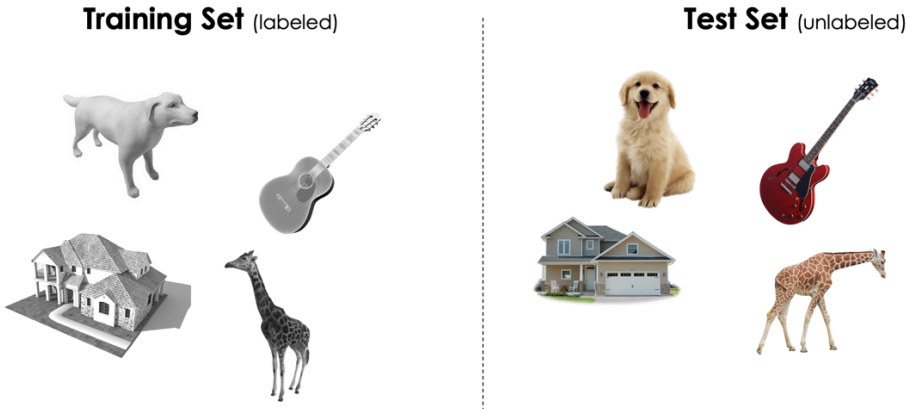
Domain Disentanglement through Natural Language

TA: Silvia Bucci, Leonardo Iurada
silvia.bucci@polito.it, leonardo.iurada@polito.it

Problem

One of the biggest limitations of deep models is their inability to work consistently well with any data met a test time. The training and the test set could be characterized by different data distributions (illumination conditions, visual style, background, etc.) leading to an inevitable decrease in the model performances. This is known as the problem of **domain shift** and it's one of the most investigated topics in the computer vision community. It can be formalized with the setting of **Domain Adaptation** where there is a **labeled Source** domain (Training Set) and an **unlabeled Target** domain (Test Set), the goal is to train a model that works well on the target distribution.



In this project, we want to tackle Domain Adaptation through **Domain Disentanglement**. We start from the assumption that every domain is characterized by features **domain-specific** and features **category-specific**, the goal is to disentangle that information to then use only the latter ones to train the object classifier. We will also investigate how a **textual description of the visual domains** can help in this process. Indeed, the use of natural language as a second modality together with the images could help to obtain a more effective disentanglement.

Before starting

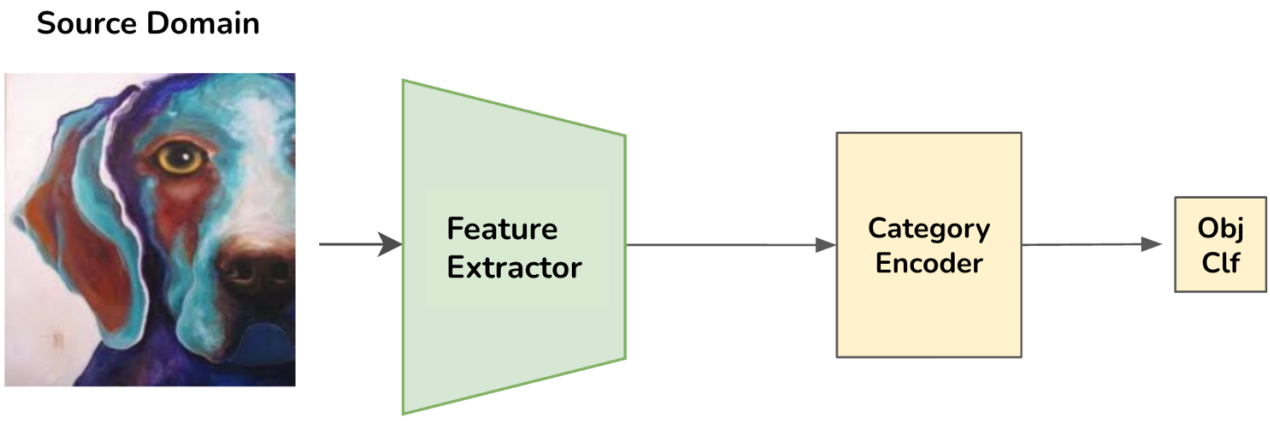
1. Read **[1]** to become familiar with the problem of **domain shift** and with the **Domain Adaptation** setting.
2. Read **[2]** to become familiar with the **domain disentanglement** technique.
3. Explore **[3]** to learn about **CLIP**, the starting model that you should use to include the natural language in the training process.

Mandatory

Starting from the GitHub repository **[4]** provided, carefully follow all the instructions in the *README.md* file to set the **Dataset [5]** and the **Environment**. Then you can start running and editing the code.

1. Reproduce the Baseline

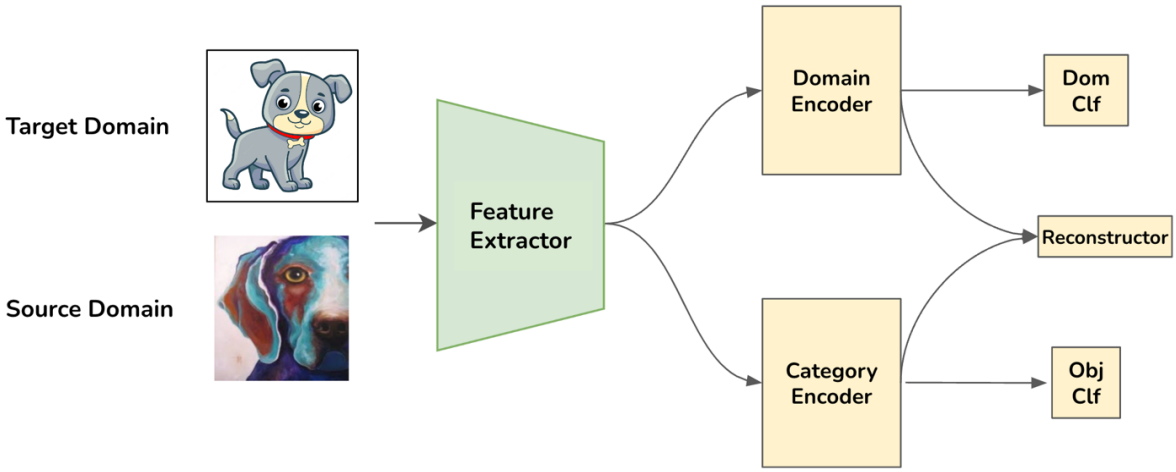
Following the instruction in the README.md file, you should be able to reproduce the **Baseline**. The experiment consists in training the **Feature Extractor** (ResNet-18), the **Category Encoder** (made by some fully connected layers), and the **Object Classifier** (made by one fully connected layer for category prediction) minimizing the Cross-Entropy Loss computed on the **Source Domain** (Art Painting).



Then you should simply test the model on the **Target Domains** (Cartoon, Sketch, Photo), separately, to reproduce the numbers in the table reported in the README.md file.

2. Domain Disentanglement

To improve the performances of the model on the Target Domains you have to implement a domain disentanglement technique to get **category-specific** features for the object classifier. You should include in the architecture a **Domain Encoder** (analogous to the Category Encoder), a **Domain Classifier** (made by one fully connected layer for domain prediction), and a **Reconstructor**.



In this experiment, we use also the **Target Domain** but it is provided without labeled so it cannot be directly used to train the object classifier. The training flow should follow [2] excluding the Mutual Information Minimization and the Gram Matrix loss contributions. To obtain the disentanglement you should consider five contributions in the final loss function:

- the minimization of the **Cross-Entropy Loss** for the object classifier.
- the minimization of the **Cross-Entropy Loss** for the domain classifier.
- the maximization of the **Entropy Loss** for the domain classifier considering the features extracted from the Category encoder.
- the maximization of the **Entropy Loss** for the object classifier considering the features extracted from the Domain Classifier.
- a **Reconstruction loss** to not miss information during the disentanglement process.

With this domain disentanglement technique, you should see an improvement in the performances of the Target Domains with respect to the baseline.

3. Label PACS dataset describing the visual domain attributes

IMPORTANT:
Before starting this step contact Silvia or Leonardo for the assignation of the images.

The task is to describe the appearance and the style of each image. The objects contained in the images are irrelevant to the description. You have to describe each image following these guides:

Level of details → If the image is rich in detail or it is a raw representation (e.g., high/mid/low-level)
Edges → Description of the contours of the objects (e.g., definite/precise/neat strokes, definite/precise/neat brush strokes)
Color saturation → The intensity and brilliance of colors in the image (e.g., high/mid/low, vivid colors, light reflections)
Color shades → If there are shades of colors in the image (e.g., no/yes, colorful/grayscale, etc.)
Background → Description of the background (e.g., monochrome/white/colorful etc.)
Single instance → If the image is composed by a single instance or multiple instances of the same object (e.g., yes/no, how many)
Text → If there is text in the picture (e.g., yes/no, dense/sparse text)
Texture → If there is a visual pattern that is repeated over the whole picture (e.g. yes/no, type of texture)
Perspective → If the three-dimensional proportions of the object parts are realistic (e.g. yes/no, unrealistic)

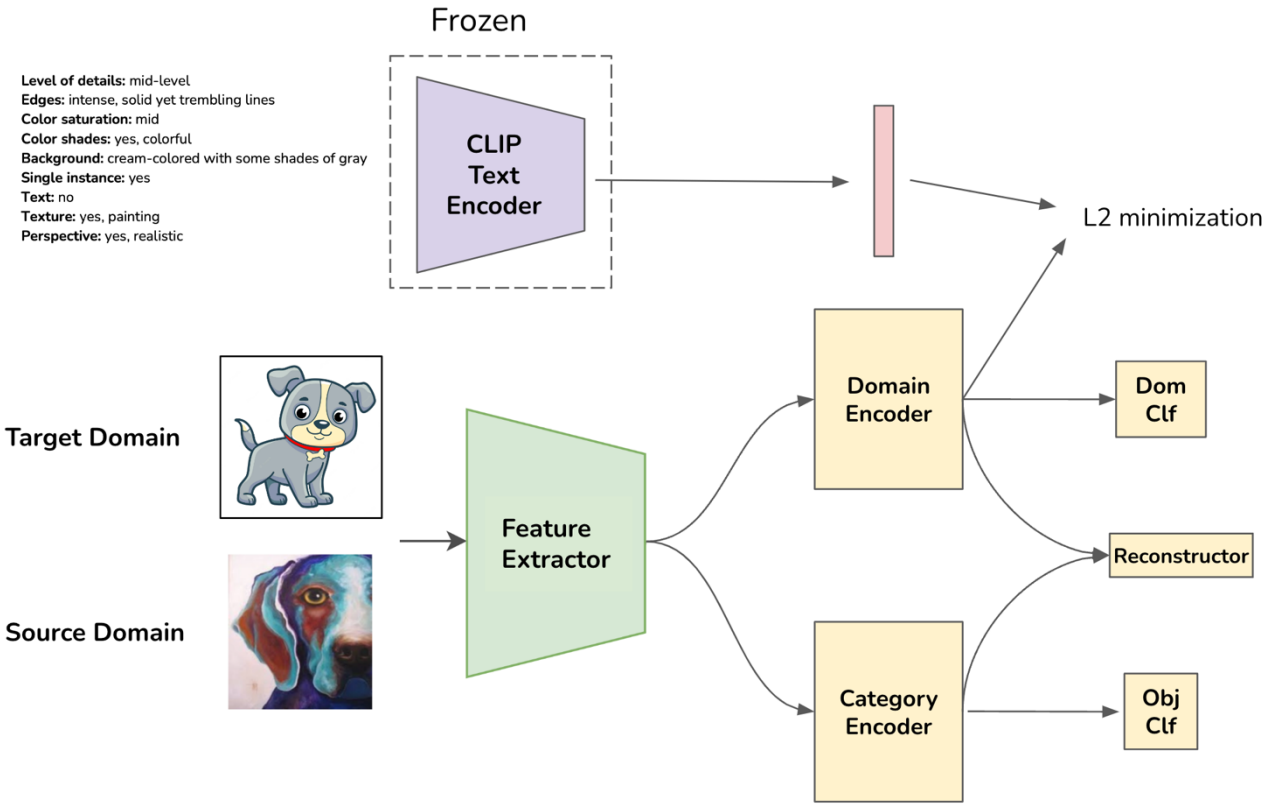
Example:

Level of details: mid-level
Edges: intense, solid yet trembling lines
Color saturation: mid
Color shades: yes, colorful
Background: cream-colored with some shades of gray
Single instance: yes
Text: no
Texture: yes, painting
Perspective: yes, realistic

N.B. You have to label at least 100 images for each domain. The labeled images have to be provided as part of the project.

4. Include CLIP using the labeled PACS

After the labeling step, you can include the text into the disentanglement process: you can use both the images that you labeled and those provided here.
In this step you should use the frozen **Text Encoder** of **CLIP** and minimize an L2 loss between the features from the Domain Encoder and those from the CLIP text encoder.
The idea is to increase the domain-specificity of the features to improve disentanglement.



Variations

Once all the mandatory steps have been completed you should choose at least one among the following variations:

1. Domain Generalization

In **Domain Generalization** (DG) we can use several labeled source domains to train the model that then has to work with a target domain, never seen during training.
The sources are characterized by different visual styles (e.g., sketch, cartoon, painting) and the target domain is characterized by a visual domain different with respect to all the sources (e.g., photo).
Read **[6]** to become familiar with the Domain Generalization setting, then repeat all the mandatory steps with this new configuration.

2. Fine-Tune CLIP model

Instead of using the pretrained text encoder of CLIP, try to use a finetuned version obtained using the PACS images with the domain descriptions.

References:

[1] Ganin, Yaroslav, et al. "Domain-adversarial training of neural networks." The journal of machine learning research 17.1 (2016): 2096-2030.

[2] Peng, Xingchao, Yichen Li, and Kate Saenko. "Domain2vec: Domain embedding for unsupervised domain adaptation." European conference on computer vision. Springer, Cham, 2020.

[3] https://openai.com/blog/clip/

[4] GitHub: https://github.com/iurada/Vision-Language-AML

[5] **PACS**: Li, Da, et al. "Deeper, broader and artier domain generalization." Proceedings of the IEEE international conference on computer vision. 2017.

[6] Carlucci, F.M., D'Innocente, A., Bucci, S., Caputo, B., Tommasi, T.: "Domain generalization by solving jigsaw puzzles." In: CVPR (2019)