANs) (Mirza and

118 Osindero, 2014) in their many variations are able to generate images from
119 labels or other input images, or both. To this extent, cGANs are often used
120 to perform lots of different image-to-image translation tasks like producing
121 sketch colorization and texture generation (Sangkloy et al., 2017; Xian et al.,
122 2018), super-resolution of images (Ledig et al., 2017) or to generate a photo-
123 realistic image from a semantic label map (Wang et al., 2018; Park et al.,
124 2019). cGANs can be trained in both a paired (Isola et al., 2017; Zhu et al.,
125 2017b) or unpaired way (Zhu et al., 2017a; Almahairi et al., 2018; Kim et al.,
126 2017a).

127 In our approach, we use cGANs without a paired dataset with only input
128 image but label-less, in order to maintain a great generalization capability
129 of the generator network. Moreover, we introduced skip connections in the
130 generator network, as in U-Nets.

131 **Multi-domain image-to-image translation.** A common trait of most of
132 the image-to-image methods is that they are only able to produce outputs
133 belonging to a single domain or class. Regarding multi-domain facial at-
134 tributes transfer, our main work of reference is StarGAN (Choi et al., 2018),
135 though there exist other relevant works like (He et al., 2019; Xiao et al., 2017;
136 Kim et al., 2017b). StarGAN proposes an unified method for multi-domain
137 image-to-image translation. It achieves great results in image synthesis tak-
138 ing strength from the multiple domain adaptations and it learns multiple
139 domains at the same time using only one underlying representation. The
140 main differences between StarGAN and the proposed method are: in our
141 approach, networks do not use labels information (while StarGAN do); our
142 training method relies on a small number of images per-iteration; and also a
143 few-shot-like approach is employed when dealing with new domains during
144 inference.

145 **Few-shots learning.** Few-shot problems are usually tackled with meta-
146 learning techniques, since recent results show great performance of meta-
147 learners on typical few-shot datasets and learning settings. There are many
148 types of meta-learners. Some learn how to parameterize the optimizer of
149 the network (Hochreiter et al., 2001; Ravi and Larochelle, 2016), while oth-
150 ers use a network as optimizer (Li and Malik, 2017; Andrychowicz et al.,
151 2016; Wichrowska et al., 2017). Furthermore, using a recurrent neural net-
152 work trained on the episodes of a set of task is one of the most general
153 approach (Santoro et al., 2016; Mishra et al., 2017; Duan et al., 2016; Wang

154 et al.). For our work, the most relevant meta-learners are the ones based on
155 hyper-parameterized gradient descent such as Reptile (Nichol et al., 2018)
156 and MAML (Finn et al., 2017). In fact, we use the Reptile algorithm applied
157 to a generation problem, where Reptile tasks are identified with our domains.
158 Reptile was already used in combination with GANs in (Clouâtre and De-
159 mers, 2019) in order to generate very simple black and white images (such
160 as MNIST digits) or in (Zhang et al., 2018) that introduced an adversarial
161 discriminator, conditioned on tasks.

162 Regarding few-shot image-to-image translation, a new method was re-
163 cently introduced in (Liu et al., 2019), coupling an adversarial training scheme
164 with a novel network design. Unlike our method, it does not use meta-
165 learning and does not act as a proper domain transfer algorithm, but rather
166 as a style transfer one: for example, in the case of face image translation
167 task, the translation output maintains the pose of the input content image,
168 but the appearance is similar to the the faces of the target person.

169 **3. Overview of the System**

170 *3.1. Idea and Notations*

171 As briefly outlined in the introduction, there are some key points from
172 which our work originates, namely, the need of a *few-shots* setting, the use
173 of a *single* GAN architecture, the *absence of labels*, and the *multi-domain*
174 adaptation. All these key points require a proper definition.

175 Starting from the most potentially ambiguous definition, we call a “do-
176 main” a set of images which share a well-defined common characteristic,
177 clearly recognizable by using a single label or keyword: for example, “black-
178 hair” in a dataset of faces denotes the domain of people with a black hair
179 color. Given the example above, it is also clear that the type of dataset is also
180 important: if the dataset contains both dogs and cats images, “black-hair”
181 should have another meaning; if it contains only landscapes, “black-hair”
182 should have no meaning at all. Moreover, domains are not mutually exclu-
183 sive, rather they could intersect each other.

184 Closely related to the concept of domain, there is the concept of “label”.
185 Usually, when approaching multi-domains problems, labels are employed to
186 identify which domains a certain image belongs to. This helps the networks
187 in detecting a target domain and thus generating images belonging to such a
188 domain. In our case, *label-less* means that the domain of the input and the
189 target images have to be inferred from other information.

190 In a classic few-shots classification setting, from which we borrow the
 191 notations, there are n classes $\{c_1, c_2, \dots, c_n\}$ and a certain number k of input
 192 examples per-class, e.g. $\{x_1, x_2, \dots, x_{k_i}\}$ are the input of the i -th class c_i .
 193 During training only N classes per iteration are used over the total number
 194 of classes n , and for each of these N classes only K input examples over
 195 the total number of examples of a class are used, where $K \ll k$. Then,
 196 the trained N -classifier has to classify a new example of a random class \tilde{c} .
 197 In our case, domains are treated as classes, and the generator-discriminator
 198 (from now on, called G and D) networks are trained on a single domain per
 199 meta-iteration ($N = 1$), in order to make G and D able to learn the domain
 200 they are working on, without labels. The number K of examples per domain
 201 varies according to the type of experiment performed (see Section 4), but we
 202 choose to perform an almost full training and a few-shot inference to allow G
 203 to learn adequately the reconstruction of images, and then to switch domain.
 204 The architecture of our multi-domain GAN is detailed in Section 3.2.

205 Finally, in the meta-learning nomenclature, we defined a *task* as a group
 206 of K images that belong to the same domain, used for the inner-iteration of
 207 the algorithm, explained in detail in Section 3.3.

208 Our approach uses a single GAN on different tasks. This forces the un-
 209 derlying weights structure of both G and D networks to learn a general yet
 210 effective representation for describing all tasks. G and D networks are con-
 211 ditioned with the use of a meta-learning algorithm, on each task/domain.
 212 Other approaches, like StarGAN, instead, needs target labels that condition
 213 the output for both G and D networks. In detail, the conditioning is implic-
 214 itly provided by the task selection performed during meta-learning. For each
 215 meta-iteration, a single task is selected, and the network is trained on that
 216 single task for a number of internal iterations. In the next meta-iteration
 217 the training is performed on another, different but related task. With this
 218 training algorithm the network learns, meta-iteration after meta-iteration, a
 219 representation that is good (but not optimal) in performing all tasks, and
 220 just needs a little final push (few epochs of training) to be moved in the
 221 direction of the target task.

222 3.2. Architecture of the Network

223 One of the strengths of our proposal is that it completely removes the
 224 need of providing specific labels for the data, because the network does not
 225 use one-hot labels or similar. If data are already labeled, labels are only
 226 useful in the preprocessing phase for dividing into domains the dataset, since

227 the main algorithm works task-by-task. It is worth noting that such domains
 228 could overlaps. On the other hand, unlabeled data has to be clustered into
 229 domains, a passage that can be completely automated (in contrast with man-
 230 ual labeling), but using a clustering method, domains does not overlap. A
 231 clusterization followed by a meta-learning approach is shown in a preliminary
 232 work on colorization (Fontanini et al., 2019).

233 Our system is composed by a single cGAN. In particular, since the objec-
 234 tive is to generate the face of a person with only a bunch of new attributes,
 235 without changing the peculiar traits of the person itself and without using
 236 labels, we conditioned the cGAN with the input face, in order to maintain the
 237 identity of the person, and, at the same time, changing the target attributes.

238 The generator network G is the same as the StarGAN one with the addi-
 239 tion of skip-connections (inspired by the classic U-Net), but input labels are
 240 removed. The introduction of skip-connection in the generator architecture
 241 aims at enhancing the quality of the reconstruction; in other words, they are
 242 useful for keeping contents of the input image unchanged in the output image
 243 (for example, the face of a person remains the same, despite the changing of
 244 hair color).

245 On the other side, the D structure is the PatchGAN from pix2pix (Isola
 246 et al., 2017). Since during each task the network tunes itself on a single do-
 247 main, there is no need of a domain classification output for our discriminator.
 248 Instead, we choose to classify images both as real or fake and as belonging or
 249 not to the current domain. By doing so our discriminator has two outputs:
 250 $D_{adv}(x)$ and $D_{dom}(x)$, one for each probability distribution.

251 Finally, we define a set of losses in order to train our architecture.

252 **Adversarial Loss.** For the discriminator, we use an adversarial loss to dis-
 253 tinguish between real and generated images:

$$\mathcal{L}_{adv}(D, G) = \mathbb{E}_{y \sim p_{\tau}} [\log D_{adv}(y)] + \mathbb{E}_{x \sim p_{data}} [1 - \log D_{adv}(G(x))], \quad (1)$$

254 where y is sampled over a distribution of current task images p_{τ} (real sam-
 255 ples), and x over the distribution of the whole dataset p_{data} ($G(x)$ are the
 256 generated samples).

257 In particular, during each task, D tries to classify if an image (or a batch
 258 of images) y belongs or not to the current domain distribution τ (all images
 259 in the batch must belong to the domain). For example, if the current task is

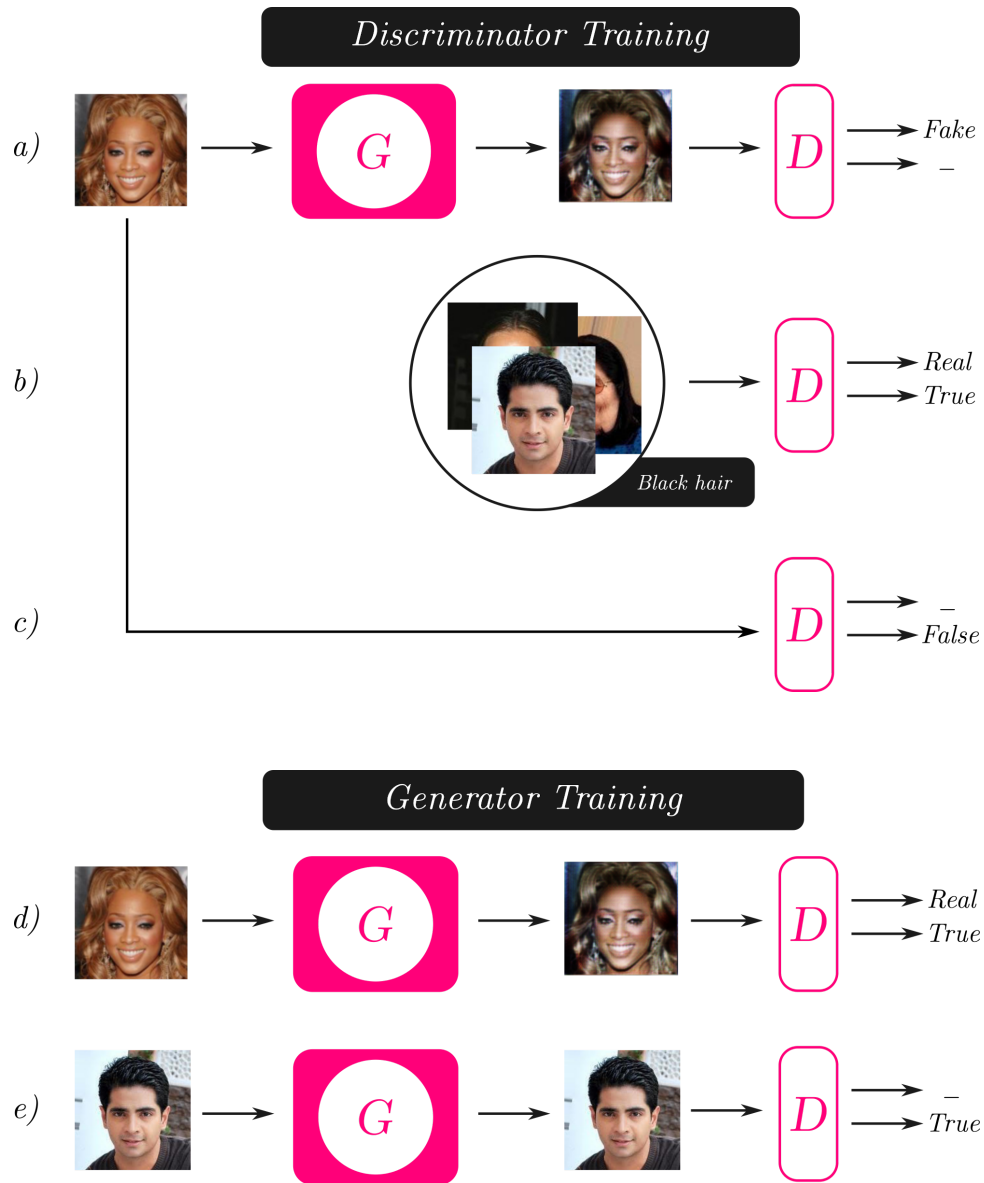


Figure 1: Complete network architecture. First, the discriminator is trained to distinguish between fake images (a) and real ones (b) and between images belonging to the current domain (b) and images that do not belong to it (c). Then, the generator is trained to fool the discriminator by labeling its outputs as real and as belonging to the current domain (d). Finally, the reconstruction step is executed and its results are labeled as part of the current domain (e).

260 to produce people with blond hair, the discriminator has to determine if an
 261 image contains a person with blond hair or not, while in classic adversarial
 262 settings it should simply decide if the image contains a face or not. The
 263 adversarial loss formula (1) does not reflect explicitly this aspect, since the
 264 only difference is the nature of the input y : in our work, y is not concatenated
 265 to any label.

266 **Domain Loss.** After we select a new task, during the training of the dis-
 267 criminator, we want images sampled from the current task to be classified as
 268 such and, on the other side, images sampled from the whole dataset to be
 269 classified as not belonging to the current task.

$$\mathcal{L}'_{\text{dom}}(D) = 2 \cdot \mathbb{E}_{y \sim p_\tau} [\log D_{\text{dom}}(y)] + \mathbb{E}_{x \sim p_{\text{data}}} [1 - \log D_{\text{dom}}(x)], \quad (2)$$

270 where the multiplicative factor before $\mathbb{E}_{y \sim p_\tau} [\log D_{\text{dom}}(y)]$ is motivated by the
 271 fact that we need to take into account that an image x may also belong to the
 272 domain identified by the current task, since it is drawn from the whole data
 273 distribution. For this reason, the first part of the equation strongly reinforces
 274 the classification of examples of the target domain, while the second part
 275 weakly penalizes every domain (that is, also the target one).

276 Instead, during the generator training, the goal is that all the generated
 277 images, even the ones obtained from the reconstruction of the input, would
 278 be classified as belonging to the current task.

$$\mathcal{L}''_{\text{dom}}(D, G) = \mathbb{E}_{x \sim p_{\text{data}}} [\log D_{\text{dom}}(G(x))] + \mathbb{E}_{y \sim p_\tau} [\log D_{\text{dom}}(G(y))] \quad (3)$$

279 Adversarial and domain losses are visually described in Figure 1.

280 **Reconstruction Loss.** This loss is crucial to guarantee that the generator
 281 maintains the content information of the source image. G has to be already
 282 tuned on the current domain/task in the meta-learning training. Since we
 283 completely removed the labels from our architecture and we tune the network
 284 on a new domain at each iteration, we cannot use a cycle consistency loss like
 285 in StarGAN, because G is able to produce images of only one target domain
 286 each task. The solution we choose to adopt is to apply the reconstruction
 287 loss on the images y belonging to the target domain. The reason is that if

288 an image already belongs to the target domain, it should be left unchanged
 289 by G . The equation for the reconstruction loss is as follows:

$$\mathcal{L}_{\text{rec}}(G) = \mathbb{E}_{y \sim p_\tau} [\|G(y) - y\|_1], \quad (4)$$

290 where $\|\cdot\|_1$ denotes the L_1 norm in the space of target images.

291 **Feature Matching Loss.** In order to regularize the training, we also include
 292 a feature matching loss following the work of (Wang et al., 2018) and (Liu
 293 et al., 2019). Feature Loss stabilizes the training since it is required to the
 294 Generator to produce natural statistic at multiple scale. We extract features
 295 from the discriminator layers located before the prediction layer. This feature
 296 extractor is called D_{feat} . The definition of the feature matching loss is as
 297 follows:

$$\mathcal{L}_{\text{feat}}(D_{\text{feat}}, G) = \mathbb{E}_{x, y \sim p_{\text{data}}, p_\tau} [\|D_{\text{feat}}(G(x)) - D_{\text{feat}}(y)\|_1]. \quad (5)$$

298 **Full Objective.** Finally, our full objective becomes:

$$\mathcal{L}_D = \mathcal{L}_{\text{adv}} + \mathcal{L}'_{\text{dom}}, \quad (6)$$

$$\mathcal{L}_G = w_{\text{adv}} \mathcal{L}_{\text{adv}} + w_{\text{dom}} \mathcal{L}''_{\text{dom}} + w_{\text{rec}} \mathcal{L}_{\text{rec}} + w_{\text{feat}} \mathcal{L}_{\text{feat}}, \quad (7)$$

299 for the discriminator and generator, respectively. Furthermore, w_{adv} , w_{dom} ,
 300 w_{rec} , w_{feat} are the weights assigned to the loss functions. The discriminator
 301 loss functions do not have weights assigned since adversarial and domain
 302 losses should contribute equally to discriminator training to obtain balanced
 303 results. Weights choices for the generator loss are more properly discussed
 304 in Section 4.

305 3.3. Algorithm

306 Our approach relies on a meta-learning algorithm based on Reptile (Nichol
 307 et al., 2018) and adapted to the image generation problem.

308 The problem setting is as follows. A large dataset of images, called \mathcal{D} ,
 309 is used to extract random input images. Let τ_j be a single task, where j
 310 ranges over the number of chosen training domains, here called N_τ . Each
 311 task dataset consists of a restriction of \mathcal{D} on the images of a single domain,
 312 called $\mathcal{D}|_{\tau_j}$. Hyper-parameters of the algorithm are the inner learning rates of
 313 G and D networks, respectively λ_G and λ_D ; the loss weights w_{adv} , w_{dom} , w_{rec} ,

314 and w_{feat} ; two thresholds t and T for keeping discriminator accuracy into a
 315 certain range (in order to neither over- nor under-train D); and a learning
 316 rate for the outer networks, i.e. a meta-learning rate λ_{ML} . Parameters of
 317 the networks, i.e. network weights and biases, are indicated as θ_G and θ_D
 318 for G and D , respectively. The algorithm is divided into two phases, as
 319 illustrated in Figure 2. The first one is the training phase, and the second
 320 one is the inference phase. In the training phase, the G - D network is trained
 321 repeatedly on a single task, randomly extracted at each epoch from the set
 322 of available tasks. During inference phase, instead, a new task τ_I is used for
 323 a last-time few-shot training to adapt the network to the new domain. A
 324 detailed explanation of training phase is given in the next paragraph and in
 325 Figure 3, and it is followed by another paragraph devoted to the description
 326 of the inference phase.

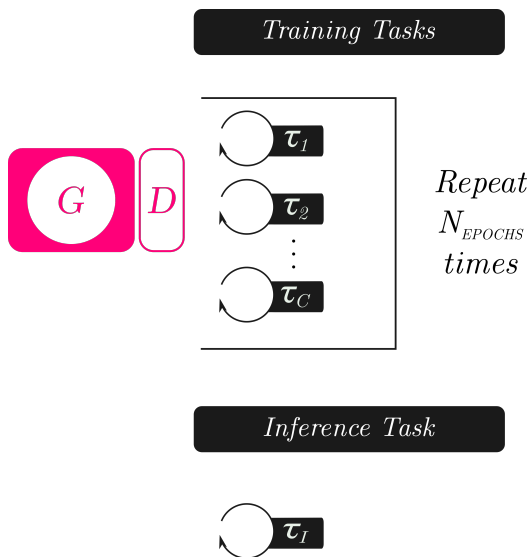


Figure 2: A full overview of the system: during the training phase, the network is trained for N epochs on a set of tasks, and then, during the inference phase, a new unknown task, i.e. not present in the training phase, is selected and added to the network.

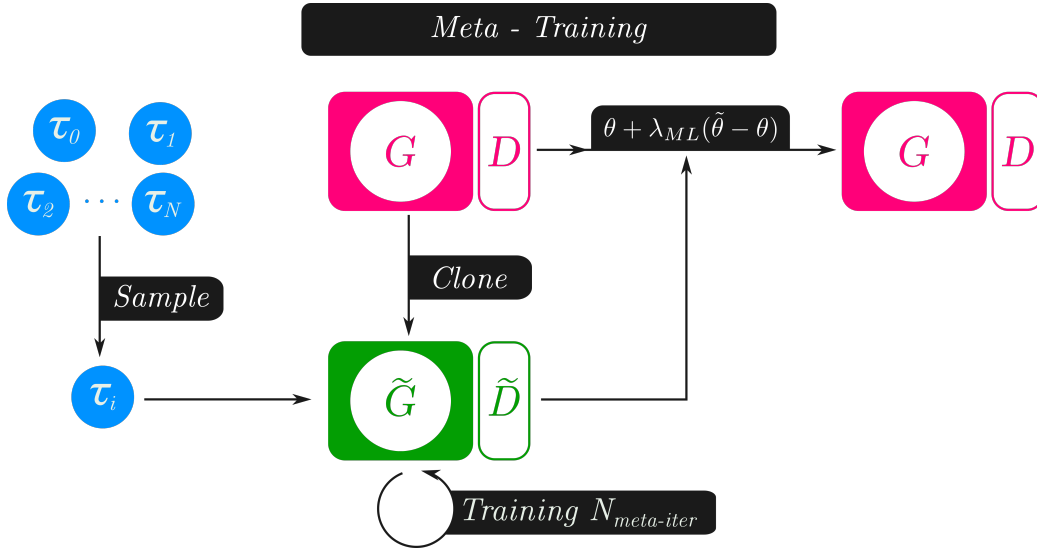


Figure 3: Scheme of a single training epoch. A task τ_i is sampled and $N_{meta-iter}$ inner iterations are performed on cloned networks. Then θ and $\tilde{\theta}$ are used to update the networks parameters using the Reptile equation.

Algorithm 1 MetalGAN algorithm

Require: N_{epochs} : number of epochs

Require: N_τ : number of selected domains

Require: λ_{ML} : meta-learning rate

- 1: load entire dataset : \mathcal{D}
 - 2: load datasets restricted to each single task τ_j : $\mathcal{D}|\tau_j$ for $j \in \{0, \dots, N_\tau\}$
 - 3: **for** $epoch \in \{0, \dots, N_{epochs}\}$ **do**
 - 4: extract randomly τ_j
 - 5: clone D into \tilde{D} of parameters $\theta_{\tilde{D}}$
 - 6: clone G into \tilde{G} of parameters $\theta_{\tilde{G}}$
 - 7: **Inner training loop** on τ_j
 - 8: $\theta_G \leftarrow \theta_G + \lambda_{ML}(\theta_{\tilde{G}} - \theta_G)$ ▷ updates generator parameters
 - 9: $\theta_D \leftarrow \theta_D + \lambda_{ML}(\theta_{\tilde{D}} - \theta_D)$ ▷ updates discriminator parameters
 - 10: **end for**
-

327 **Training.** The algorithm for training consists of an outer and an inner loop,
 328 similar to Reptile. The outer loop is responsible of training the actual G - D

329 networks, updating their parameters, epoch-by-epoch as a traditional learn-
 330 ing algorithm. At each epoch, a task τ is randomly sampled from a distri-
 331 bution of tasks. It is recalled that, in our case, a task is a domain and the
 332 associated dataset is the few-shot subset of the set of images in the domain.
 333 Then, G and D networks are cloned into \tilde{G} and \tilde{D} networks of parameters
 334 $\theta_{\tilde{G}}$ and $\theta_{\tilde{D}}$, respectively.

335 The cloned networks are trained in the inner training loop, where the
 336 traditional DCGAN training is performed by using task images, here indi-
 337 cated as y . This is needed in order to learn the current domain. Also a small
 338 portion of generic images from \mathcal{D} is used, to teach the generator to perform
 339 domain switch. These images are called x . It is required to the generator
 340 to learn the transformation from the random image x of the dataset into an
 341 output image “similar” to y , i.e., the generated image should belong to the
 342 extracted task.

343 Finally, the obtained parameters are used to update G and D weights,
 344 with the Reptile rule (a sort of SGD step where the gradient is approxi-
 345 mated by the difference between inner and outer weights). This baseline is
 346 illustrated in Figure 3.

347 In detail, the outer training loop is shown in Algorithm 1. Lines 8–9 of
 348 Algorithm 1 are responsible of the parameter adaptation for networks G and
 349 D and such an operation is performed layer by layer.

350 The inner training loop is illustrated in Algorithm 2. It is nothing more
 351 than a classic DCGAN training, but performed on the cloned networks. For
 352 each iteration, a small part of two datasets, that is the task dataset and
 353 the full training dataset, is used. The whole \mathcal{D} is sampled randomly only
 354 for $N_{\text{meta.iter}}$ iterations, using only few batches of images. The chosen task
 355 dataset $\mathcal{D}|_{\tau}$ is used for extracting domain specific images. The first part is
 356 the discriminator training. Domain loss and adversarial loss are computed
 357 as in Section 3.2, and \tilde{D} parameters are updated if the accuracy of the
 358 discriminator is under a certain threshold T . On the contrary, the second
 359 part, that is the generator training, is executed only if the accuracy of the
 360 discriminator is above a certain threshold t . During this step, adversarial,
 361 task reconstruction, domain, and feature losses are all employed to update
 362 \tilde{G} parameters.

363 **Inference.** The inference part is also a crucial one. In our work, we experi-
 364 ment the use of few images for adapting the trained model to new, unseen,
 365 domains, directly during the inference phase. The idea is to feed the trained

Algorithm 2 Inner training loop

Require: τ : extracted task in the outer loop

Require: $N_{\text{meta_iter}}$: number of inner epochs

Require: λ_D, λ_G : learning rates of D and G

Require: $w_{\text{adv}}, w_{\text{dom}}, w_{\text{rec}}, w_{\text{feat}}$: adversarial, domain, reconstruction, and feature weights

Require: t, T : minimum and maximum thresholds for discriminator accuracy

```
1: for  $i \in \{0, \dots, N_{\text{meta\_iter}}\}$  do
2:   sample  $y$  from  $\mathcal{D}|_{\tau}$ 
3:   sample  $x$  from  $\mathcal{D}$ 
4:    $\triangleright$  Discriminator training:
5:    $\varepsilon_D \leftarrow \nabla_{\theta_{\tilde{D}}} \mathcal{L}_{\text{adv}}(\tilde{D}, \tilde{G})$   $\triangleright x$  is considered fake,  $y$  real
6:    $\varepsilon_{\text{dom}} \leftarrow \nabla_{\theta_{\tilde{D}}} \mathcal{L}'_{\text{dom}}(\tilde{D})$   $\triangleright x$  is considered false,  $y$  true
7:   calculate accuracy  $a_{\tilde{D}}$  of discriminator  $\tilde{D}$ 
8:   if  $a_{\tilde{D}} < T$  then
9:      $\theta_{\tilde{D}} \leftarrow \theta_{\tilde{D}} - \lambda_D(\varepsilon_D + \varepsilon_{\text{dom}})$ 
10:  end if
11:  if  $a_{\tilde{D}} > t$  or  $i = 0$  then
12:     $\triangleright$  Generator training:
13:     $\varepsilon_G \leftarrow \nabla_{\theta_{\tilde{G}}} \mathbb{E}_{\tilde{G}(x) \sim p_{\tau}} [\log \tilde{D}(\tilde{G}(x))]$   $\triangleright \tilde{G}(x)$  is considered real
14:     $\varepsilon_{\text{task\_rec}} \leftarrow \nabla_{\theta_{\tilde{G}}} \mathcal{L}_{\text{rec}}(\tilde{G})$   $\triangleright$  the reconstruction is made with  $y$ 
15:     $\varepsilon_{\text{dom}} \leftarrow \nabla_{\theta_{\tilde{G}}} \mathcal{L}''_{\text{dom}}(\tilde{D}, \tilde{G})$   $\triangleright$  both  $y$  and  $\tilde{G}(x)$  are considered true
16:     $\varepsilon_{\text{feat}} \leftarrow \nabla_{\theta_{\tilde{G}}} \mathcal{L}_{\text{feat}}(\tilde{D}_{\text{feat}}, \tilde{G})$ 
17:     $\theta_{\tilde{G}} \leftarrow \theta_{\tilde{G}} - \lambda_G(w_{\text{adv}}\varepsilon_G + w_{\text{rec}}\varepsilon_{\text{rec}} + w_{\text{dom}}\varepsilon_{\text{dom}} + w_{\text{feat}}\varepsilon_{\text{feat}})$ 
18:  end if
19: end for
```

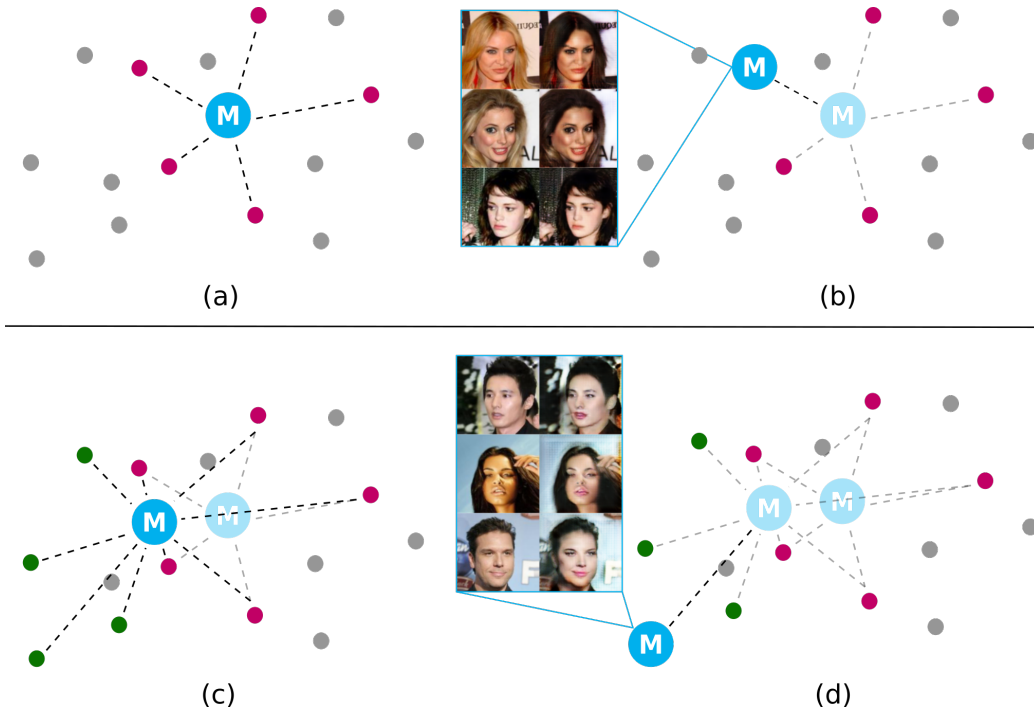


Figure 4: Inference algorithm in the case of one of the training domains, ‘Black Hair’ (b) and in the case of an unseen domain, ‘Heavy Makeup’ (d). The violet dots represent the domains used during training while the green dots are the unseen domains. The model M can move towards a known model, i.e. from state (a) to state (b); the other option is fine-tuning towards a set of unseen domains (c), then converge to a specific one (d).

380 to ensure a faster adaptation. The second part is used only for generating
 381 the results, and it resembles a more classic inference. The inner training
 382 loop of the meta-learning algorithm is used, but obtained parameters are not
 383 updated for the next domain.

384 An explanation of inference algorithm is visually provided in Figure 4
 385 where two exemplar cases are shown: the top row of the figure (Figure 4(a)
 386 and 4(b)) shows the case of the seen domain ‘Black Hair’, whereas the bottom
 387 row (Figure 4(c) and 4(d)) shows the case of unseen domain ‘Heavy Makeup’.
 388 In the image, the model, called M , consists of the G - D networks trained as
 389 in Algorithm 1 on five different example domains. In Figure 4(a), a naïve
 390 illustration of the domain space is provided. Given the dots as domains,
 391 and the violet dots as the seen domains, the model M has learned, during
 392 training, a sub-optimal representation for the seen domains. Intuitively, such

393 a representation permits the model to easily move towards the optimal one
394 in few steps of the inner training loop. In Figure 4(b), an example inference
395 on ‘Black Hair’ domain is shown. Since ‘Black Hair’ belongs to the trained
396 domains set, the model is cloned and with some steps of inner training loop,
397 the cloned model learns an optimal representation for the domain. Finally,
398 parameters of the cloned model are not saved nor used for updating the main
399 model, so the situation returns to the one depicted in Figure 4(a), ready to
400 make another inference. When a domain, or a set of domains, are completely
401 new to the generator and discriminator, a preliminary fine-tuning is needed.
402 In Figure 4(c), green dots represents these unseen domains. The inference
403 pre-training moves the (already trained) model in a sub-optimal position for
404 both the trained domains and the new domains. In Figure 4(d), the inference
405 on the updated model is shown. As Figure 4(d) shows, the fine-tuning for
406 the unseen domains moves the model M in a new “position” in the space,
407 closer to the unseen domains. In this way, during the inference (Figure 4(d))
408 the ‘Heavy Makeup’ domain is better learnt and more correct images are
409 generated, even if the domain has not been seen during the training.

410 Please note that the figure is completely exemplifying, since it depicts
411 domains in random positions, and does not take in account the intersec-
412 tion between them, nor their ‘real’ positioning in an actual domain space
413 (which is unknown). The idea of the model moving towards a sub-optimal
414 yet effective representation during training epochs—minimizing the expected
415 distance from all tasks—, and an informal proof of the idea using Euclidean
416 distances in the manifold of optimal solutions of a task, is provided in Reptile
417 paper (Nichol et al., 2018).

418 4. Experimental Results

419 This section presents visual and quantitative results of performed exper-
420 iments of MetalGAN, compared with StarGAN ones. All our experiments
421 were conducted using the CelebA dataset (Liu et al., 2015) which is a large-
422 scale face attributes dataset with more than 200k celebrity images, each with
423 40 attribute annotations. We decided to test our algorithm on this dataset
424 for three main reasons: first of all, since it contains images of faces with all
425 kind of attributes, it is suitable for multi-domain image-to-image task; sec-
426 ondly, it was used by StarGAN so it allows a clear comparison between the
427 results of the two different algorithms; and finally, even though our approach

428 is completely label-less, it is very easy to automatically divide a-priori the
 429 dataset in its different domains.

430 Experiments are divided into two categories: test results on seen do-
 431 mains (i.e., tasks the G - D networks were trained on), and results on unseen
 432 domains. In case of experiments on StarGAN, since their algorithm requires
 433 labels, we trained their network on some domains with a few number of im-
 434 ages (1000), and we call these “unseen” domains. This workaround permits
 435 us to compare StarGAN with our inference on unseen domains. It is worth
 436 to note that this approach is unfair for us, since StarGAN is fully trained
 437 for each of these “unseen” domains, while we only perform a small inference
 438 step. This is due to the fact that we can choose to add new domains to our
 439 network at every time, while StarGAN needs to define all the domains at the
 440 training stage.

441 4.1. Results on Trained Domains

Table 1: Hyper-parameters of MetalGAN training phase.

N_{epochs}	100000
λ_{ML}	0.01
λ_G, λ_D (Adam)	0.0001
$N_{\text{meta_iter}}$	20
batch size	16
w_{adv}	1
w_{dom}	1
w_{rec}	10
w_{feat}	1

442 We trained G - D networks model on 5 domains, namely ‘Eyeglasses’,
 443 ‘Male’, ‘Blond Hair’, ‘Black Hair’, and ‘Pale Skin’ for $N_{\text{epochs}} = 100000$ using
 444 the MetalGAN algorithm, and on the same domains for 200000 epochs using
 445 StarGAN.

446 Table 1 presents the main settings for our experiments. We set the Reptile
 447 learning rate λ_{ML} to 0.01 and optimized the generator and discriminator
 448 networks using Adam with a learning rate equals to 0.0001. Furthermore,
 449 we set the number of meta-iterations $N_{\text{meta_iter}}$ during training equals to



Figure 5: Results on training classes. In the first column, the input images. From second to fifth column there are the outputs of the model moved towards the respective domain, in case of MetalGAN, or labeled with the indication of the domain, in case of StarGAN. The last column of MetalGAN results is the output of the model without moving it from the sub-optimum.

450 20 since we empirically found that this value represents the best trade-off
 451 between speed and accuracy of the algorithm. For coherence with StarGAN,
 452 batch size is set to 16 during training. Weights for MetalGAN objective
 453 during training are left to 1 except for reconstruction weight, that is set to
 454 10, in order to obtain an accurate reconstruction of the image and gain more
 455 quality in results.

456 Figure 5 shows some visual results on a batch of eight input images.
 457 Figure 5(a) contains the outputs of MetalGAN algorithm, while Figure 5(b)
 458 shows the StarGAN outputs. In addition, a greater number of results on some
 459 of the training classes are shown in Figure 6 and 7. Figure 6 shows generated
 460 images on ‘Eyeglasses’ domain, where input images are put side-by-side to
 461 MetalGAN outputs and StarGAN outputs. In the same fashion, results on
 462 ‘Black Hair’ domain are reported in Figure 7. We decided to choose these
 463 two domains since they are very different in terms of features and since our

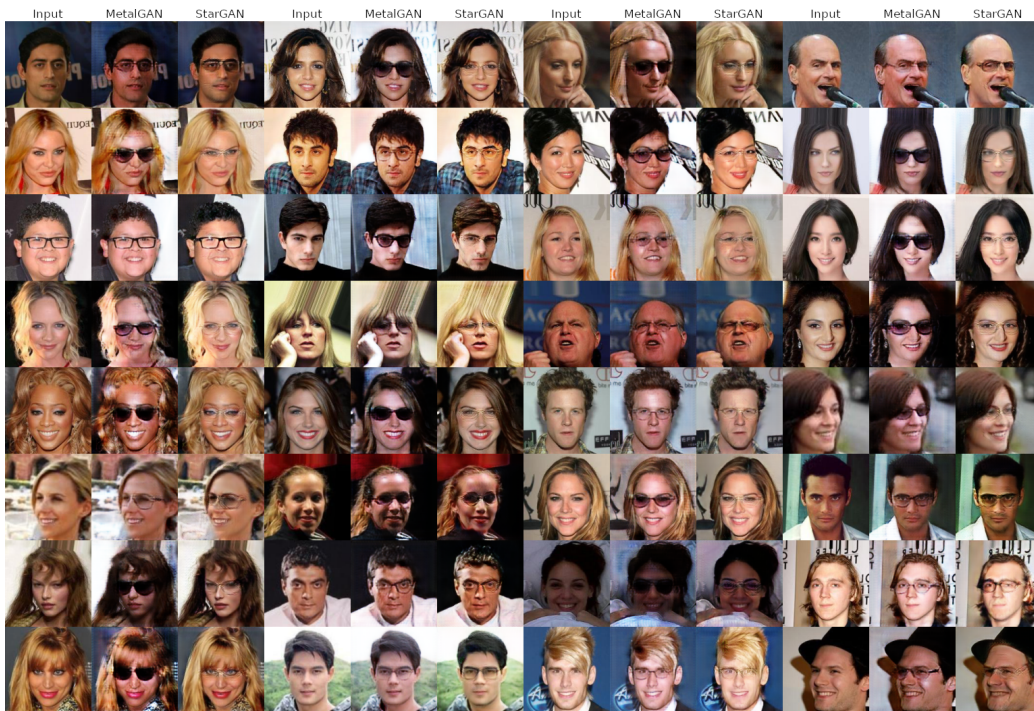


Figure 6: Results on training domain Eyeglasses. The image triplets are composed by input image, MetalGAN output and StarGAN output.

464 method performs very well on ‘Eyeglasses’ and, on the contrary, it is not so
 465 good on ‘Black Hair’. It is also worth noting that MetalGAN on ‘Eyeglasses’
 466 produces a great variability of examples, compared to StarGAN, generating
 467 both simple glasses and sunglasses.

468 However, the image generation should be considered successful and visu-
 469 ally close to StarGAN one. As a matter of fact, we can see how our label-less
 470 approach produces results that are visually very similar to the ones produced
 471 by StarGAN. In particular, our algorithm is able to understand the different
 472 target domains just by seeing few examples of them each epoch, and can
 473 correctly produce these domains from the input images even without labels
 474 or supervision.

475 In addition, we performed quantitative analysis of the produced results.
 476 As far as we know, no pure theoretical framework is available for a precise
 477 quantification of our model contributions and advantages, in order to com-
 478 pare it to others, but there exists some relevant metrics that are suitable for
 479 a numerical placement of our proposal. Metrics considered in this work are



Figure 7: Results on training domain Black Hair. The image triplets are composed by input image, MetalGAN output and StarGAN output.

480 FID (Frechet Inception Distance) (Heusel et al., 2017) and PRD (Precision
 481 and Recall for Distributions) (Sajjadi et al., 2018), described below. We use
 482 FID to calculate the distribution matching between the original CelebA im-
 483 ages for each training domains and our results, and we compare our score
 484 with the one obtained on StarGAN images. This comparison is presented
 485 in Figure 8 with lower values indicating the better scores. Our method per-
 486 forms slightly better than StarGAN for ‘Eyeglasses’, ‘Male’ and ‘Blond Hair’
 487 domains and slightly worse than StarGAN for ‘Black Hair’ and ‘Pale Skin’,
 488 confirming the visual evaluation of the images.

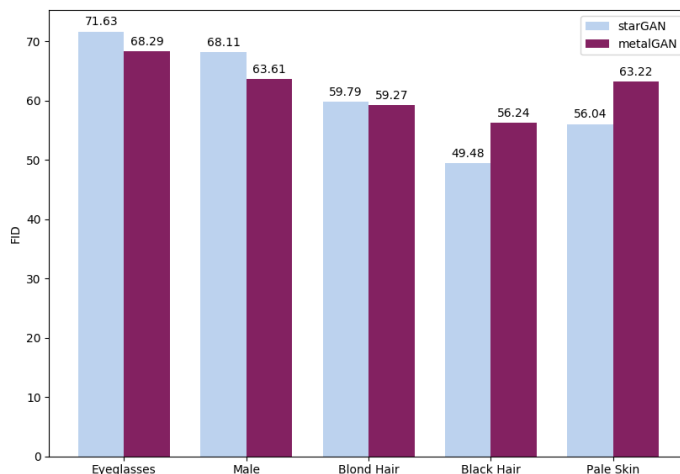


Figure 8: FID score on training domains (the lower the better).

489 Another quantitative analysis is based on PRD for both StarGAN and
 490 MetalGAN methods, using classes of images of CelebA as target datasets.
 491 Precision is a measure of raw quality of generated images, and does not take
 492 in account the internal variability of the distribution, while recall measures
 493 how well the generated images resembles the “class distribution” of the tar-
 494 get dataset. We choose to measure a single domain at once. Results are
 495 shown in five different graphics, in Figure 9. As shown, MetalGAN PRD on
 496 ‘Eyeglasses’, ‘Blond Hair’, and ‘Black Hair’ are very similar to each other
 497 and close to StarGAN results. The main difference between StarGAN and
 498 MetalGAN in case of hair domains is that StarGAN is usually more precise
 499 (it produces images with a better quality w.r.t. the target distribution), but
 500 it has a lower recall, meaning that the distribution of StarGAN generated
 501 images is less varied than MetalGAN one. Regarding ‘Male’ and ‘Pale Skin’,

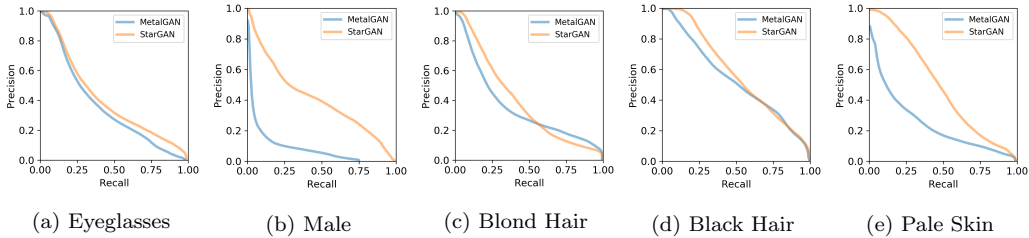


Figure 9: PRD on each training domains.

502 the precision of MetalGAN suffers from the fact that such domains require a
 503 significant change in all the faces in the input image, highlighting a weakness
 504 in MetalGAN global reconstruction. On the other hand, the domain change
 is successful, as confirmed for ‘Male’ FID score. In Figure 10, global PRD,

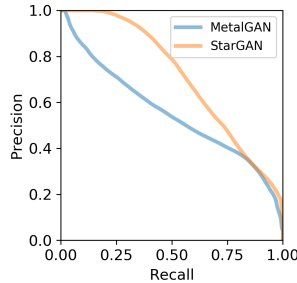


Figure 10: Global PRD on training domains.

505
 506 computed on all training domains at once, resembles the previous consider-
 507 ation, showing a worse precision of MetalGAN generated distribution, but a
 508 similar high recall for StarGAN and MetalGAN distributions.

509 It is worth emphasizing once more that MetalGAN achieves these results
 510 without labels, showing in any case comparable quantitative results and often
 511 better qualitative results.

512 4.2. Results on Unseen Domains

513 During the inference step, we modify the hyper-parameters of MetalGAN
 514 as shown in Table 2. In particular, in order to allow the network to quickly
 515 adapt to the new domains, we increment the λ_{ML} to 0.1 and we set w_{adv} and
 516 w_{dom} to 100. Furthermore, since the network already learned to reconstruct
 517 the content of the input images we lower w_{rec} to 1.

Table 2: Hyper-parameters of MetalGAN inference phase.

N_{epochs}	10
λ_{ML}	0.1
λ_G, λ_D (Adam)	0.0001
$N_{\text{inf_train}}$ (train)	20
$N_{\text{inf_test}}$ (test)	100
batch size	16
w_{adv}	100
w_{dom}	100
w_{rec}	1
w_{feat}	1

518 Finally, we tested the MetalGAN trained model on 6 unseen domains,
 519 namely ‘Big Lips’, ‘Bushy Eyebrows’, ‘Heavy Makeup’, ‘Smiling’, ‘Gray Hair’,
 520 and ‘Mustache’ using the MetalGAN inference. MetalGAN, trained on the
 521 5 seen domains of Section 4.1, performs 10 further outer iterations (on each
 522 new domain), each of them consisting of 20 inner iteration, where 320 task
 523 images are seen for the first time. In this way, a fine-tuned model is obtained,
 524 as in Figure 4(c). Then, images are generated by specializing the fine-tuned
 525 model on the chosen domain, as in Figure 4(d). Such a specialization is done
 526 performing 100 inner iterations per domain.

527 On the other side, we trained 6 *new* StarGAN models with the same
 528 domains used during training, plus one unseen domain for each model, i.e. we
 529 obtained a StarGAN model specialized also in ‘Big Lips’, another StarGAN
 530 model specialized also on ‘Bushy Eyebrows’, and so on. This is necessary,
 531 since StarGAN uses image labels, so adding a new domain is possible only
 532 by retraining the model. All six new StarGAN models were fully trained
 533 for $N_{\text{epochs}} = 200000$. For StarGAN, “unseen” means that only 1000 input
 534 images are selected for that domain, as already described in the beginning of
 535 Section 4.

536 Visual qualitative results for unseen domains for both MetalGAN and
 537 StarGAN are presented in Figure 11, 12 and 13. In particular, for MetalGAN,
 538 the results produced without performing the fine-tuning iterations are also
 539 shown. Our algorithm is able to produce compelling images even in this case
 540 and further improves the visual appearance of the images after the fine-tuning

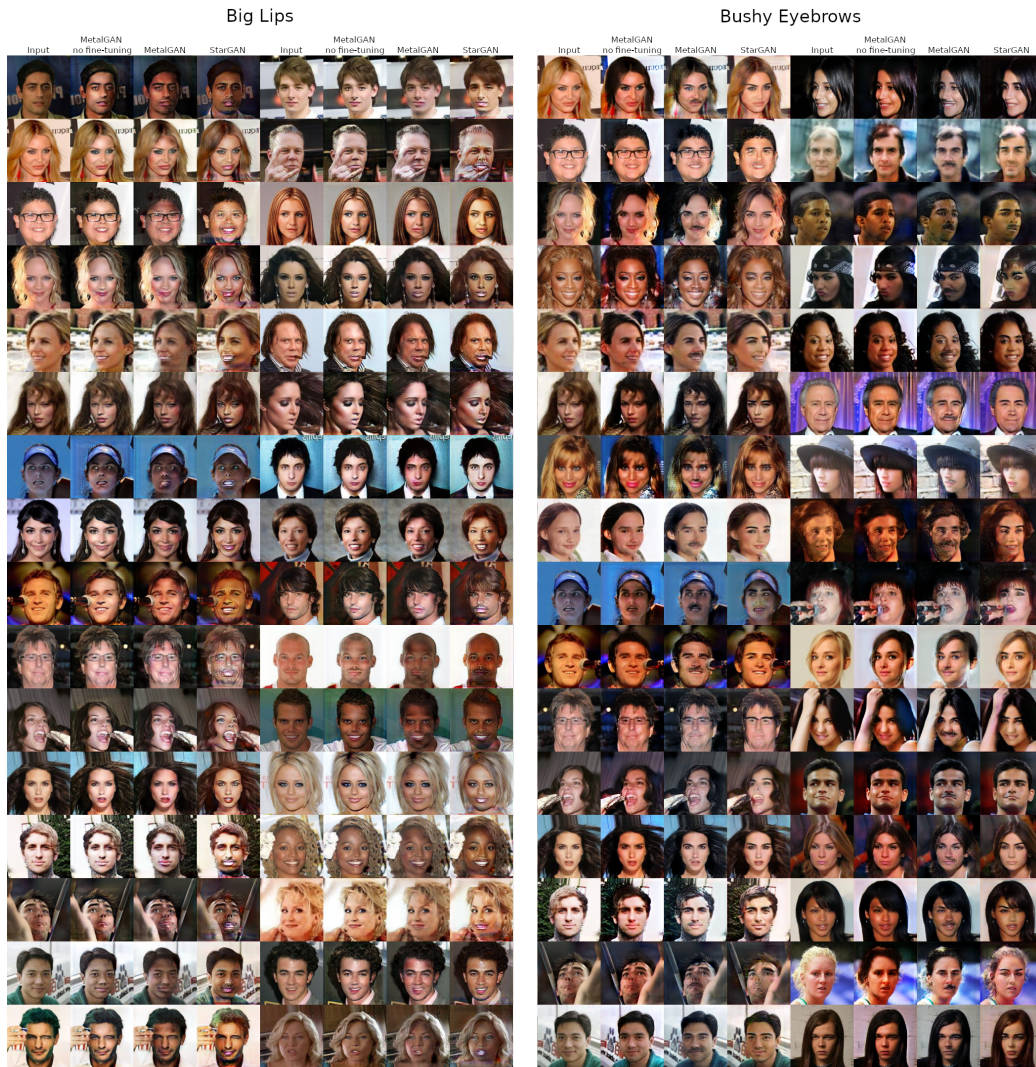


Figure 11: Results on unseen domains Big Lips and Bushy Eyebrows. The image triplets are composed by input image, MetalGAN output without fine-tuning, MetalGAN output with fine-tuning and StarGAN output.

541 step. In addition, MetalGAN applies the unseen domains to the input images
542 in a more soft and natural way than StarGAN. This is particularly evident
543 in ‘Big Lips’ and ‘Smiling’, where StarGAN produced results that could be
544 described as “creepy”. A further consideration is the fact that sometimes,
545 during the unseen domain transfer, MetalGAN tends to apply unwanted
546 features to the images. For example in ‘Bushy Eyebrows’ the network often
547 changes the hair color to black or applies mustaches. This is due to the fact
548 that people with bushy eyebrows generally have darker hair and facial hair.
549 The same reasoning can be applied to ‘Gray Hair’, where the network tends
550 to produce older people, because people with gray hair are usually old. The
551 reason for this behavior is that, because of the lack of labels, the network has
552 to infer which is the domain to be transferred without any help. This is also
553 a big advantage, because produces a much greater flexibility to the network
554 and allows to add new domains to the network very easily.

555 We calculated both FID and PRD also for inference domains, as in the
556 previous section. In Figure 14, FID scores are reported. As for training,
557 FID scores depend heavily on the selected domain, but in general, StarGAN
558 and MetalGAN scores are close to each other. In particular, MetalGAN
559 performs better on ‘Big Lips’, ‘Smiling’, and ‘Bushy Eyebrows’, confirming
560 visual evaluation of results.

561 In Figure 15, PRD graphs for each unseen domain are reported. All
562 results show how both StarGAN and MetalGAN decrease their precision
563 in this phase, as reasonable. As we can see in Figure 11, 12, and 13, the
564 overall quality of the reconstruction is slightly worse than the one of trained
565 domains. However, despite the unfair comparison, PRD for MetalGAN and
566 StarGAN are pretty similar. Looking at the global PRD, calculated on all six
567 unseen domains at once (Figure 16), MetalGAN shows better performances
568 especially on distribution recall.

569 4.3. Additional Experiments

570 4.3.1. Results on Radboud Faces Database

571 In order to further prove the effectiveness of our method, we also trained
572 the G - D network on another multi-domain dataset, with MetalGAN algo-
573 rithm. Such a dataset is Radboud Faces Database (RAFD) (Langner et al.,
574 2010). RAFD is a set of pictures of 67 models displaying 8 emotional ex-
575 pressions. We trained the model for 20k iterations with MetalGAN on 5
576 different emotions (*disgusted*, *fearful*, *happy*, *sad* and *surprised*), maintain-
577 ing the same configuration used with the CelebA dataset. A batch of visual



Figure 12: Results on unseen domains Heavy Makeup and Smiling. The image triplets are composed by input image, MetalGAN output without fine-tuning, metalGAN output with fine-tuning and starGAN output.

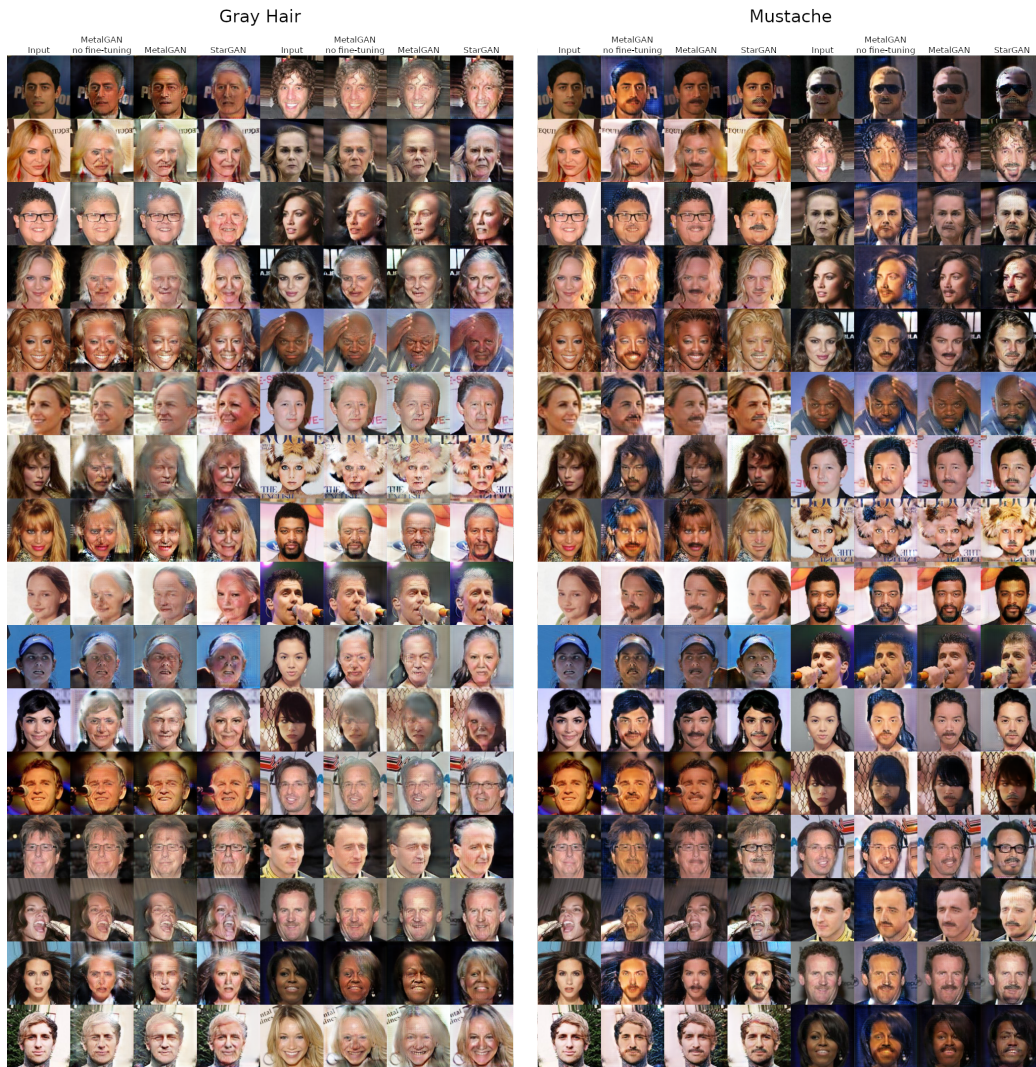


Figure 13: Results on unseen domains Gray Hair and Mustache. The image triplets are composed by input image, MetalGAN output without fine-tuning, metalGAN output with fine-tuning and starGAN output.

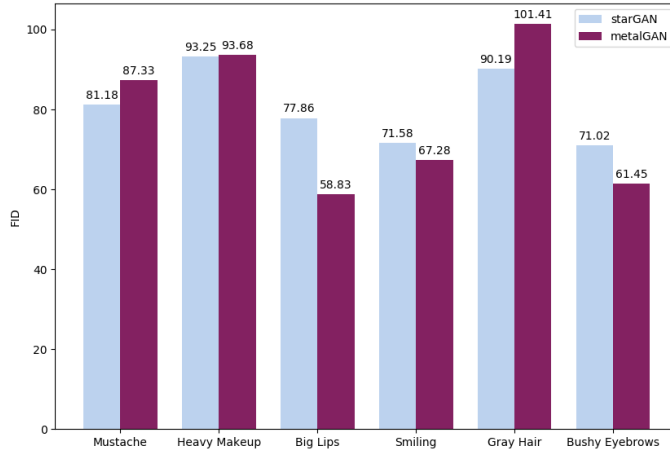


Figure 14: FID score on inference domains (the lower the better).

578 results is shown in Figure 17 (a), where only the trained domains are tested.
 579 Then, additional unseen domains were added to further test our method on
 580 this new dataset, as for CelebA. Such new domains are *angry*, *contemptuous*
 581 and *neutral*. Inference configurations are the same of Table 2. In Figure
 582 17 (b), the visual results of MetalGAN inference on RAFD are reported.
 583 Results are comparable to trained ones, even if they were obtained by few
 584 iterations on the trained model, and with few input images. The naïve reason
 585 could be that changing the facial expression involves few attributes of the
 586 image, thus switching from the input facial expression domain to an unseen
 587 one shares a lot of knowledge with the switching between the input and the
 588 trained domains. In other words, the main task is the same: changing the
 589 facial expression, and little differences between domains are handled easily
 590 by the inference steps.

591 In addition to the qualitative comparison, we also trained a classification
 592 network in order to obtain a quantitative evaluation of our method. We
 593 choose ResNet-18 as classification network (following the StarGAN paper)
 594 and we produced classification results on the different emotions both for our
 595 architecture as well as for StarGAN trained on the same domains. Results
 596 can be seen in Table 3. Following the considerations that were made for
 597 the CelebA results, our results for the RAFD dataset are in line with the
 598 StarGAN ones, but without the use of label or supervision. The only excep-
 599 tion is the *sad* domain where our network tends to only change the mouth

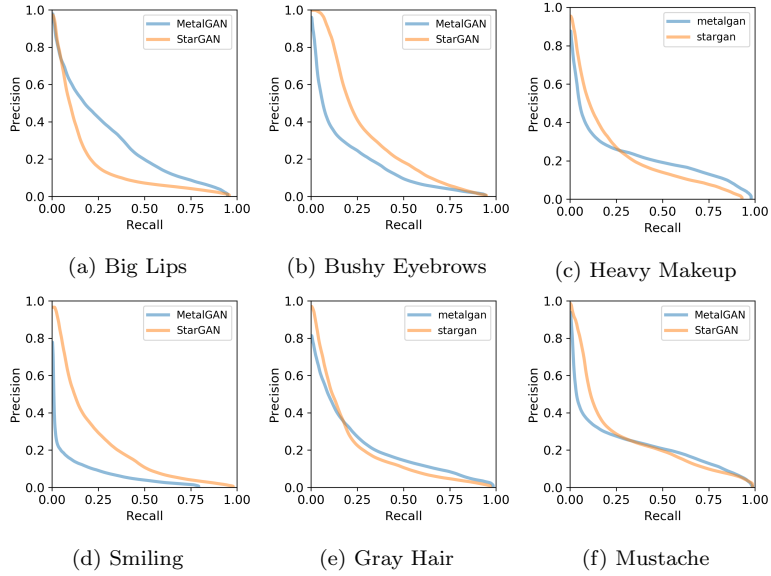


Figure 15: PRD graphs on inference domains.

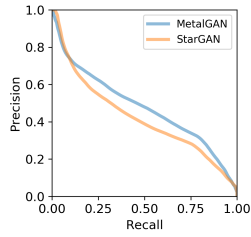


Figure 16: Global PRD on inference domains.

600 leaving the rest of the face almost unchanged. Therefore, if the input image
 601 has another emotion strongly characterized by the eyes or by the eyebrows
 602 (such as *surprised*), such features are not changed during the domain switch
 603 leading to misclassification.

Table 3: Classification results on RAFD dataset.

	disgusted	fearful	happy	sad	surprised
StarGAN	98.6%	97.2%	98.6%	97.7%	97.1%
MetalGAN (ours)	98.4%	93.1%	97.3%	69.7%	95.2%



(a) Training domains.

(b) Unseen domains.

Figure 17: MetalGAN results on RAFD on trained and unseen domains. Columns in first image represent disgusted, fearful, happy, sad, and surprised domains. Columns in second image represent angry, contemptuous, and neutral domains.



Figure 18: Results on CelebA without the contribution of Domain Loss.

604 *4.3.2. Contribution of Loss Functions in the Experiments*

605 The effectiveness of each loss function of the architecture (see Section
 606 3.2) is discussed below. The basic GAN structure, as theoretically proposed

607 in (Goodfellow et al., 2014), relies on the Adversarial Loss function (being
608 a minimax two-players game between G and D networks). Hence, every
609 GAN model takes advantage of such a loss function. We did not choose to
610 empirically test its effectiveness since the only way is to get rid of the loss
611 and see what changes happen to the results. But performing an experiment
612 without the Adversarial Loss function is unfeasible for GAN architectures,
613 unless the goal is to change the fundamentals of GAN, which is clearly out
614 of the scope of this paper.

615 The necessity of the Domain Loss in the MetalGAN algorithm is not
616 self-evident: for this reason, we performed an experiment with the same con-
617 figurations of MetalGAN standard training (see Table 1), but nullifying the
618 domain weight, i.e. $w_{\text{dom}} = 0$. It is worth noting that this setting still relies
619 on meta-learning, that is the major boost for domain adaptation without
620 labels. Nevertheless, visual results on CelebA, for MetalGAN without Do-
621 main Loss, show that the domain change loses quality, as it should be seen
622 in Figure 18.

623 The Reconstruction Loss is used to keep unchanged the facial feature of
624 the subject of the input image. Figure 5 last column of MetalGAN results
625 shows, in fact, the only contribution of Reconstruction Loss, before moving
626 the model to the optimum position for a certain domain (inference). Re-
627 construction Loss also makes sure that input images that already belong to
628 the chosen output domain are left completely unchanged by our architecture.
629 Performed experiments show this behavior, e.g. in Figure 5 fifth row, second
630 column: the input subject wear eyeglasses, so there is no need to change its
631 domain, and MetalGAN simply transfer the eyeglasses of the input image
632 into the output one, without changing facial attributes.

633 Finally, the Feature Loss is a base building block of cGAN for regular-
634 ization. As detailed in (Wang et al., 2018), and briefly remarked in Section
635 3.2, Feature Loss stabilizes the training and it is a common state-of-the-art
636 method for cGAN.

637 5. Conclusions

638 We proposed a new architecture for multi-domain label-less image-to-
639 image translation. Our system has many features that distinguish it from
640 the state-of-the-art.

641 First of all, instead of relying on labels for switching the domains, like
642 other state-of-the-art architectures, we had chosen to use meta-learning and

643 in particular Reptile. Furthermore, getting rid of labels allowed the archi-
644 tecture to be more flexible, since there is no need of providing hard-coded
645 vectors of labels. It is possible to arbitrarily change the number of domains,
646 and to add a new one during inference. Such an approach was completely
647 unfeasible in previous algorithms like StarGAN, that needs hard-coded la-
648 bels at training time, and this was a very serious limitation. Finally, beside
649 the lack of labels, an immediate advantage of the meta-learning approach is
650 that such a method has been used for few-shot learning. Not only, as high-
651 lighted above, a new, unseen, task can be added, but in order to do so, just
652 few examples are needed, and neither tedious and long-lasting annotations
653 of labels, nor a full retraining of the model are required.

654 We proved the effectiveness of our approach with face attributes transfer
655 using the CelebA dataset, and we evaluated it using both FID and PRD
656 quantitative metrics. Moreover, we performed additional experiments on
657 RAFD dataset, and tested our approach by nullifying the contribution of
658 Domain Loss, showing its necessity.

659 Regarding future work, our first objective would be to explore more deeply
660 the possibilities and limitations of meta-learning in order to further improve
661 our algorithm and to prove its effectiveness on others tasks like image gen-
662 eration and semantic segmentation.

663 **References**

664 Almahairi, A., Rajeswar, S., Sordoni, A., Bachman, P., Courville, A., 2018.
665 Augmented cyclegan: Learning many-to-many mappings from unpaired
666 data. arXiv preprint arXiv:1802.10151 .

667 Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M.W., Pfau, D., Schaul,
668 T., Shillingford, B., De Freitas, N., 2016. Learning to learn by gradient
669 descent by gradient descent, in: Advances in neural information processing
670 systems, pp. 3981–3989.

671 Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J., 2018. Stargan:
672 Unified generative adversarial networks for multi-domain image-to-image
673 translation, in: Proceedings of the IEEE Conference on Computer Vision
674 and Pattern Recognition, pp. 8789–8797.

675 Clouâtre, L., Demers, M., 2019. Figr: Few-shot image generation with reptile.
676 arXiv preprint arXiv:1901.02199 .

- 677 Duan, Y., Schulman, J., Chen, X., Bartlett, P.L., Sutskever, I., Abbeel, P.,
678 2016. RL^2 : Fast reinforcement learning via slow reinforcement learning.
679 arXiv preprint arXiv:1611.02779 .
- 680 Finn, C., Abbeel, P., Levine, S., 2017. Model-agnostic meta-learning for fast
681 adaptation of deep networks, in: Proceedings of the 34th International
682 Conference on Machine Learning-Volume 70, JMLR. org. pp. 1126–1135.
- 683 Fontanini, T., Iotti, E., Prati, A., 2019. MetalGAN: a cluster-based adaptive
684 training for few-shot adversarial colorization, in: International Conference
685 on Image Analysis and Processing (ICIAP), pp. 280–291.
- 686 Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D.,
687 Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets, in:
688 Advances in neural information processing systems, pp. 2672–2680.
- 689 He, Z., Zuo, W., Kan, M., Shan, S., Chen, X., 2019. Attgan: Facial attribute
690 editing by only changing what you want. IEEE Transactions on Image
691 Processing .
- 692 Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S., 2017.
693 Gans trained by a two time-scale update rule converge to a local nash
694 equilibrium, in: Advances in Neural Information Processing Systems, pp.
695 6626–6637.
- 696 Hochreiter, S., Younger, A.S., Conwell, P.R., 2001. Learning to learn using
697 gradient descent, in: International Conference on Artificial Neural Net-
698 works, Springer. pp. 87–94.
- 699 Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A., 2017. Image-to-image transla-
700 tion with conditional adversarial networks, in: Proceedings of the IEEE
701 conference on computer vision and pattern recognition, pp. 1125–1134.
- 702 Kim, T., Cha, M., Kim, H., Lee, J.K., Kim, J., 2017a. Learning to discover
703 cross-domain relations with generative adversarial networks, in: Proceed-
704 ings of the 34th International Conference on Machine Learning-Volume 70,
705 JMLR. org. pp. 1857–1865.
- 706 Kim, T., Kim, B., Cha, M., Kim, J., 2017b. Unsupervised visual at-
707 tribute transfer with reconfigurable generative adversarial networks. arXiv
708 preprint arXiv:1707.09798 .

- 709 Langner, O., Dotsch, R., Bijlstra, G., Wigboldus, D.H., Hawk, S.T.,
710 Van Knippenberg, A., 2010. Presentation and validation of the radboud
711 faces database. *Cognition and emotion* 24, 1377–1388.
- 712 Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A.,
713 Aitken, A., Tejani, A., Totz, J., Wang, Z., et al., 2017. Photo-realistic
714 single image super-resolution using a generative adversarial network, in:
715 Proceedings of the IEEE conference on computer vision and pattern recog-
716 nition, pp. 4681–4690.
- 717 Li, K., Malik, J., 2017. Learning to optimize neural nets. arXiv preprint
718 arXiv:1703.00441 .
- 719 Liu, M.Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., Kautz,
720 J., 2019. Few-shot unsupervised image-to-image translation, in: IEEE
721 International Conference on Computer Vision (ICCV).
- 722 Liu, Z., Luo, P., Wang, X., Tang, X., 2015. Deep learning face attributes in
723 the wild, in: Proceedings of International Conference on Computer Vision
724 (ICCV).
- 725 Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks
726 for semantic segmentation, in: Proceedings of the IEEE conference on
727 computer vision and pattern recognition, pp. 3431–3440.
- 728 Mirza, M., Osindero, S., 2014. Conditional generative adversarial nets. arXiv
729 preprint arXiv:1411.1784 .
- 730 Mishra, N., Rohaninejad, M., Chen, X., Abbeel, P., 2017. A simple neural
731 attentive meta-learner. arXiv preprint arXiv:1707.03141 .
- 732 Nichol, A., Achiam, J., Schulman, J., 2018. On first-order meta-learning
733 algorithms. *CoRR*, abs/1803.02999 2.
- 734 Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y., 2019. Semantic image syn-
735 thesis with spatially-adaptive normalization, in: Proceedings of the IEEE
736 Conference on Computer Vision and Pattern Recognition, pp. 2337–2346.
- 737 Ravi, S., Larochelle, H., 2016. Optimization as a model for few-shot learning
738 .

- 739 Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks
740 for biomedical image segmentation, in: International Conference on Med-
741 ical image computing and computer-assisted intervention, Springer. pp.
742 234–241.
- 743 Sajjadi, M.S., Bachem, O., Lucic, M., Bousquet, O., Gelly, S., 2018. As-
744 sessing generative models via precision and recall, in: Advances in Neural
745 Information Processing Systems, pp. 5228–5237.
- 746 Sangkloy, P., Lu, J., Fang, C., Yu, F., Hays, J., 2017. Scribbler: Controlling
747 deep image synthesis with sketch and color, in: Proceedings of the IEEE
748 Conference on Computer Vision and Pattern Recognition, pp. 5400–5409.
- 749 Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., Lillicrap, T., 2016.
750 Meta-learning with memory-augmented neural networks, in: International
751 conference on machine learning, pp. 1842–1850.
- 752 Schmidhuber, J., 2020. Generative adversarial networks are special cases of
753 artificial curiosity (1990) and also closely related to predictability mini-
754 mization (1991). *Neural Networks* .
- 755 Wang, J.X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J.Z., Munos,
756 R., Blundell, C., Kumaran, D., Botvinick, M., . Learning to reinforcement
757 learn, 2016. arXiv preprint arXiv:1611.05763 .
- 758 Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B., 2018.
759 High-resolution image synthesis and semantic manipulation with condi-
760 tional gans, in: Proceedings of the IEEE conference on computer vision
761 and pattern recognition, pp. 8798–8807.
- 762 Wichrowska, O., Maheswaranathan, N., Hoffman, M.W., Colmenarejo, S.G.,
763 Denil, M., de Freitas, N., Sohl-Dickstein, J., 2017. Learned optimizers that
764 scale and generalize, in: Proceedings of the 34th International Conference
765 on Machine Learning-Volume 70, JMLR. org. pp. 3751–3760.
- 766 Xian, W., Sangkloy, P., Agrawal, V., Raj, A., Lu, J., Fang, C., Yu, F.,
767 Hays, J., 2018. Texturegan: Controlling deep image synthesis with texture
768 patches, in: Proceedings of the IEEE Conference on Computer Vision and
769 Pattern Recognition, pp. 8456–8465.

- 770 Xiao, T., Hong, J., Ma, J., 2017. Dna-gan: learning disentangled repre-
771 sentations from multi-attribute images. arXiv preprint arXiv:1711.05415
772 .
- 773 Zhang, R., Che, T., Ghahramani, Z., Bengio, Y., Song, Y., 2018. Meta-
774 gan: An adversarial approach to few-shot learning, in: Advances in Neural
775 Information Processing Systems, pp. 2365–2374.
- 776 Zhu, J.Y., Park, T., Isola, P., Efros, A.A., 2017a. Unpaired image-to-image
777 translation using cycle-consistent adversarial networks, in: Proceedings of
778 the IEEE international conference on computer vision, pp. 2223–2232.
- 779 Zhu, J.Y., Zhang, R., Pathak, D., Darrell, T., Efros, A.A., Wang, O., Shecht-
780 man, E., 2017b. Toward multimodal image-to-image translation, in: Ad-
781 vances in Neural Information Processing Systems, pp. 465–476.