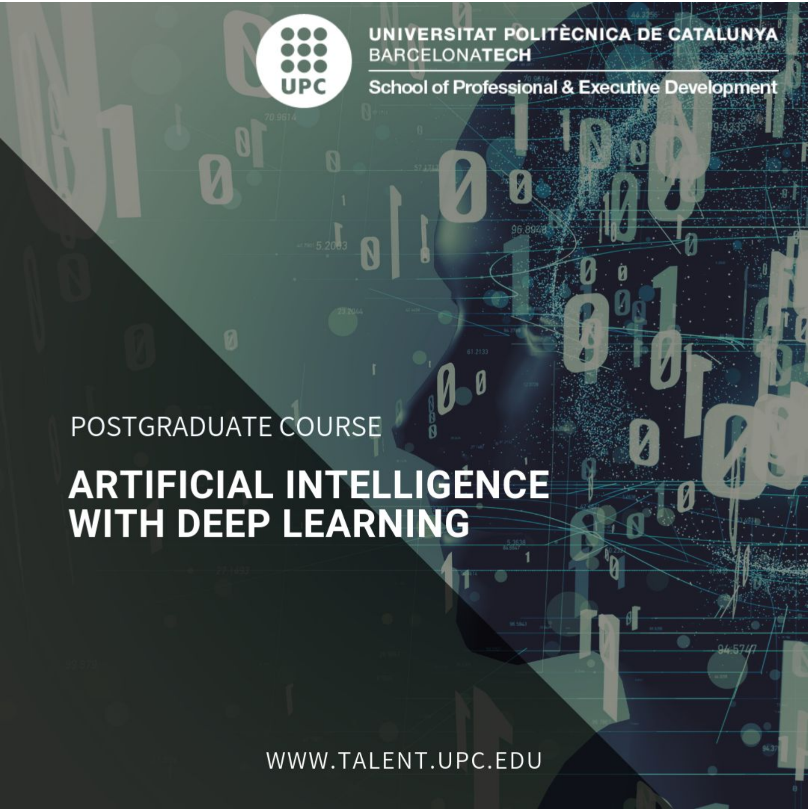



Final REVIEW

# GAN HYPER RESOLUTION

**Homepage:** https://github.com/AIDL-PROJ-2022/enhanced-srgan

# FINAL REVIEW

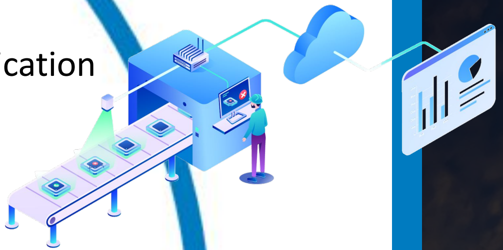



PROJECT ADVISOR:
DANI FOJO

MARC BERMEJO
RAUL PUENTE
FERRAN TORRES

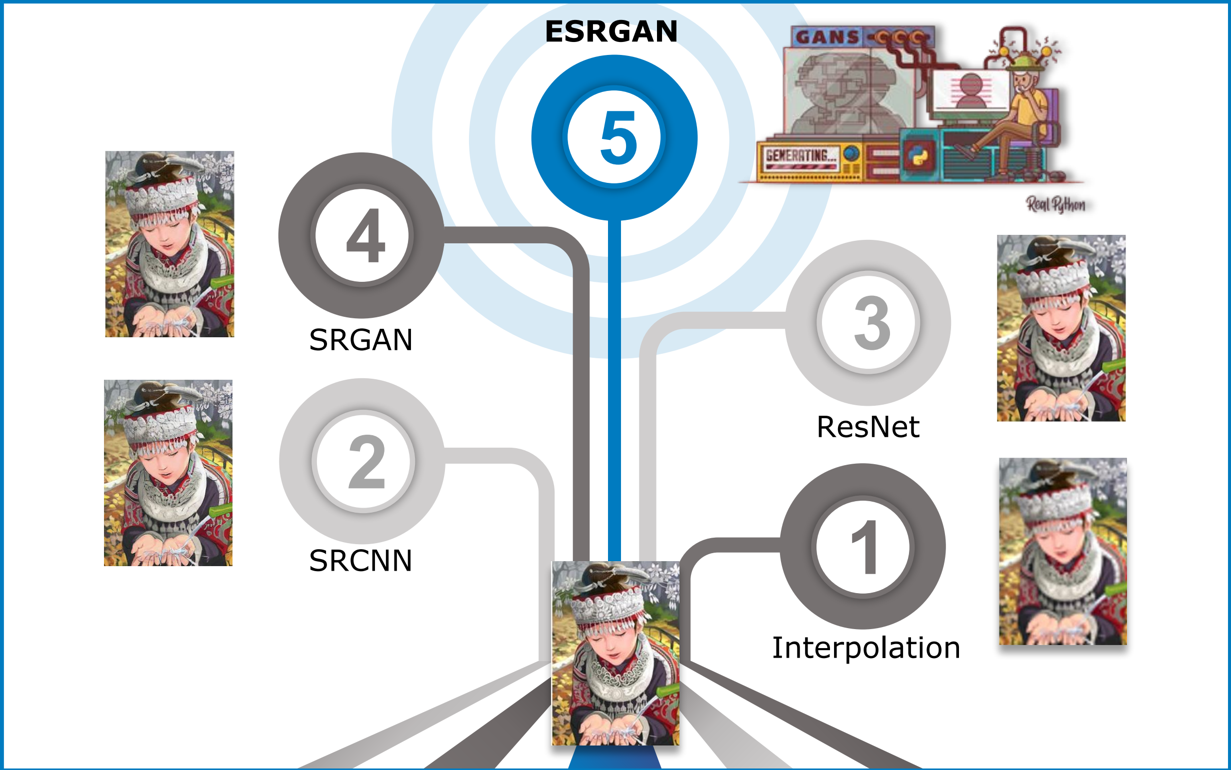
# MOTIVATION OF THE PROJECT

Super Resolution and its application



Deeper understanding of GANs

Solution at business level




# INTRODUCTION TO SUPER RESOLUTION

*"The estimation of a **high-resolution (HR)** image from **a single low-resolution (LR)** counterpart is referred to as **super-resolution (SR)**.*


PROJECT ADVISOR:
DANI FOJO

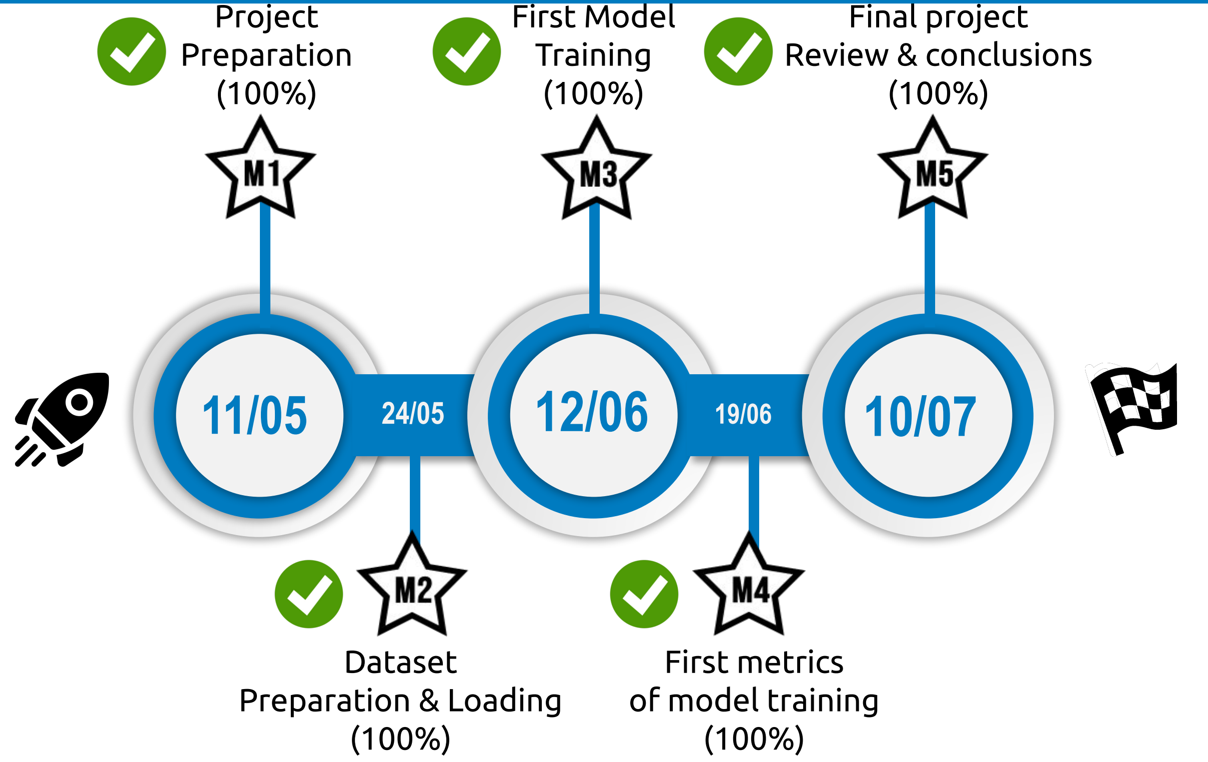


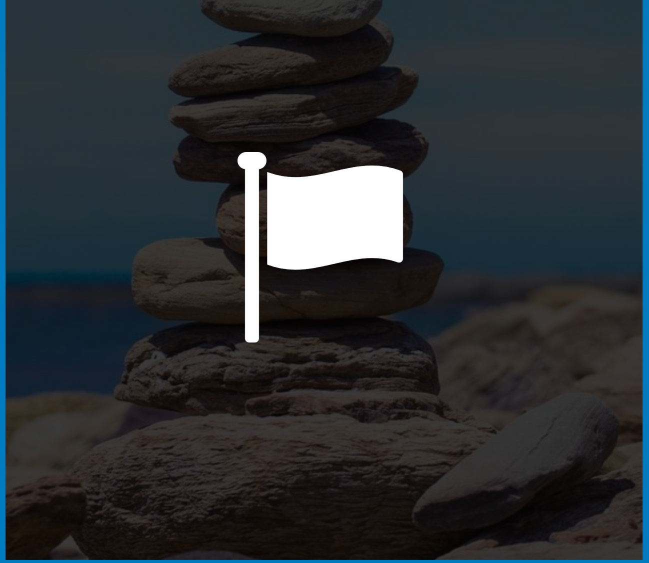

MARC BERMEJO

RAUL PUENTE

FERRAN TORRES

# PROJECT MILESTONES



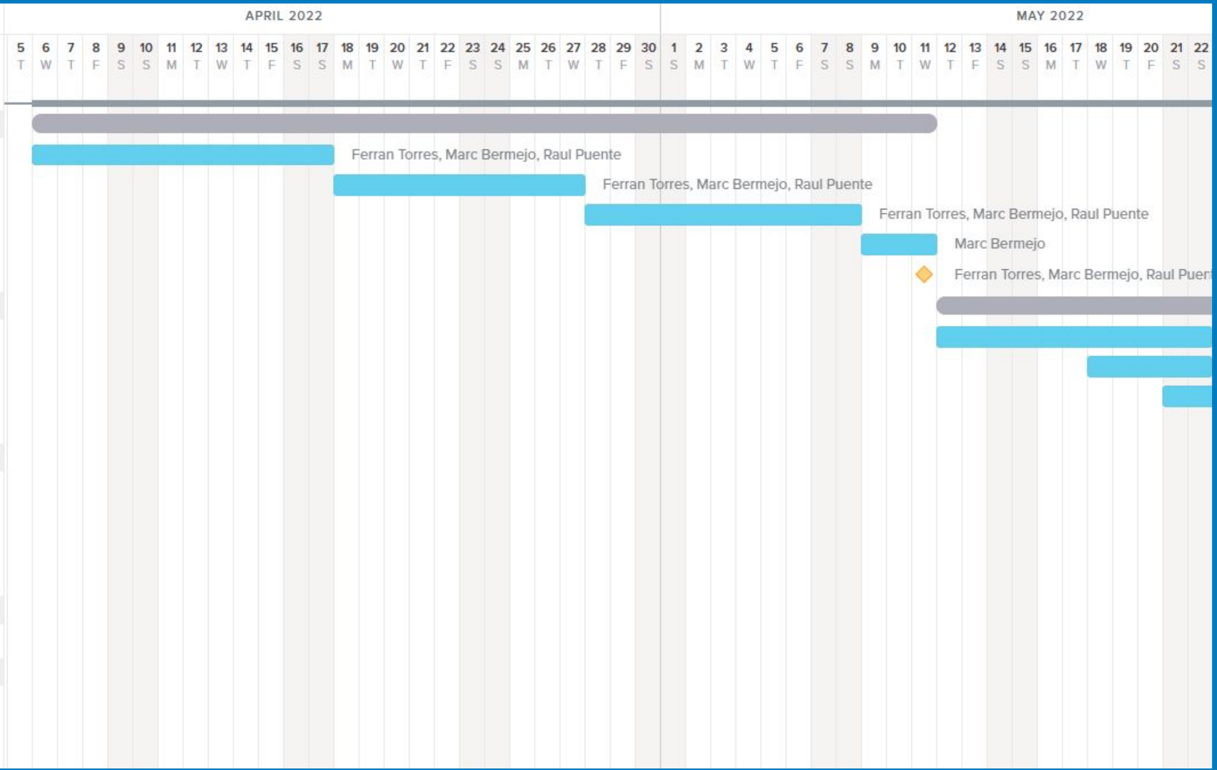



PROJECT ADVISOR:
DANI FOJO



MARC BERMEJO
RAUL PUENTE
FERRAN TORRES

| | Assigned | Progress |
|---|---|---|
| Torch ESRGAN | | 100% |
| ▾ Project Kick-off | | 100% |
| Look for SR models using GANs | Ferran Torres, Marc | 100% |
| Read ESRGAN paper (https://arxiv.org/pdf/1809.00219.pdf) | Ferran Torres, Marc | 100% |
| Look for other implementations of ESRGAN network | Ferran Torres, Marc | 100% |
| Decide initial project structure | Marc Bermejo | 100 |
| Project preparation completed | Ferran Torres, Marc | ☑ |
| ▾ Initial Dataset loading | | 100% |
| Implement base dataset class | Marc Bermejo | 100 |
| Implement image augmentation pipelines | Marc Bermejo | 100 |
| Implement BSDS500 dataset class | Marc Bermejo | 100 |
| Dataset preparation and loading ready | Marc Bermejo | ☑ |
| ▾ Generator model | | 100% |
| Implement residual dense block module | Marc Bermejo | 100 |
| Implement residual on residual dense block module | Marc Bermejo | 100 |
| Implement sub-pixel convolution module | Marc Bermejo | 100 |
| Implement full SR generator neural model | Marc Bermejo | 100 |
| ▾ Discriminator model | | 100% |
| Implement relativistic discriminator of ESRGAN model | Marc Bermejo | 100 |
| ▾ Training | | 100% |
| Create initial training script with hard-coded hyperparameters | Marc Bermejo | 100 |
| First model training | Marc Bermejo | ☑ |
| Store and load check-points during training | Marc Bermejo, Raul | 100 |




PROJECT ADVISOR:
DANI FOJO

GAN HYPER RESOLUTION PROJECT



MARC BERMEJO

RAUL PUENTE

FERRAN TORRES

| | Assigned | Progress |
|---|---|---|
| ▾ Initial Dataset loading | | 100% |
| Implement base dataset class | Marc Bermejo | 100 |
| Implement image augmentation pipelines | Marc Bermejo | 100 |
| Implement BSDS500 dataset class | Marc Bermejo | 100 |
| Dataset preparation and loading ready | Marc Bermejo | ☑ |
| ▾ Generator model | | 100% |
| Implement residual dense block module | Marc Bermejo | 100 |
| Implement residual on residual dense block module | Marc Bermejo | 100 |
| Implement sub-pixel convolution module | Marc Bermejo | 100 |
| Implement full SR generator neural model | Marc Bermejo | 100 |
| ▾ Discriminator model | | 100% |
| Implement relativistic discriminator of ESRGAN model | Marc Bermejo | 100 |
| ▾ Training | | 100% |
| Create initial training script with hard-coded hyperparameters | Marc Bermejo | 100 |
| First model training | Marc Bermejo | ☑ |
| Store and load check-points during training | Marc Bermejo, Raul | 100 |
| ▾ Model criterions and performance metrics | | 100% |
| Impement PSNR metric | Ferran Torres, Marc | 100% |
| Implement structural similarity metric | Ferran Torres, Marc | 100% |
| ▾ Logging | | 100% |
| Implement abstract logging base class | Marc Bermejo, Raul | 100 |

JUNE 2022

PROJECT ADVISOR:
DANI FOJO

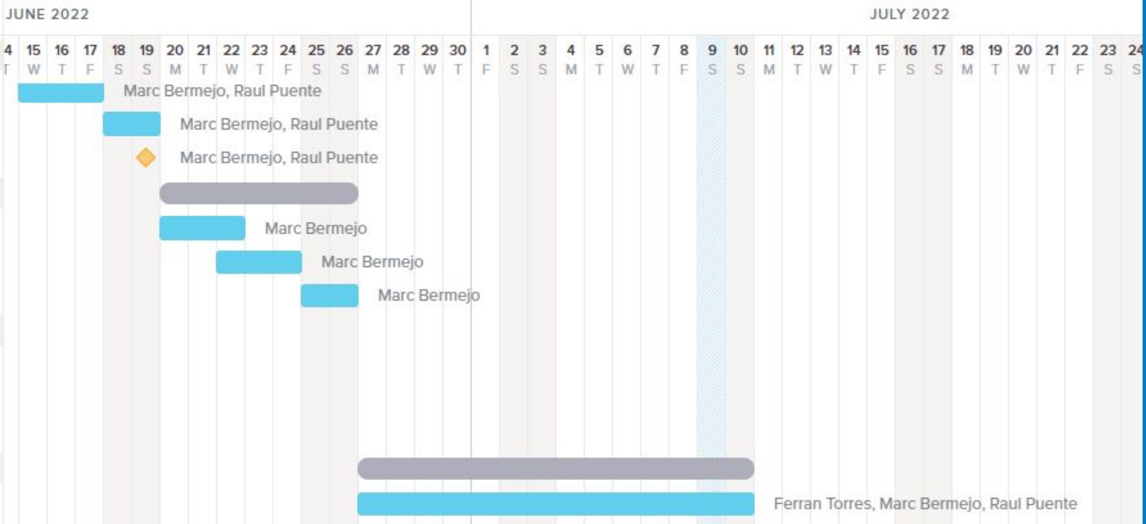


MARC BERMEJO

RAUL PUENTE

FERRAN TORRES

| | Assigned | Progress |
|---|---|---|
| Logging | | 100% |
| Implement abstract logging base class | Marc Bermejo, Raul | 100 |
| Implement TensorBoard logging | Marc Bermejo, Raul | 100 |
| Visualize first metrics of model training | Marc Bermejo, Raul | ☑ |
| Hyper-parameter optimization | | 100% |
| Define hyper-parameter scheduling approach | Marc Bermejo | 100 |
| Implement hyper-parameter scheduler and optimizer | Marc Bermejo | 100 |
| Record performance of different hyper-parameter settings | Marc Bermejo | 100 |
| Implement network interpolation, test and inference | | 0% |
| Create network interpolation Python script | Marc Bermejo | 100 |
| Create generator model test Python script | Marc Bermejo | 100 |
| Create model inference Python script | Marc Bermejo | 100 |
| Final project conclusions | | 100% |
| Set conclusions of model performance with different configurat... | Ferran Torres, Marc | 100% |




PROJECT ADVISOR:
DANI FOJO



MARC BERMEJO

RAUL PUENTE

FERRAN TORRES

# DATASETS CONSIDERED

## BSDS500

- Standard benchmark for edge and contour detection segmentation.
- Consists of 500 images, each with 5 different ground truth segmentations.
- Contains:
  - 200 images for training.
  - 100 images for validation.
  - 200 images for testing.

## DIV2K

- Recommended for SR given the different types of degradations contained in this dataset.
- Different upscaling and downscaling steps applied to obtain those degradations..
- 1000 images. All with 2K resolution.
- Contains:
  - 800 images for training.
  - 100 images for validation.
  - 100 images for testing


PROJECT ADVISOR:
DANI FOJO







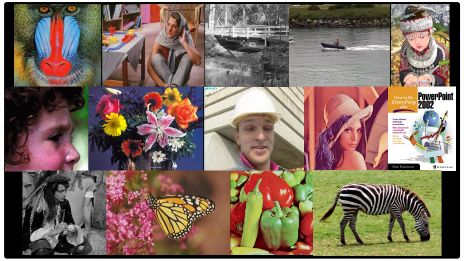
MARC BERMEJO
RAUL PUENTE
FERRAN TORRES

# DATASETS CONSIDERED

## SET5

- Dataset that consists of 5 images (“baby”, “bird”, “butterfly”, “head”, “woman”).
- Commonly used for testing performance of Image Super-Resolution models.

## SET14




- Dataset consisting of 14 images.
- Commonly used for testing performance of Image Super-Resolution models.

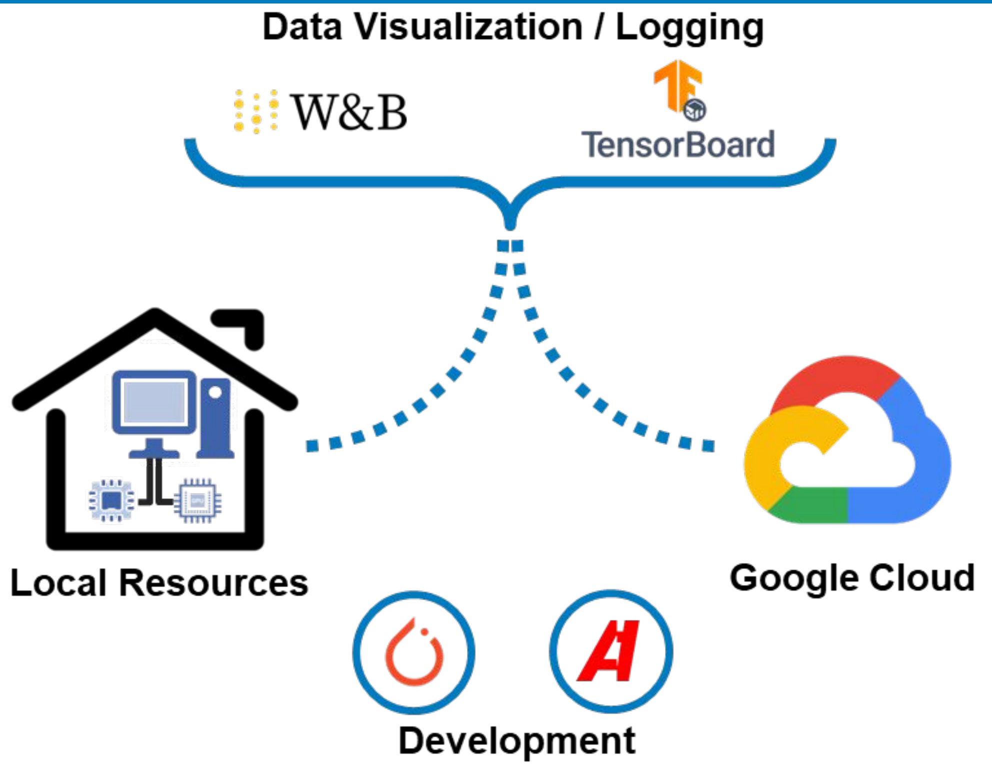
PROJECT ADVISOR:
DANI FOJO



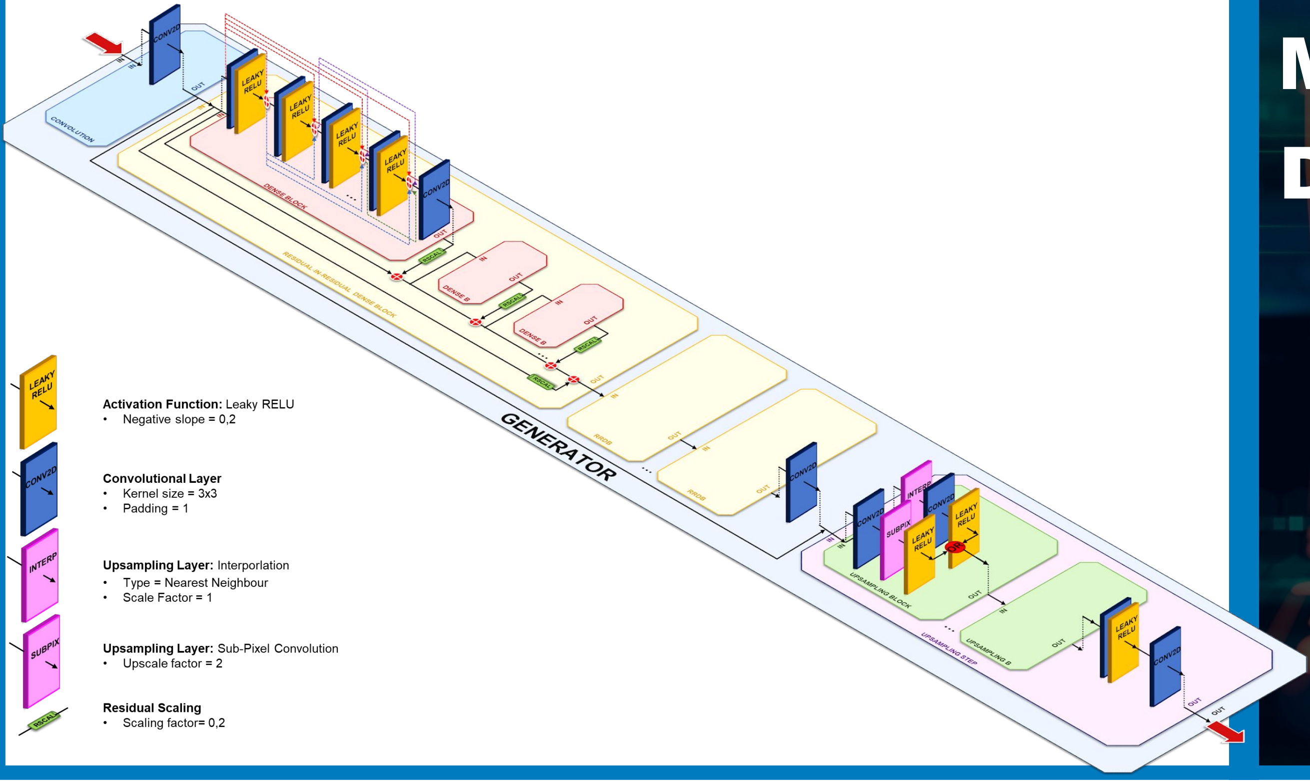


MARC BERMEJO RAUL PUENTE FERRAN TORRES

# ENVIRONMENT USED



# MODEL DEFINITION




**Activation Function:** Leaky RELU
- Negative slope = 0,2

**Convolutional Layer**
- Kernel size = 3x3
- Padding = 1

**Upsampling Layer:** Interporlation
- Type = Nearest Neighbour
- Scale Factor = 1

**Upsampling Layer:** Sub-Pixel Convolution
- Upscale factor = 2

**Residual Scaling**
- Scaling factor= 0,2

PROJECT ADVISOR:
DANI FOJO

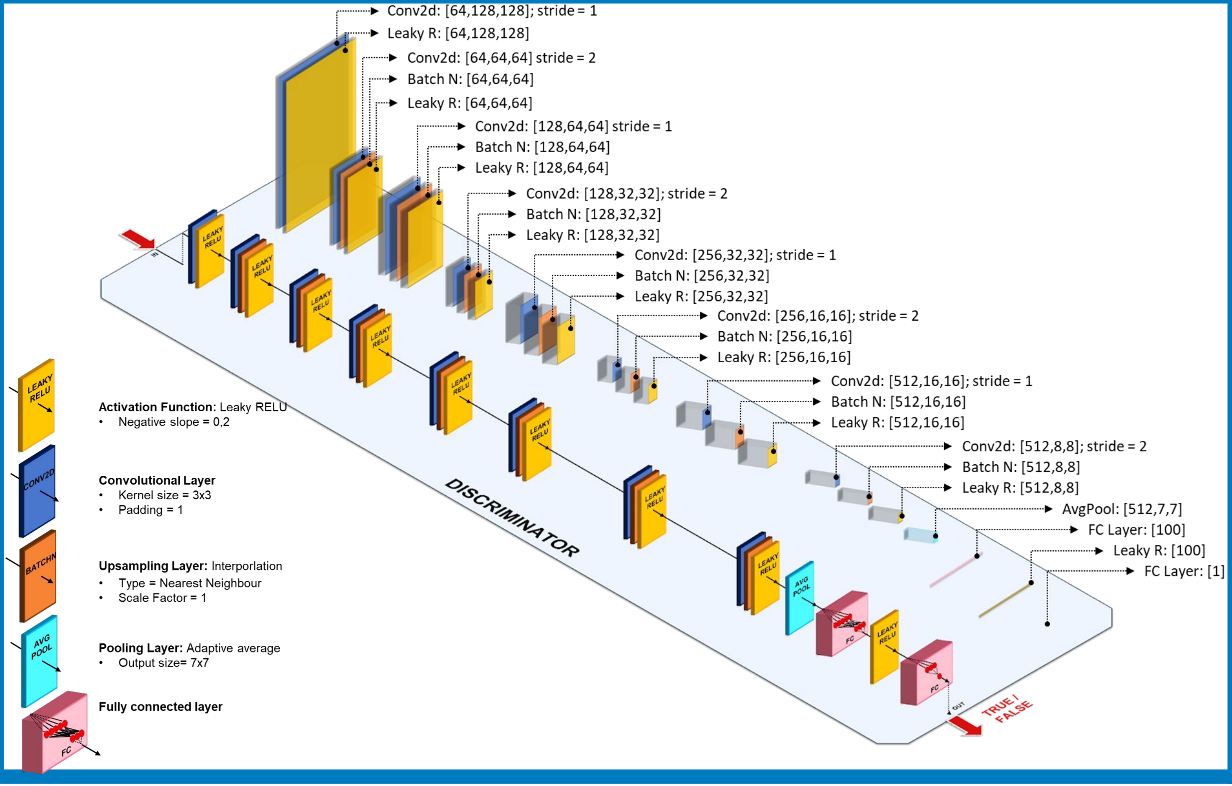



MARC BERMEJO


RAUL PUENTE


FERRAN TORRES

# MODEL DEFINITION



## Network Interpolation:

- Used to remove unpleasant noise generated by the GAN maintaining a Good perceptual quality.

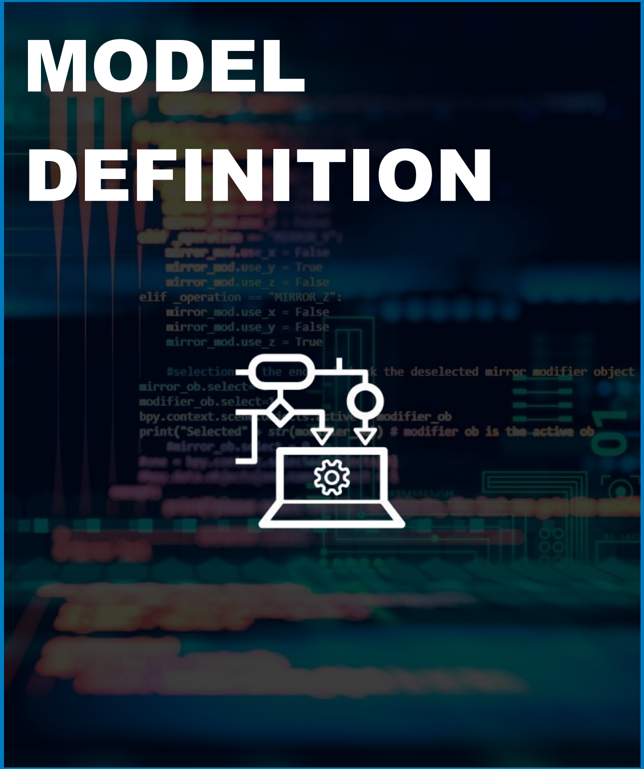
- Process followed:

  1. Train a PSNR-oriented Generator ($G_{\mathrm{PSNR}}$).
  2. Fine tune to obtain the GAN-based network $G_{\mathrm{GAN}}$.
  3. Apply interpolation to all the corresponding paramaters of the 2 networks to reach a final interpolated model.

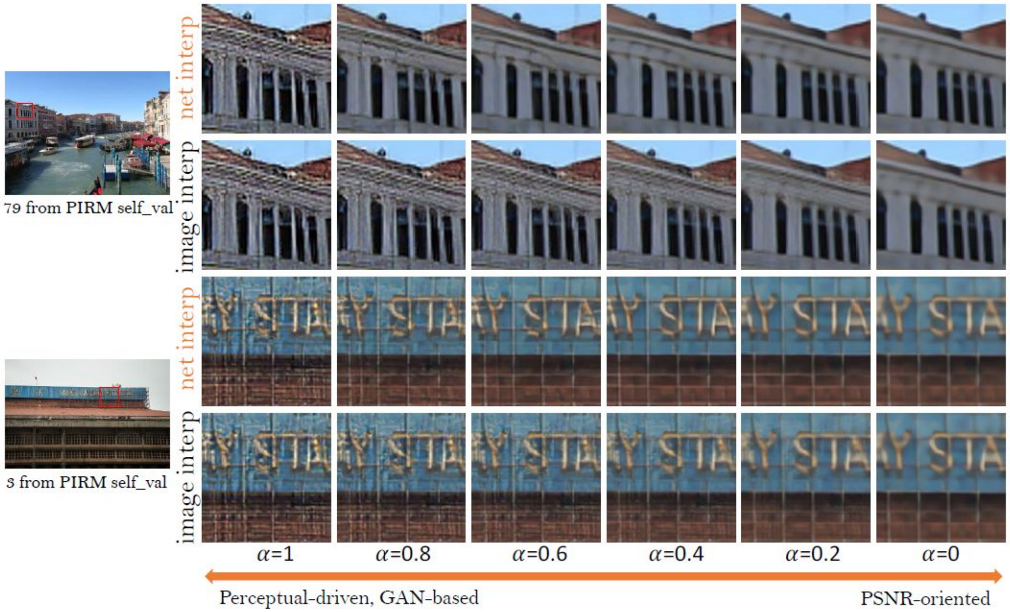
**Formula:**

$$\theta_G^{\mathrm{INTERP}} = (1-\alpha)\ \theta_G^{\mathrm{PSNR}} + \alpha\ \theta_G^{\mathrm{GAN}}$$

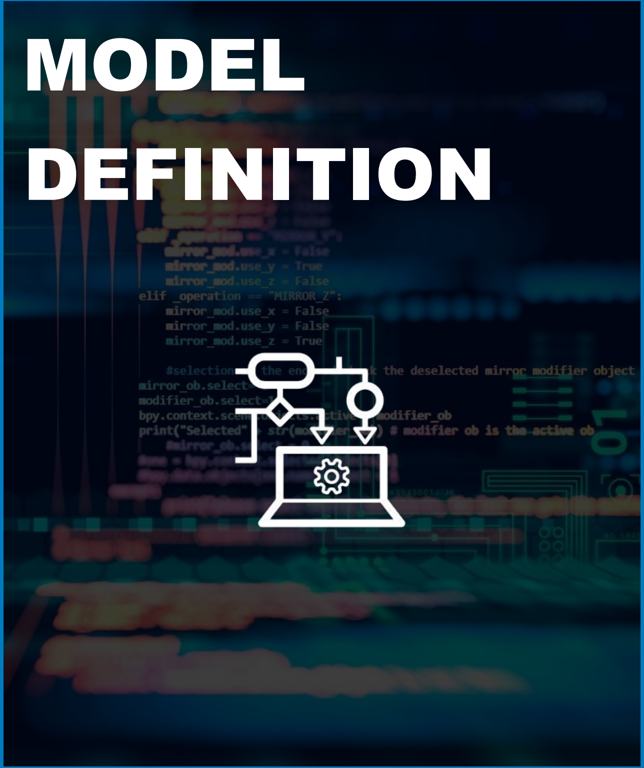
- α= Interpolation parameter



## Network Interpolation:




# MODEL DEFINITION



PROJECT ADVISOR:
DANI FOJO



MARC BERMEJO
RAUL PUENTE
FERRAN TORRES

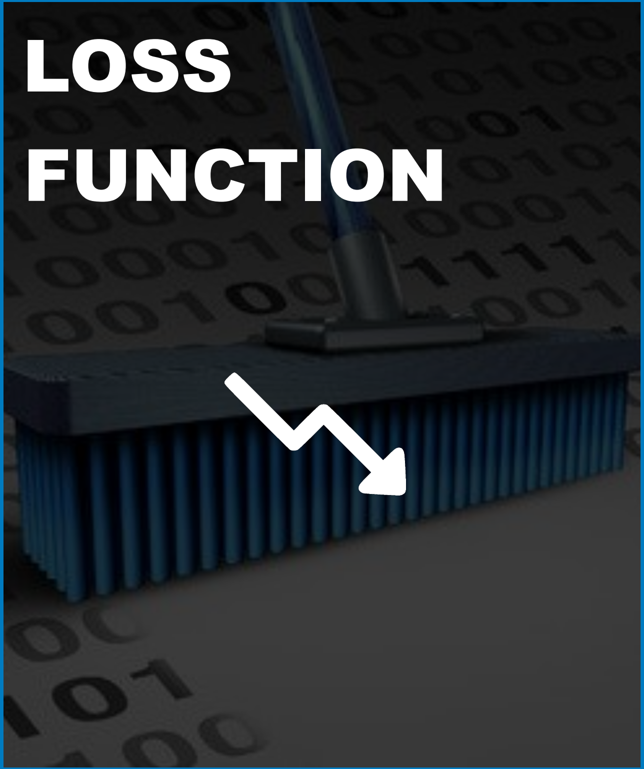
## Content Loss ($L_{content}$):

- Mean error calculated between each pixel from the real image from the generated one.
- Ensures that the activations of higher layers are similar between the Ground Truth and the generated images.
- The project is prepared to work either with L1 (Mean Absolute Error) or L2 (Mean Squared Error) functions. Set to L1 by default.

**Formula:**

$$L1LossFunction = \sum_{i=1}^{n} |y_{true} - y_{predicted}|$$

$$L2LossFunction = \sum_{i=1}^{n} (y_{true} - y_{predicted})^2$$





PROJECT ADVISOR:
DANI FOJO




MARC BERMEJO

RAUL PUENTE

FERRAN TORRES

## Relativistic Adversarial Loss ($L_D^{Ra}$ , $L_G^{Ra}$ ):

- Error that tries to predict if the real image ($x_r$) is relatively more realistic than the generated one ($x_f$):

$D_{Ra}(x_r, x_f) = \sigma(C($ 

$) - \mathbb{E}[C($ 

$)]) \to 1$ More realistic than fake data?

$D_{Ra}(x_f, x_r) = \sigma(C($ 

$) - \mathbb{E}[C($ 

$)]) \to 0$ Less realistic than real data?

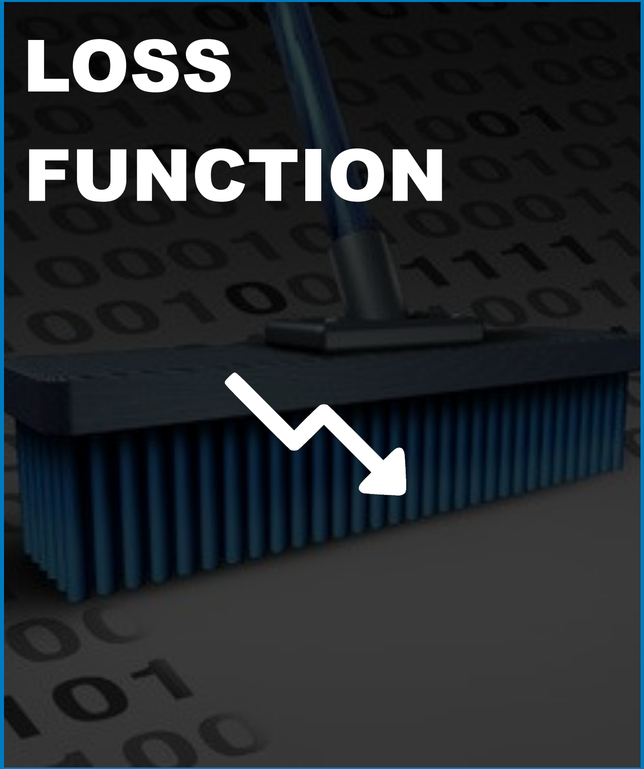
  - σ = Sigmoid
  - C(x) = Non-Transformed discriminator output
  - $E_{xf}[\cdot]$ = Average for all fake data in a minibatch

**Formula:**

$$L_D^{Ra} = -E_{x_r}[log(D_{Ra}(x_r, x_f))] - E_{x_f}[log(1 - D_{Ra}(x_f, x_r))]$$

$$L_G^{Ra} = -E_{x_r}[log(1 - D_{Ra}(x_r, x_f))] - E_{x_f}[log(D_{Ra}(x_f, x_r))]$$

  - $x_f$=G($x_i$)
  - $x_f$=LR image

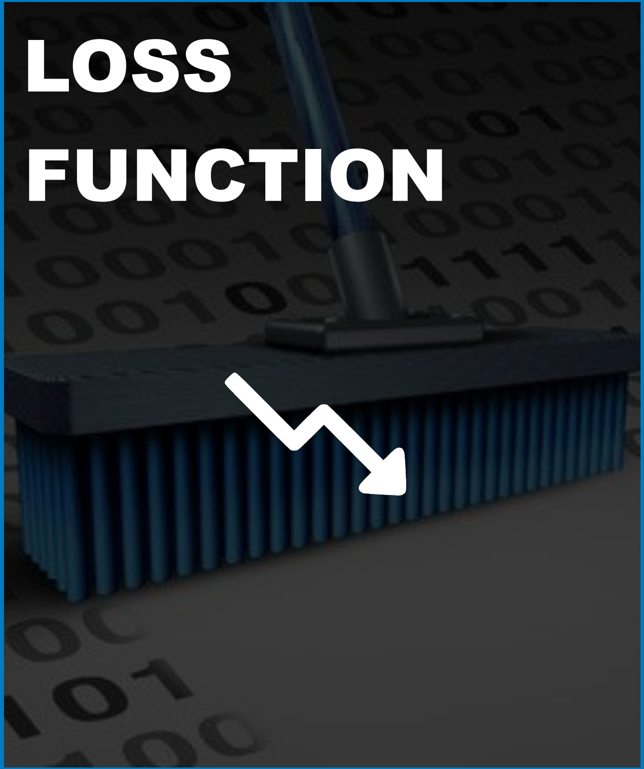


## Perceptual Loss ($L_{percept}$):

- Also known as VGG Loss because this error estimation between both generated and real images is done from a pretrained network (VGG19 in this case).
- Evaluated before last VGG activation layer → Improvement compared to SRGAN.

**Formula:**

$$l_{VGG/i,j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} \left( \phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y} \right)^2$$



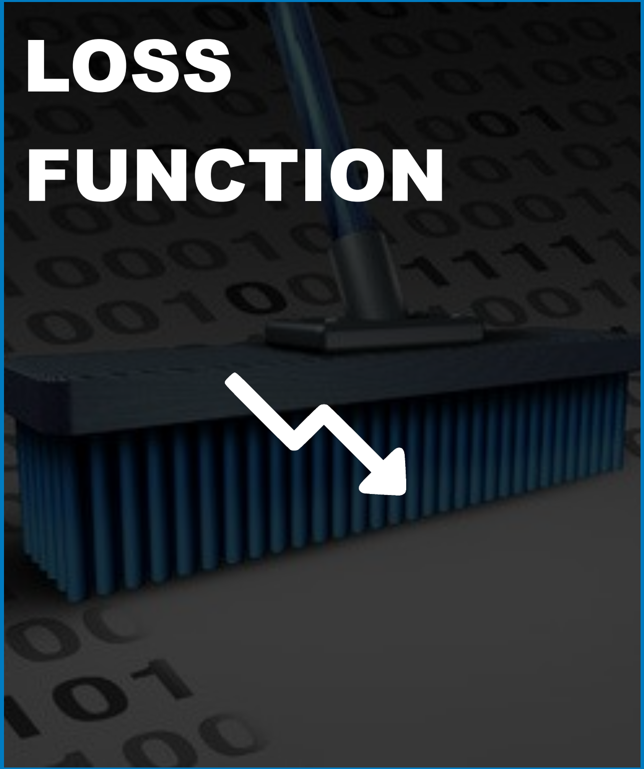
# Total Loss ($L_G$):

- Loss function considered during the training of the complete GAN model (Generator + Discriminator)

**Formula:**

$$L_G = L_{percep} + \lambda L_G^{Ra} + \eta L_{content}$$

- **λ, η**= weights assigned to each Loss function to balance terms



1 Image retrieval

A HR & LR

HR & LR Load

B HR

Center Crop

( × Scale Factor)

2 LR image generation

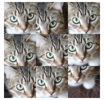
"Nearest" Interpolation

A Train

Random Crop

B Validate

Center Crop



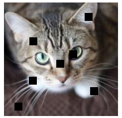
3 User-defined transforms

## 3 User-defined transforms

### 4 Warm-Up (Pre-training)

**Hard**

- **Compression** P=0,25
- **Coarse Dropout** P=0,25



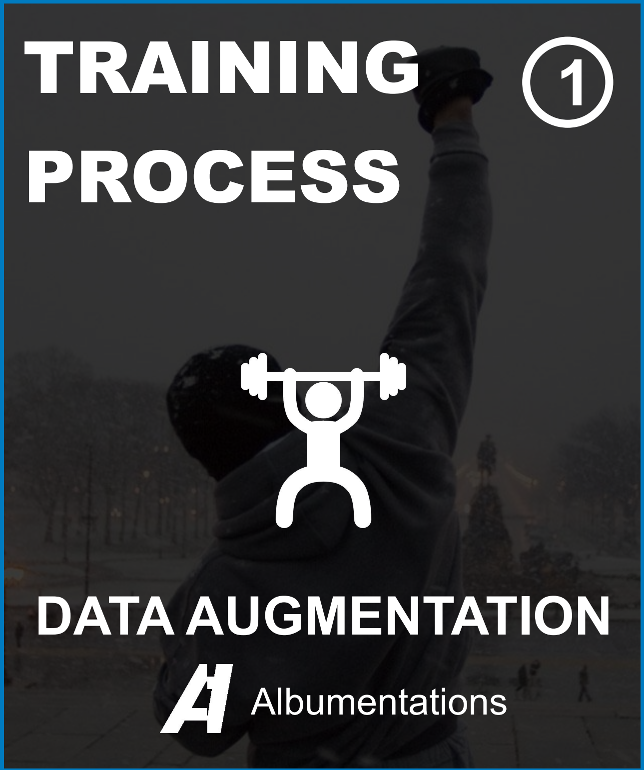
**Spatial**

- **Flip** P=0,25 **OR** **Transpose** P=0,75

### 5 Training (Complete GAN)

**Spatial**

- **Flip** P=0,25 **OR** **Transpose** P=0,75






MARC BERMEJO
RAUL PUENTE
FERRAN TORRES

## Image data Augmentation

Training: Paired Random Crop
Validation: Paired Center Crop

## Loss Function

Content Loss

## Training process

Generator

## Optimization

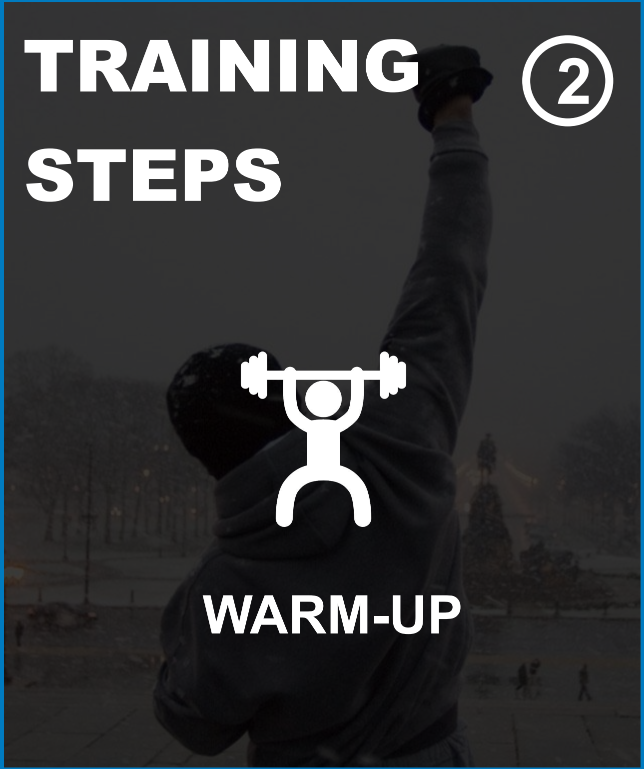
Adam Optimizer with Learning rate = 2e-4

## Scheduler

Steps applied = 175000 & $\gamma$=0,5

## Metrics Logging

Pre-training: Content Loss
Validation: Content Loss / Perceptual Loss / SSIM / PSNR



1

## Image data Augmentation

Random Crop for Training / Random Center Crop for Validation
Spatial Transformation

2

## Loss Function

Content Loss, Perceptual Loss / Generator Adversarial Loss

3

## Training process

1- Generator training for every mini-batch freezing the Discriminator.
2- Discriminator training for same mini-batch freezing the Generator.

4

## Optimization

Adam Optimizer with Learning rate = 1e-4

5

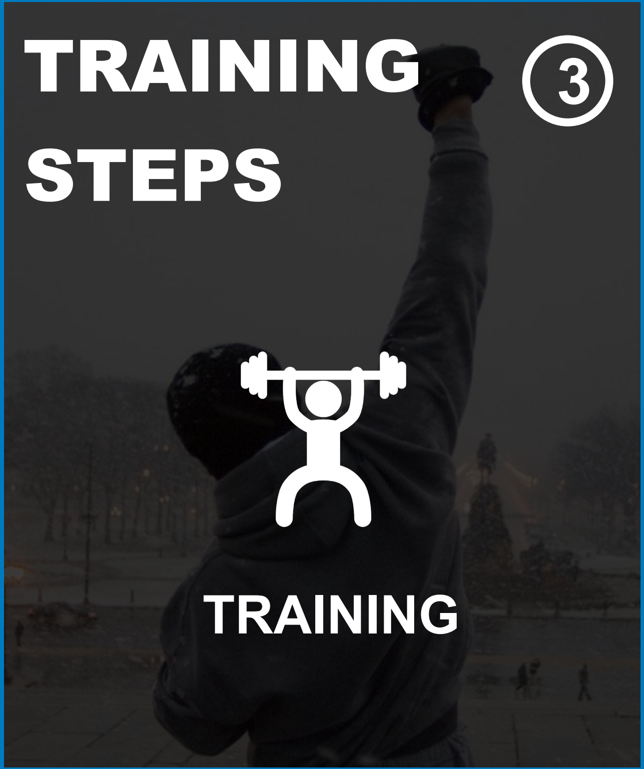
## Scheduler

Steps applied = [50000, 100000, 200000, 300000] & γ=0,5

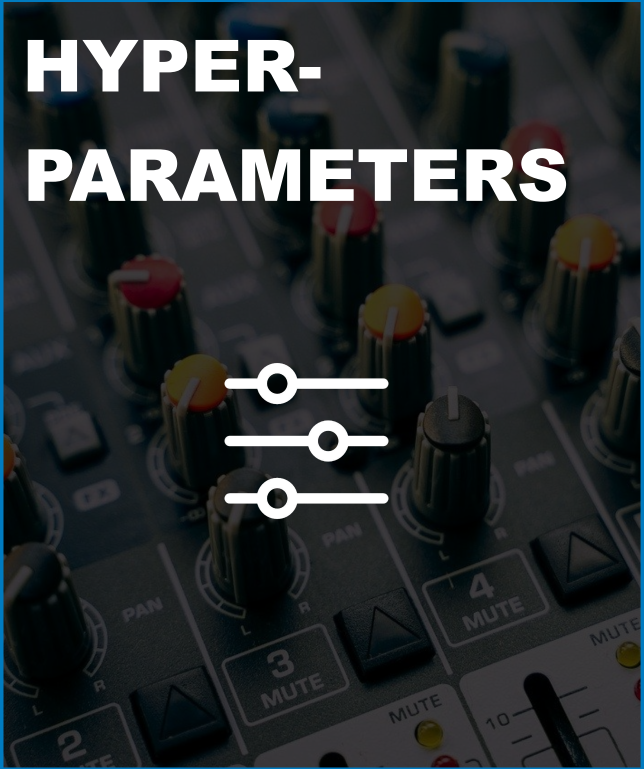
6

## Metrics Logging

Content Loss, Perceptual Loss / Generator Adversarial Loss / Total Loss/
Discriminator Adversarial Lossss / Perceptual Loss/ PSNR / SSIM



| Hyperparameters | Description |
|---|---|
| scale_factor | Scale factor relation between the low resolution and the high resolution images. |
| batch_size | Data loader configured mini-batch size. |
| pretraining/num_epoch | Number of epoch needed to complete the pre-training (PSNR-driven) step. |
| pretraining/cr_patch_size | High-resolution image crop size. Needs to be a tuple of (H, W). If set to None, any crop transform will be applied. |
| pretraining/lr | Configured learning rate for the pre-training step optimizer. Adam optimizer will be used for this step. |
| pretraining/sched_step | Learning rate scheduler decay rate for the pre-training step. |
| pretraining/sched_gamma | Multiplicative factor of the learning rate scheduler decay for the pre-training step. |
| pretraining/train_datasets | Dataset(s) used during training of the pre-training step. Must be one of 'div2k', 'bsds500'. |
| pretraining/val_datasets | Dataset(s) used during validation of the pre-training step. Must be one of 'div2k', 'bsds500'. |
| training/num_epoch | Number of epoch needed to complete the training step. |
| training/cr_patch_size | Number of epoch needed to complete the pre-training (GAN-driven) step. |
| training/g_lr | Configured generator's learning rate for the training step optimizer. Adam optimizer will be used for this step. |
| training/d_lr | Configured discriminator's learning rate for the training step optimizer. Adam optimizer will be used for this step. |
| training/g_sched_steps | List of mini-batch indices learning rate decay of the generator's training scheduler. |
| training/g_sched_gamma | Multiplicative factor of the generator's learning rate scheduler decay for the training step. |
| training/d_sched_steps | List of mini-batch indices learning rate decay of the discriminator's training scheduler. |
| training/train_datasets | Dataset(s) used during training of the training step. Must be one of 'div2k', 'bsds500'. |
| training/val_datasets | Dataset(s) used during validation of the training step. Must be one of 'div2k', 'bsds500'. |
| generator/num_basic_blocks | Number of basic (a.k.a residual-on-residual dense blocks) of the generator network. |
| training/d_sched_gamma | Multiplicative factor of the discriminator's learning rate scheduler decay for the training step. |




PROJECT ADVISOR:
DANI FOJO



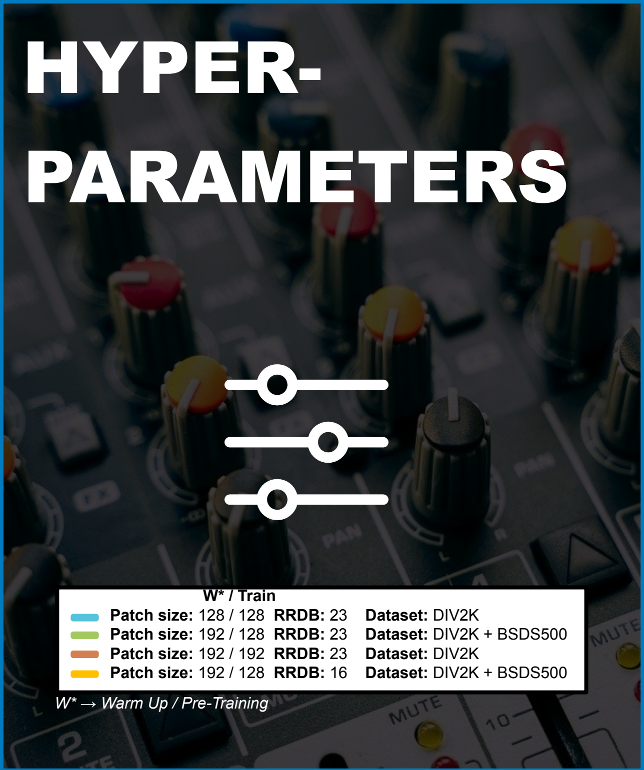

MARC BERMEJO


RAUL PUENTE


FERRAN TORRES

## Hyperparameters

| Hyperparameters | | | | |
|---|---|---|---|---|
| scale_factor | 4 | 4 | 4 | 4 |
| batch_size | 16 | 16 | 16 | 16 |
| pretraining/num_epoch | 8000 | 8000 | 8000 | 8000 |
| pretraining/cr_patch_size | [192,192] | [128, 128] | [192,192] | [192,192] |
| pretraining/lr | 0,0002 | 0,0002 | 0,0002 | 0,0002 |
| pretraining/sched_step | 175000 | 175000 | 175000 | 175000 |
| pretraining/sched_gamma | 0.5 | 0.5 | 0.5 | 0.5 |
| pretraining/train_datasets | ["div2k"] | ["div2k"] | ["bsds500", "div2k"] | ["bsds500", "div2k"] |
| pretraining/val_datasets | ["div2k"] | ["div2k"] | ["div2k"] | ["div2k"] |
| training/num_epoch | 6000 | 6000 | 6000 | 6000 |
| training/cr_patch_size | [192,192] | [128, 128] | [128, 128] | [128, 128] |
| training/g_lr | 0,0001 | 0,0001 | 0,0001 | 0,0001 |
| training/d_lr | 0,0001 | 0,0001 | 0,0001 | 0,0001 |
| training/g_sched_steps | [50000, 100000, 175000, 250000] | [50000, 100000, 175000, 250000] | [50000, 100000, 175000, 250000] | [50000, 100000, 175000, 250000] |
| training/g_sched_gamma | 0.5 | 0.5 | 0.5 | 0.5 |
| training/d_sched_steps | [50000, 100000, 175000, 250000] | [50000, 100000, 175000, 250000] | [50000, 100000, 175000, 250000] | [50000, 100000, 175000, 250000] |
| training/train_datasets | ["div2k"] | ["div2k"] | ["bsds500", "div2k"] | ["bsds500", "div2k"] |
| training/val_datasets | ["div2k"] | ["div2k"] | ["div2k"] | ["div2k"] |
| generator/num_basic_blocks | 23 | 23 | 16 | 23 |
| training/d_sched_gamma | 0.5 | 0.5 | 0.5 | 0.5 |




**PROJECT ADVISOR:**
**DANI FOJO**





**MARC BERMEJO** **RAUL PUENTE** **FERRAN TORRES**

## Structural Similarity Index(SSIM):

- Given 2 images, SSIM is an index with values in the range (-1,1), which estimates the level of similarity between those two images.
  - **+1** = very similar or the same.
  - **-1** = very different
- It combines different comparison functions:
  - Luminance $l(\mathbf{x},\mathbf{y})$.
  - Contrast $c(\mathbf{x},\mathbf{y})$
  - Structure $s(\mathbf{x},\mathbf{y})$

**Formula:**

$$\mathrm{SSIM}(\mathbf{x},\mathbf{y}) = [l(\mathbf{x},\mathbf{y})]^{\alpha} \cdot [c(\mathbf{x},\mathbf{y})]^{\beta} \cdot [s(\mathbf{x},\mathbf{y})]^{\gamma}$$

- α,β,γ= weights assigned to each feature

## Peak Signal-to-Noise (PSNR):

- Ratio between maximum possible value (power) of a signal and power of distorting noise that affects the quality of its representation.

- Metric used to compare different image enhancement algorithms systematically to evaluate which produces better results using the same dataset.

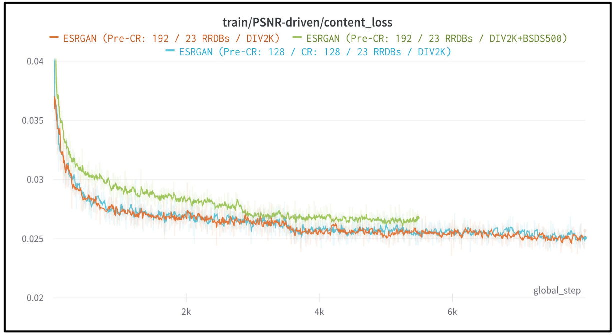
**Formula:**

$$PSNR = 20\log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right)$$

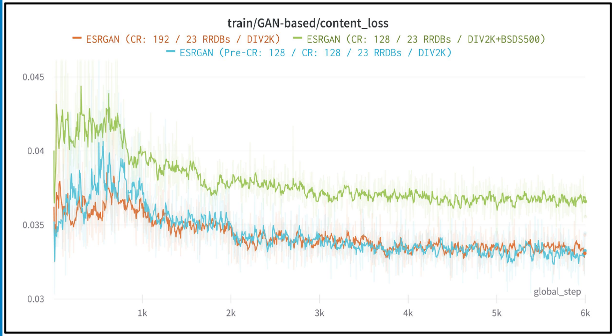
- **MSE =** L2 loss
- **$MAX_f$** = Maximum existing signal value in our original "known to be good" image.

# RESULTS FOR DISCUSSION

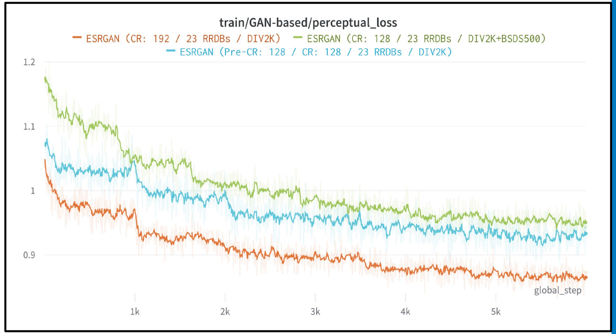
## CONTENT LOSS WARM-UP




## CONTENT LOSS TRAINING




## PERCEPTUAL LOSS TRAINING




PROJECT ADVISOR:
DANI FOJO

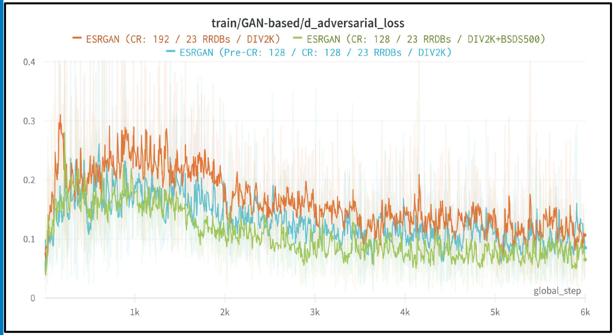


MARC BERMEJO

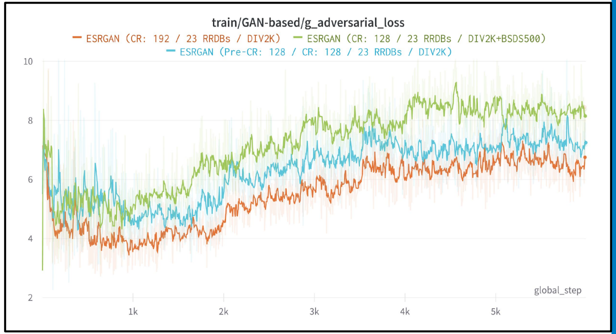
RAUL PUENTE

FERRAN TORRES

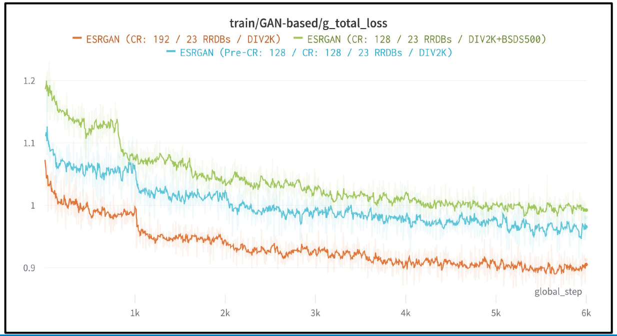
## DISCRIMINATOR ADVERSARIAL LOSS TRAINING




## GENERATOR ADVERSARIAL LOSS TRAINING



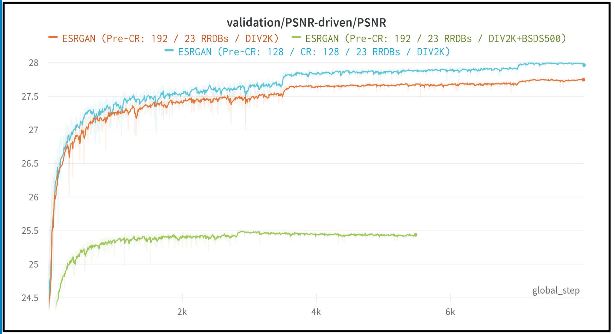

## TOTAL LOSS GAN TRAINING



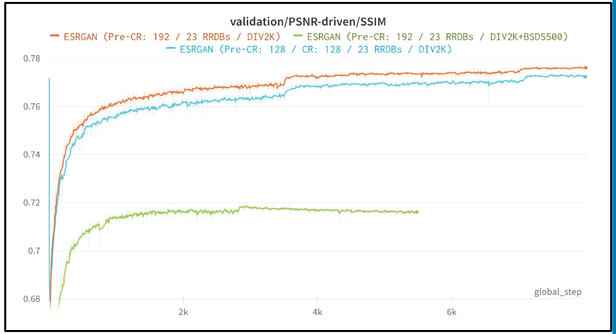

# RESULTS FOR DISCUSSION

# RESULTS FOR DISCUSSION

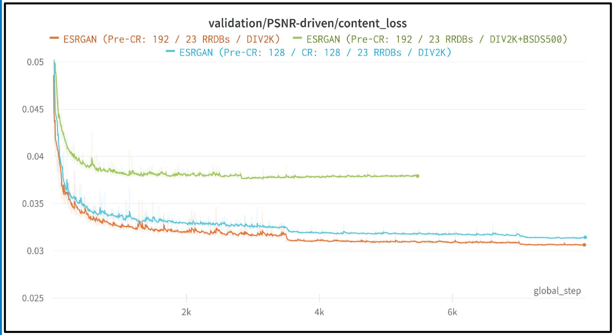
## PSNR WARM-UP VALIDATION



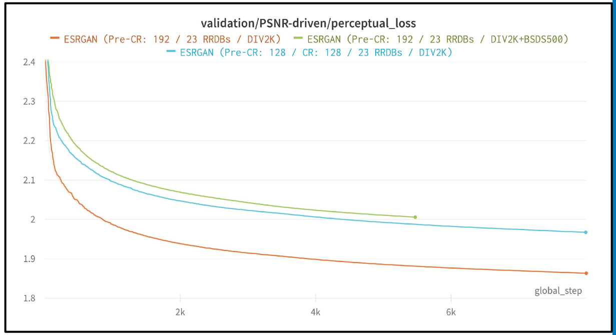

## SSIM WARM-UP VALIDATION




## CONTENT LOSS WARM-UP VALIDATION




## PERCEPTUAL LOSS WARM-UP VALIDATION





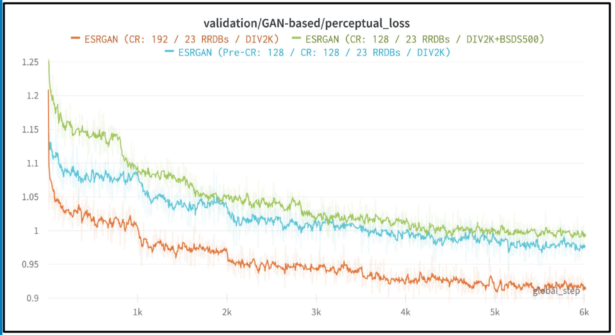
PROJECT ADVISOR:
DANI FOJO

MARC BERMEJO

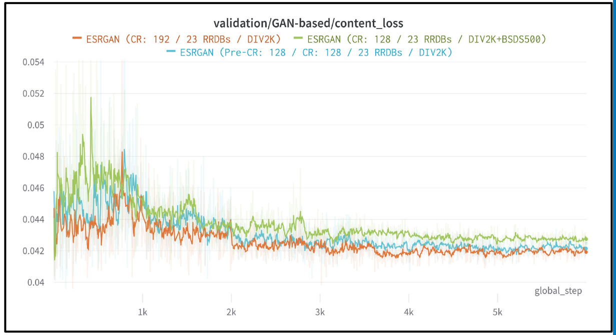
RAUL PUENTE

FERRAN TORRES

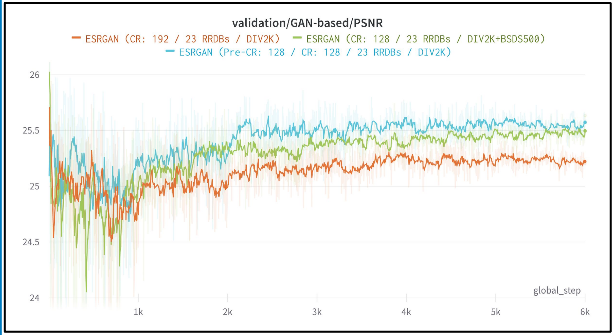
## PERCEPTUAL LOSS GAN VALIDATION



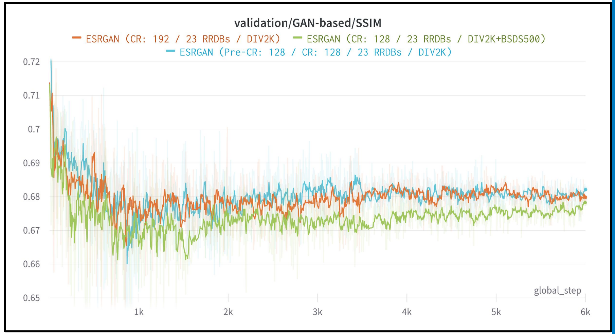

## CONTENT LOSS GAN VALIDATION




## PSNR GAN VALIDATION




## SSIM GAN VALIDATION



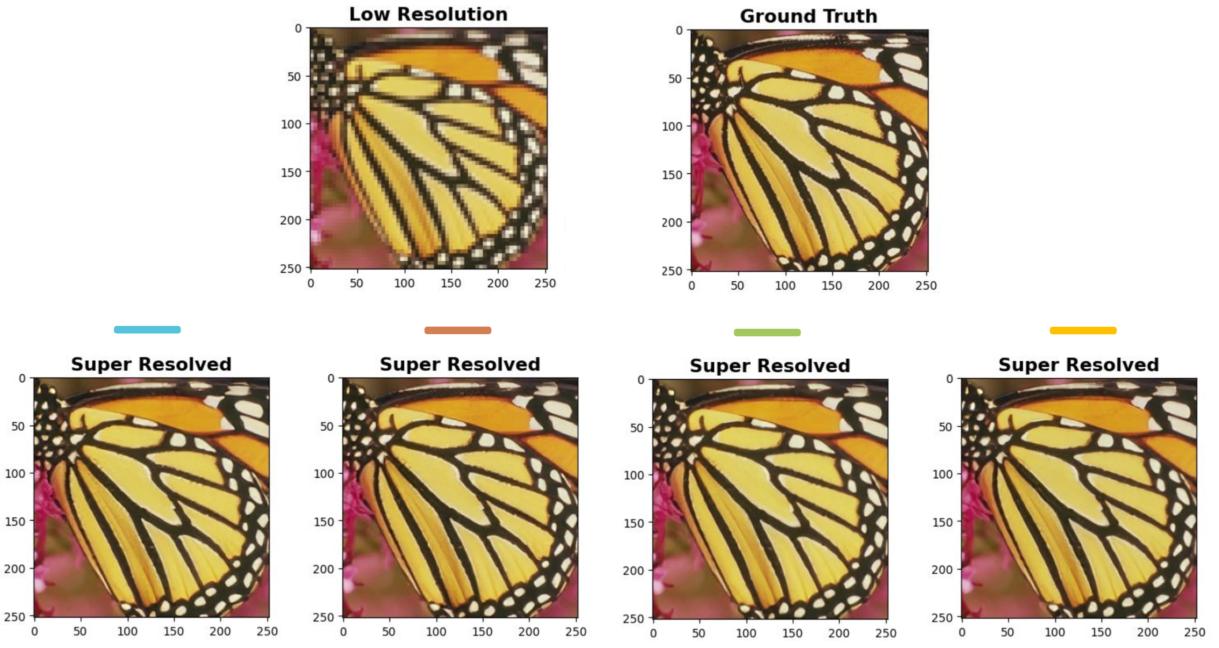


PROJECT ADVISOR:
DANI FOJO





MARC BERMEJO
RAUL PUENTE
FERRAN TORRES




PSNR: 25.97db / SSIM: 0.87 | PSNR: 26.41db / SSIM: 0.88 | PSNR: 26.11db / SSIM: 0.88 | PSNR: 25.96db / SSIM: 0.88

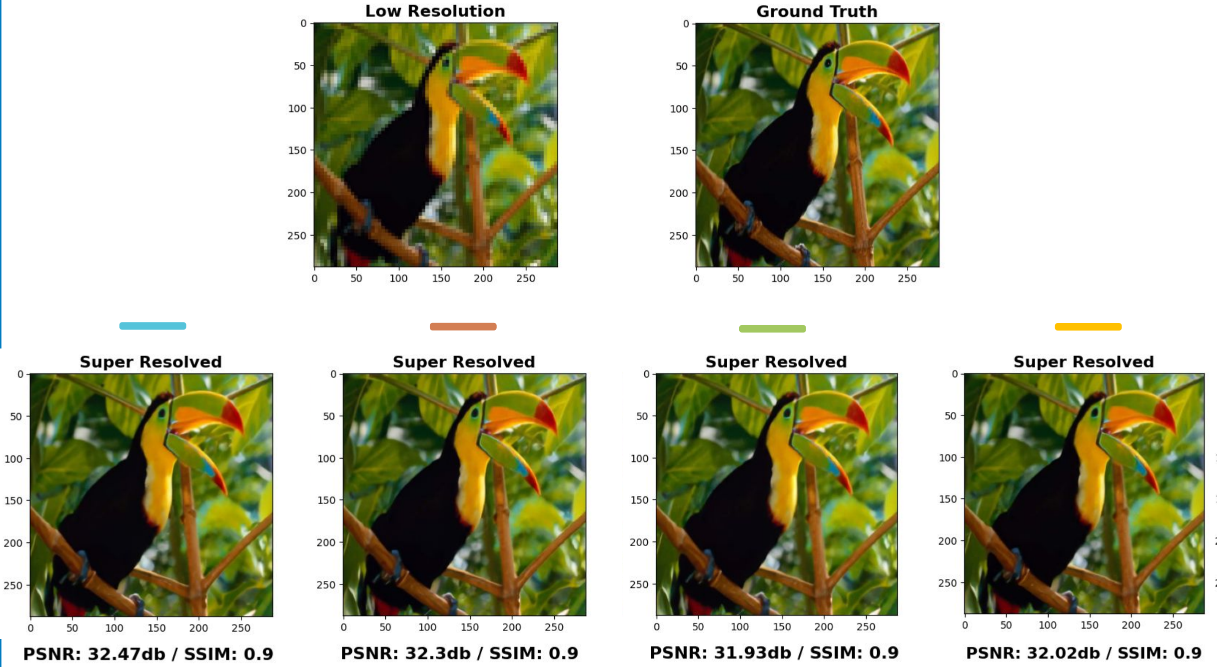
# RESULTS FOR DISCUSSION

**W* / Train**

| | | |
|---|---|---|
| **Patch size:** 128 / 128 | **RRDB:** 23 | **Dataset:** DIV2K |
| **Patch size:** 192 / 128 | **RRDB:** 23 | **Dataset:** DIV2K + BSDS500 |
| **Patch size:** 192 / 192 | **RRDB:** 23 | **Dataset:** DIV2K |
| **Patch size:** 192 / 128 | **RRDB:** 16 | **Dataset:** DIV2K + BSDS500 |

*W* → Warm Up / Pre-Training*

# RESULTS FOR DISCUSSION



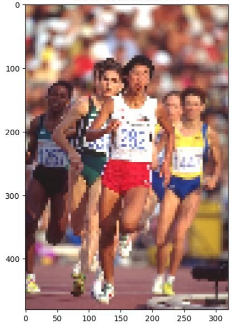

**W* / Train**

| | | | |
|---|---|---|---|
| (blue) | **Patch size:** 128 / 128 | **RRDB:** 23 | **Dataset:** DIV2K |
| (green) | **Patch size:** 192 / 128 | **RRDB:** 23 | **Dataset:** DIV2K + BSDS500 |
| (orange) | **Patch size:** 192 / 192 | **RRDB:** 23 | **Dataset:** DIV2K |
| (yellow) | **Patch size:** 192 / 128 | **RRDB:** 16 | **Dataset:** DIV2K + BSDS500 |

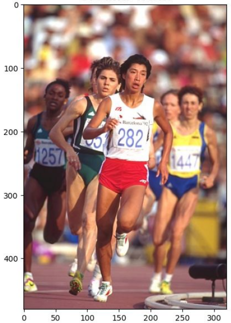
*W* → Warm Up / Pre-Training*

# RESULTS FOR DISCUSSION

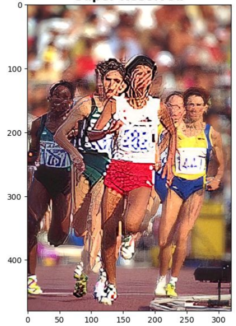
Low Resolution



Ground Truth

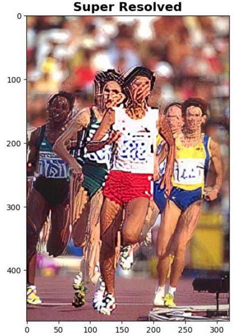


Super Resolved



PSNR: 16.18db / SSIM: 0.44

Super Resolved

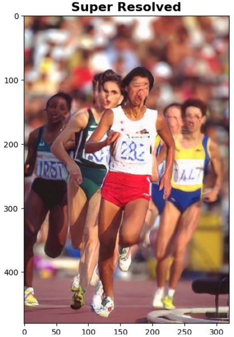


PSNR: 16.44db / SSIM: 0.45

Super Resolved

PSNR: 20.37db / SSIM: 0.67

Super Resolved



PSNR: 20.12db / SSIM: 0.66

**W* / Train**

- **Patch size:** 128 / 128 **RRDB:** 23 **Dataset:** DIV2K
- **Patch size:** 192 / 128 **RRDB:** 23 **Dataset:** DIV2K + BSDS500
- **Patch size:** 192 / 192 **RRDB:** 23 **Dataset:** DIV2K
- **Patch size:** 192 / 128 **RRDB:** 16 **Dataset:** DIV2K + BSDS500

*W* → Warm Up / Pre-Training*

PROJECT ADVISOR:
DANI FOJO



MARC BERMEJO
RAUL PUENTE
FERRAN TORRES

# RESULTS FOR DISCUSSION

| Model | Datasets | SET5 | | SET14 | |
|---|---|---|---|---|---|
| | | PSNR | SSIM | PSNR | SSIM |
| Pre-CR: 128 / CR: 128 / 23 RRDBs | DIV2K | 32,47dB | 0,9335 | 31,01dB | 0,9749 |
| Pre-CR: 192 / CR: 192 / 23 RRDBs | DIV2K | 32,30dB | 0,9329 | 31,20dB | 0,9746 |
| Pre-CR: 192 / CR: 128 / 23 RRDBs | DIV2K+BSDS500 | 31,93dB | 0,9266 | 30,57dB | 0,9717 |
| Pre-CR: 192 / CR: 128 / 16 RRDBs | DIV2K+BSDS500 | 32,01dB | 0,9274 | 30,75dB | 0,9730 |
| ESRGAN (Original) - Benchmark | DIV2K | 32,73dB | 0,9011 | 28,99dB | 0,7917 |

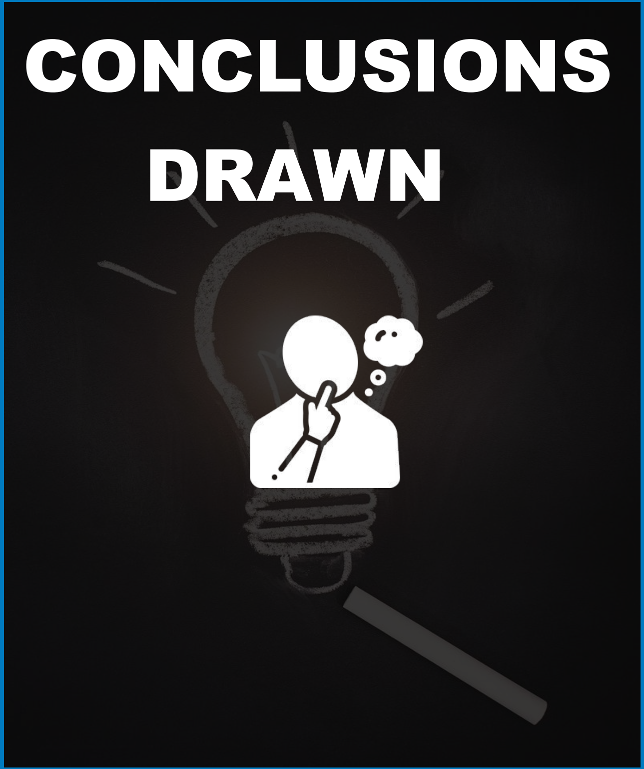
PROJECT ADVISOR:
DANI FOJO

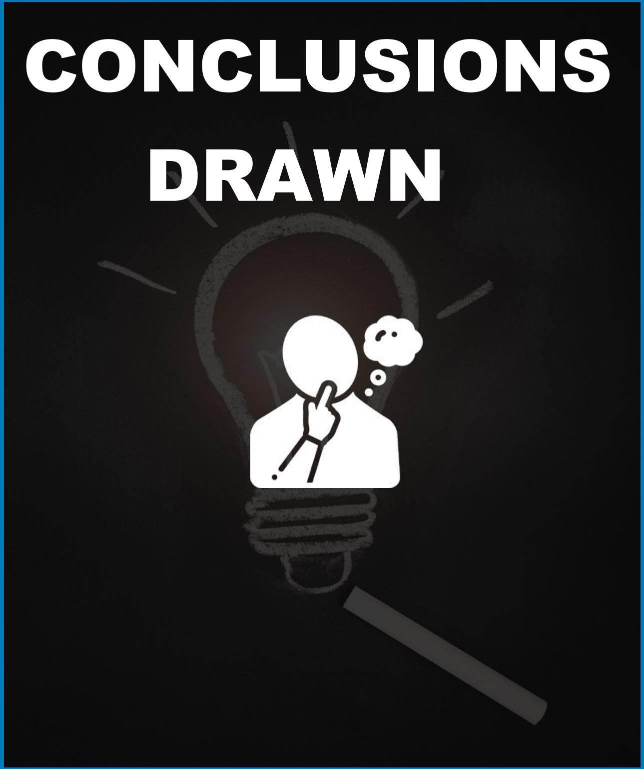
MARC BERMEJO RAUL PUENTE FERRAN TORRES

Training balance between Generator & Discriminator is key.

Network Interpolation reduces noise generated during GAN training. Good perceptual performance is kept.

Deeper models improve recovered textures and reduce unpleasant noise.






- Broader data helps the generator to generalize but reduces overall performance.
- To achieve Good performance for a specific scene type, model needs training with similar scenes firstly.
- Datasets with sharper edges (BSDS500) help improving performance with les information and reducing artifacts.
- Datasets with sharper edges (BSDS500) have lower overall performance for easier images.

# THANK YOU!