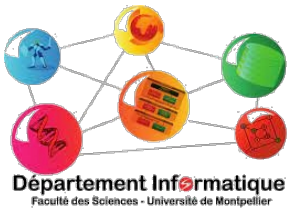


University of Montpellier
Faculty of Science

MASTER 2 INFORMATIQUE
IMAGINE course

AI and beauty modeling: the contribution of variational autoencoders to characterize the efficiency of cerebral information processing

Roland BERTIN-JOHANNET

Tutors at the host establishment: Jérôme Pasquet and Julien Renoult Tutor at the university: William Puech

Contents

Thanks

I'd like to thank Théo Oriol for lending me his access to the Jean Zay supercomputer, Sonia Mai for helping me sort out account problems on the Meso-LR mesocenter, and Nicolas Dibot for welcoming me to his office and helping m e g e t settled in at the CEFE.
I'd like to thank Julien Renoult and Jérome Pasquet for their excellent guidance and the freedom to maneuver they gave me, while providing me with a wealth of ideas.
I'd also like to thank William Puech for making sure that everything was running smoothly on several occasions.

Introduction

1 Brief summary of the course

We've provided a brief summary of the course to help you get your bearings later on. More detailed definitions of the various terms are given throughout the report.
The aim was to test different metrics of Fluency (the ease with which information is processed in the brain) as predictors of beauty. To model information processing in the brain, the idea was to use Variational Autoencoders (VAEs), which are Deep Learning models.

Initially, we set out to develop a VAE that would meet our various needs. This took some time, as the best-performing models in the state of the art had an architecture far removed from brain models, and simple models close to neuroscience models performed poorly.

Next, we sought to improve our models in two ways: first, we ex- plored different ways of doing without our autoencoder's decoder (explained below), in order to get rid of its bias. Secondly, we sought to add a sparsity constraint (defined below) to our VAE training to make it more biologically realistic.

We then sought to implement various fluency metrics on our trailed networks, and carried out various statistical analyses on these metrics.

After exploring these fluency metrics, we began to look at attention mechan- isms, and how the attention one pays to an image can indicate whether one finds it beautiful or not. We considered this approach at least partially independent of Fluency theory; nevertheless, we realized that with state-of-the-art attention mechanisms in Deep Learning, it was difficult to represent the type of attention we had in mind. We then interpreted our attention measure as a measure of image complexity, and performed statistical analyses on it.

This has not yet been achieved at the time of writing; nevertheless, the aim in the final month is to bring together the development work carried out during the internship in an easy-to-use code interface, with documentation and potentially explanatory videos.

2 Presentation of the host organization

TETIS (Territories, environment, remote sensing and spatial information): UMR Tetis is structured around two scientific dimensions, which are reflected in its title: "Territoires, envi- ronnement (thematic dimension), télédétection et information spatiale (methodological dimension). **CEFE**: CEFE is one of France's leading ecological research laboratories.

The CEFE project aims to understand the dynamics, functioning and evolution of living organisms, from "bacteria to elephants", and "from the genome to the planet".

3 Presentation of the mission

3.1 The beauty

Beauty is a phenomenon that has been studied for thousands of years (Plato and Aristotle, for example, discuss it in detail). Historically, explanations of this phenomenon were initially **universalist** (in Plato, beauty corresponds to what is good and advantageous), before the **subjective** dimension was taken into account (by Hume, for example). Universalist approaches explain the characteristics that make any object beautiful (symmetry, cleanliness, health, etc.), whereas subjective approaches emphasize that each observer perceives the object differently, and therefore the phenomenon of beauty depends on the observer. This internship is part of a project to study an **interactionist** approach to beauty (interactionist: beauty arises from the interaction of the observer with the observed, and not from one or the other alone): **Fluency theory**, according to which a *stimulus is beautiful if it is processed efficiently by the sensory system that perceives it*.

3.2 Ecological point of view

This internship took place at CEFE, an ecology laboratory. The ecological project behind our study of beauty is to find out whether animals have a sensitivity to Beauty, and if so, what are its origins and effects. Certainly, in our eyes, the tail of a peacock, the song of a bird and the agile movements of a cat are beautiful. There are several theories that attempt to explain the evolution of these signals in nature [42]. Here are three of them:

3.2.1 Disability theory:

[4] According to this theory, animals with a surplus of resources would invest in a "hand-icap" (the peacock's gigantic tail, for example, which prevents it from flying) that other animals could not afford to display. In the eyes of potential reproductive partners, the presence of this handicap would be an assurance that the individual possesses more resources than others. This me- canism may explain the extravagance of sexual signals, but it does not explain their specificity (the peacock's tail is long, but what about the refined and elaborate patterns visible on its feathers), universal aspects or diversity.

3.2.2 Fisher's drift:

[39] [5] This theory introduces a mechanism whereby if a preference exists for a certain characteristic in reproductive partners, then 1) having that characteristic is advantageous, and
2) so is having this preference, since it results in more attractive offspring. So, the more the characteristic is preferred, the more advantageous it is to prefer it. Add to this the fact that, as the characteristic is preferred, its presence in the population will increase, and we uncover a *race-to-the-bottom* mechanism that pushes the expression of an arbitrary characteristic to the extreme. This *race to the top* explains both the extravagance of the signals and their diversity, since the mechanism applies to an arbitrary characteristic. This mechanism can also explain the specificity of signals, but not their universal aspects.

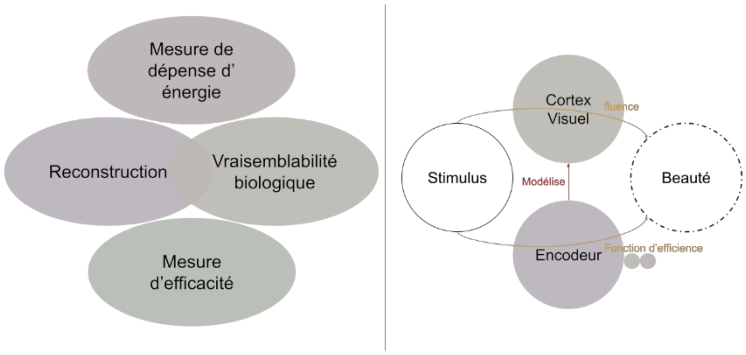


Figure 1: The aim of the internship is to establish efficiency and effectiveness metrics to extract from our model of visual cortex, a VAE, a measure of fluency that we are attempting to compare with beauty scores on faces.

3.2.3 Exploiting latent preferences :

[45] According to this theory, there are biases in animals that cause them to arbitrarily prefer certain signals to others, regardless of the usefulness of their meaning. Such a preference could, for example, be the result of a bias in the visual system that causes pleasure in response to certain fur patterns, or certain colors of scales. Such a preference is latent because it is a side-effect of the evolution of a perceptual system made to solve practical problems. Sexual signals would then evolve that exploit these latent preferences. This theory can explain both extravagance (exploiting preference to the full), diversity (perceptual systems are diverse), universal aspects (but share common characteristics), and specificity (perhaps the patterns on a peacock's tail are adapted to create a particular reaction in the observer's brain).

Finally: Do animals have aesthetic taste, as Darwin thought? Unlike humans, we can't ask them. As a result, we're seeking to establish a mathematical model of beauty in humans, which will then be tested in animals.

3.3 Motivation and idea for internship

As part of this internship, we are testing Fluency theory as an explicator of the phenomenon of beauty in humans (see figure 1 for details). To understand the mathematical model we are implementing, it is necessary to be familiar with **Fluency theory**, **Convolutional Neural Network** modeling of the brain, and **Variational Autoencoders**. [15] [44] [28]

3.3.1 Fluency Theory

- Fluency theory is one of the dominant theories to explain the phenomenon of beauty [57]. It states that beauty emerges as a consequence of the ease with which we can process information. information in the perceptual system[41]. This theory explains the preference for prototypes (stimuli that display the general characteristics of their category)[56], symmetry, etc.
  It is interesting to note that one of the shortcomings of this theory is that it does not explain the phenomenon of boredom when faced with a stimulus that is too simple. An approach in [11], based *on* the *process theory*

duals, adds to the theory of fluency a "need for cognitive enrichment" that drives the subject to seek out and appreciate the difficulty of processing information.

- Information processing fluency can be broken down into two aspects [44]: information processing **efficiency**, i.e. the amount of information extracted, and information processing **economy, i.e.** the amount of information processed.
  information processing, which is greater if less energy has been expended to extract the information.

- Mathematical approaches to fluency modeling have already been developed.
  Approaches without neural networks: In [43], images of faces are expressed as
  as combinations of predefined filters similar to those found in the human visual cortex. A measure of the sparsity of use of these filters in the combination is shown to be a good predictor of the beauty of these faces. In [2], information processing fluency is modeled as its correspondence to an observer's expectation. Reinforcement learning concepts are applied to add a learning dimension to this expectation.

- **Neural network approaches:** In the E3CO team where the internship took place, two approaches have already been implemented: Melvin Bardin has modeled fluency as typicality
  of the visual system's reaction to an image, Nicolas Dibot has modeled it as the sparsity of activations in the visual system.

3.3.2 Modeling the brain with Convolutional Neural Networks (CNN)

- **Neuroscience background:** In a famous experiment (see figure 2), Hubel and Wiesel [18] have demonstrated (in simplified terms) that certain neurons (known as "single cells") in the cortex
  The visual cells of cats respond selectively to visual bars of a certain orientation, at a certain location on the retina. Other cells (called "complex cells") respond to bars of a certain orientation, in a certain area of the retina. The proposed mechanism is that the complex cells integrate the responses of the simple cells by, for example, activating as soon as one of them is activated. The pattern detected by the complex cell is more abstract than that detected by the simple cell (see figure 4). In this way, a succession of layers of neurons, each integrating information from several neurons in the previous layer, would enable the cells at the top of the hierarchy to activate in response to highly complex stimuli (e.g. our grandmother's face, with the "grandmother cells").

- **Convolutional neural networks**: in [8], the neocognitron model is proposed, a model of the visual cortex based on Hubel and Wiesel's findings. Yann LeCun, inspired in part
  by the noecognitron, then created the Convolutional Neural Networks (CNN) [26], which are based on a convolution operation, where neurons activate in response to a certain pattern (simple cells), and a *pooling* operation, where the activations of several neurons are integrated into a single one (complex cells) using a max or mean operator, for example (see figure 3).

- **Validity as a model of the brain** Beyond the fact that the CNNs are directly inspired by the functioning of the visual system, there are numerous results that lead us to believe that
  that they are very good models [28]: CNNs tend to make the same type of errors as humans on image classification tasks. What's more, neural representations learned during CNN training are structurally correlated with representations in the brain, and activations in a CNN are good predictors of activations in the brain.

3.3.3 Variational autoencoders

- **Different types of learning in Machine Learning:** Supervised learning refers to a learning method based on pairs (input, output) where
  An output is known for each input, and the model must learn to predict the output. Unsupervised learning is a training method in which the model learns an output without being given any examples of these outputs. The best-known example of unsupervised learning is the K-Means algorithm. Some articles [51] point to the advantage of

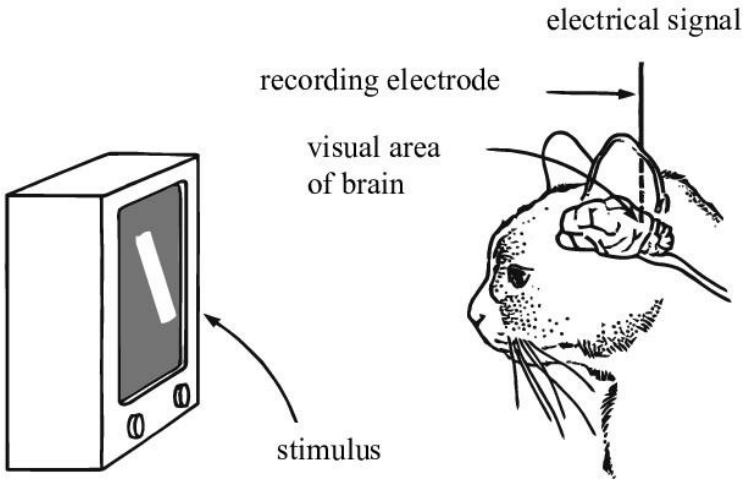


Figure 2: Device developed by Hubel and Wiesel: an electrode measures the activation frequencies of a single neuron, while the cat looks at a screen displaying a bar at different orientations.

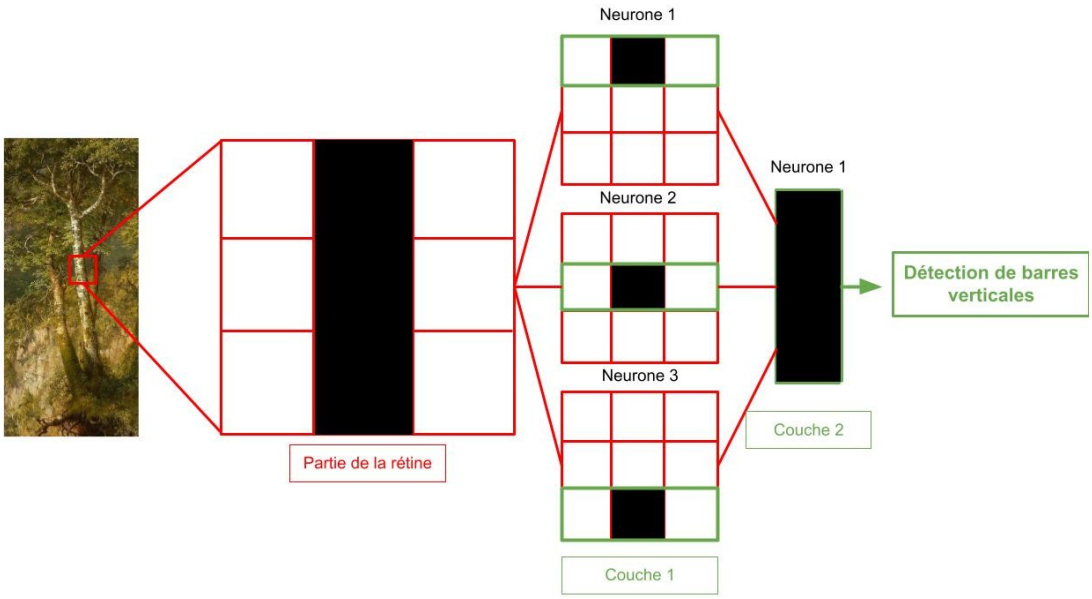


Figure 3: One of the various ways of intuitively understanding how CNNs work, in relation to visual cortex: if we simplify the trunk as a vertical bar, we can see that the three layer 1 neurons, if activated at the same time indicate a vertical bar. As the layer 2 neuron activates when the three layer 1 neurons are activated, it detects vertical bars. With more neurons and more layers, we can have neurons that activate in response to more complex patterns (e.g., a face).

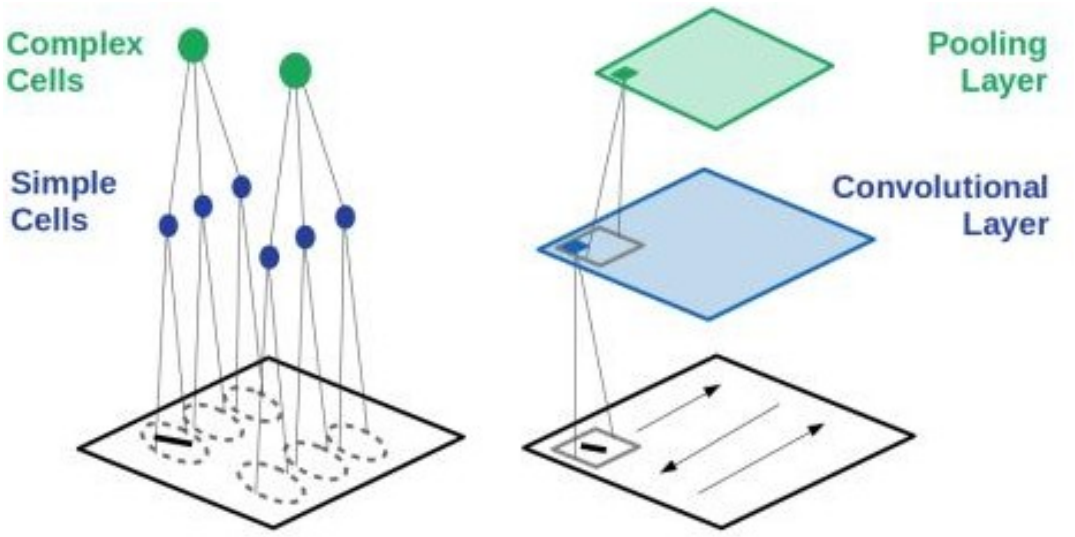


Figure 4: Link between pooling and convolution layers in CNNs, and simple and complex cells in visual cortex: complex cells integrate activations of simple cells, and pooling layers integrate activations of convolution layers.

self-supervised methods (very similar to unsupervised methods) to obtain good brain models.

- **Variational Autoencoders (VAEs)** VAEs [23] are models designed to address a number of problems within the framework of Bayesian inference (see Figure 5 for more details).
details). Briefly, Bayesian inference consists in estimating probability distributions by incorporating *a priori* knowledge into our estimate. This is expressed mathematically by **Bayes' Theorem**:

$$P(X) = \frac{P(X/Z)\,P}{(Z)\,P(Z/X)}.$$

Where $P(Z)$ is called the *à-priori* and $P(Z/X)$ is called the *à-posteriori*.
To get a clearer idea of how VAE works, let's take a concrete example directly related to
our subject. Let's imagine that we have at our disposal a database of X images from which we draw X images$_i$. A square image of 255 pixels on a side is heavy and there are many redundancies (spatial auto-correlation, etc.) as well as compression that could be achieved with a semantic understanding of the image. So we're looking to extract a low-cost representation of the image.
dimension, Z, of our high-dimensional base X. In particular, for a given X, we want to have a distribution of possible Z's, i.e. $P(Z/X)$ (the *à-posteriori*). Analytically, starting from Bayes' theorem,

$$P(X) = \frac{P(X/Z)\,P}{(Z/X)} \Rightarrow P(Z/X) = \frac{P(X/Z)\,P(Z)}{P(X)}$$

But $P(X)$ has no analytical solution (certainly in our example it does). Thanks to VAEs, we can estimate the *à-posteriori* without needing $P(X)$. The VAE has (see figure 5 for a diagram of the architecture) a stochastic encoder Q_ϑ (ϑ represents the VAE parameters) which will take an image X_i as input and give a distribution $Q_\vartheta(Z_i/X_i)$ as output. The decoder, P_ϑ , takes Z_i as input and outputs $P_\vartheta(X_i/Z_i)$, (note that this is necessary for the theory, but in practice we are content to say that the decoder outputs $x\hat{}_i$ directly), $x\hat{}_i$ being its approximation of X_i .
Since VAE is a generative model, its objective is to maximize the probability of

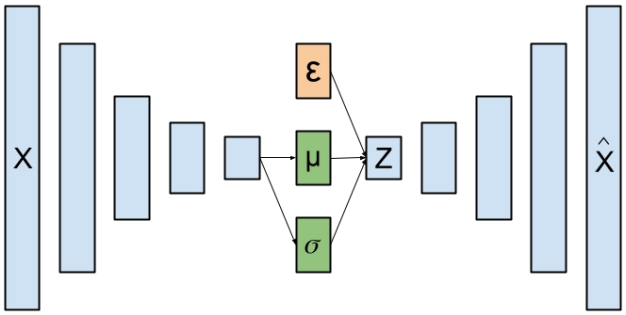


Figure 5: Variational Autoencoders consist of an encoder (convolutional layers) and a decoder (techniques detailed later). The VAE encoder is stochastic, and the *re- parametrization trick* consists in removing the stochasticity from the encoder itself, by asking the encoder to generate a mean μ and a standard deviation σ, then draw a sample ϵ from a reduced centered normal distribution, multiply it by σ and add μ to it.

natural images in the distribution of the images it generates. In other words, we seek to maximize $P(X)$, or $ln(P(X))$, which is equivalent and easier. However,

$$ln(P(X)) = ln(\int P_\vartheta(X|Z)P(Z)dz)$$

(Where P(Z) is an a-priori that we can choose on the form that P(Z|X) must take (for example, a Gaussian)).

$$\Rightarrow ln(P(X)) = ln(\int \frac{P_\theta(X|Z)P(Z)}{Q_\theta(Z|X)} * Q_\vartheta(Z|X)dz)$$

$$\Rightarrow ln(P(X)) = ln(E_{z \sim Q_\theta(Z|X)} \frac{P_\theta(X|Z)P(Z)}{Q_\theta(Z|X)})$$

By Jensen's inequality

$$\Rightarrow ln(P(X)) \geq E_{z \sim Q_\theta(Z|X)} ln(\frac{P_\vartheta(X|Z)P(Z)}{Q_\theta(Z|X)})$$

$$\Rightarrow ln(P(X)) \geq \mathbf{E}_{z \sim Q(Z|X)}[ln(P_\vartheta(X|Z)) + ln(P(Z)) - ln(Q_\vartheta(Z|X))]$$

Finally,

$$ln(P(X)) \geq \mathbf{E}_{z \sim Q_\theta(Z|X)}[ln(P_\vartheta(X|Z))] - \mathbf{E}_{z \sim Q_\theta(Z|X)}[ln(Q_\vartheta(Z|X)) - ln(P(Z))] \quad (1)$$

The expression on the right-hand side of equation (1) is the VAE maximization objective, called *Evidence Lower Bound*(ELBO). Since $ln(P(X))$ is greater than this, maximizing ELBO implies maximizing $ln(P(X))$ and therefore also $P(X)$.
If we manage to train our VAE well so that it maximizes ELBO, then the encoder will give us our *à-posteriori* values $Q_\vartheta(Z|X)$.
ELBO consists of two terms:

- $\mathbf{E}_{z \sim Q(Z|X)}[ln(P_\vartheta(X|Z))]$ represents the probability of images X in the distributions generated by the decoder. The model is therefore well-trained to maximize the probability natural images in the distributions it generates.

- $\mathbf{E}_{z\sim Q(z|x)}\,[ln(Q_\vartheta(Z|X)) - ln(P(Z))]$ is the Kullback-Leibler distance between the distributions generated by the encoder, and the *à-priori* we've chosen. This term implies only the encoder, and penalizes it if it ignores the *a-priori* we've set on the form its distributions should take.

It is common practice to use a modified version of this cost function, as ELBO as it stands tends not to give the best performance. In our case, we replace the first term by the LPIPS perceptual distance [59] between the input and output images. In addition, we choose a centered reduced Gaussian as an *à-priori.*

Things to remember:
In this way, VAEs unsupervised learn a latent representation of the input, which can then be inverted to retrieve the input. In our case, we're simply dealing with a model which, for an input image, will give us a distribution over the latent space. If we sample a value from this distribution and then decode it with the decoder, we obtain another image that must be close to the input image.

- **Usefulness of VAEs in our case :** The usefulness of VAEs in our case is manifold: firstly, it should be noted that the "encoder" part of the VAE is nothing other than a CNN, and secondly, the "encoder" part of the VAE is nothing other than a CNN.
As a result, this is a good model of the brain from which to extract metrics of information processing *economy*. In addition, the ability to reconstruct images from the latent representation provides a metric of information processing *efficiency*. Ultimately, unsupervised learning seems a more realistic training method, as for humans in the real world access to "ground truths" is rare but learning is constantly taking place, indicating unsupervised learning.

- **Usefulness of the reparametrization trick** (The reparametrization trick is explained in figure 5) Here we justify analytically the use of the re-parametrization trick.
To anchor our justification, let's explain how a CNN-type model is usually trained by a **gradient descent** algorithm described as follows:

1. Run an image through the model to obtain the outputs.
2. For these outputs, calculate the cost function (ELBO in our case).
3. By backpropagation (recursive application of the rule $\frac{\delta f(g(x))}{\delta x} = \frac{\delta f(g(x))}{\delta g(x)} * \frac{\delta g(x)}{\delta x}$), find the ∇ gradient$_w$ of the cost function with respect to each of the model parameters w.
4. Change each of the weights w with the rule $w \leftarrow w - \nabla_w * \lambda$ where λ is a parameter we choose, called the *learning rate*.
5. Repeat steps 1 to 4 until the cost function converges.

The reparametrization trick takes place in step 3.
So let's assume we're trying to approximate the ELBO gradient without the reparametrization trick:

$$\nabla_\theta \mathbb{E}_{\vartheta Q(Z|X)}\,[log(p_\vartheta(x|z)) - log(\frac{Q_\theta(Z|X)}{p(Z)})$$

This is equal to

$$\nabla_\theta \int_\vartheta [(log(p_\vartheta(x|z)) - log(\frac{Q_\theta(Z|X)}{p(Z)})) * Q_\vartheta(Z|X)]dz$$

In general, the conditions are met for passing the gradient in the integral. Moreover, since the gradient is distributive over the multiplication ("product rule") :

$$= \int_\vartheta (\nabla_\vartheta [log(p_\vartheta(x|z)) - log(\frac{Q_\vartheta(Z|X)}{p(Z)})]) Q_\vartheta(Z|X)) dz + \int_\vartheta ((log(p_\vartheta(x|z)) - log(\frac{Q_\vartheta(Z|X)}{p(Z)})) * \nabla_\vartheta [Q_\vartheta(Z|X)])dz \quad (2)$$

We end up with two integrals in equation (2).
In practice, the idea is to approximate the value of the integrals by sampling several values (*Monte Carlo* approach). The problem here lies in the second integral, which is not

in the form of an expectation according to $Q_\vartheta(Z|X)$: this does not allow us to estimate it stochastically in Monte Carlo. We can do so, however, as there's nothing to stop us implementing it,
but the gradient estimates will have a high variance.
On the contrary, let's take a look at what happens if we use the re-parametrization trick:

With the re-parametrization trick, instead of directly generating $Q_\vartheta(Z|X)$ with the encoder, we generate the vectors μ_ϑ and σ_ϑ. To sample z, we sample a value $\epsilon \sim N(0, Id)$ and calculate $z = \mu_\vartheta + \sigma_\vartheta * \epsilon$. This procedure is equivalent to sampling an N law $(\mu_\vartheta, diag(\sigma_\vartheta))$. Nevertheless, if we now calculate the ELBO gradient (as for equation (2)):

$$\nabla_\theta \mathbb{E}_{\vartheta Q(Z|X)}[log(p_\vartheta(x|z)) - log(\frac{Q_\theta(Z|X)}{p(Z)})] \text{ becomes}$$
$$\nabla_\theta \mathbb{E}_{\vartheta p(\epsilon)}[log(p_\vartheta(x|\mu_\vartheta + \sigma_\vartheta * \epsilon)) - log(\frac{q_\theta(\mu_\theta + \sigma_\theta * \epsilon|x)}{p(\mu_\theta + \sigma_\theta * \epsilon)})] \text{ Let's}$$

keep in mind that $z = \mu_\vartheta + \sigma_\vartheta * \epsilon$. We can write:

$$\begin{aligned}
&\nabla_\theta \mathbb{E}_{\vartheta p(\epsilon)}[log(p_\vartheta(x|\mu_\vartheta + \sigma_\vartheta * \epsilon)) - log(\frac{q_\theta(\mu_\theta + \sigma_\theta * \epsilon|x)}{p(\mu_\theta + \sigma_\theta * \epsilon)})] \\
&= \nabla_\vartheta \int [(log(p_\vartheta(x|z)) - log(\frac{Q_\theta(Z|X)}{p(Z)})) * p(\epsilon)]d\epsilon \\
&= \int \nabla_\vartheta [p(\epsilon) log(p_\vartheta(x|z)]dz - \int \nabla_\vartheta [p(\epsilon) log(\frac{q_\theta(Z)}{p(Z)}]d\epsilon
\end{aligned}$$

Now, this time, since we are making the expectation on ϵ which does not depend on ϑ, we can re-write this expression as follows:

$$\nabla_\theta \mathbb{E}_{\vartheta p(\epsilon)}[log(p_\vartheta(x|\mu_\vartheta + \sigma_\vartheta * \epsilon)) - log(\frac{q_\vartheta(\mu_\vartheta + \sigma_\vartheta * \epsilon|x)}{p(\mu_\vartheta + \sigma_\vartheta * \epsilon)})] = \mathbf{E}_{p(\epsilon)} \nabla_\vartheta [log(p_\vartheta(x|z)] - \mathbf{E}_{p(\epsilon)} \nabla_\vartheta [log(\frac{q_\vartheta(Z)}{p(Z)}] \tag{3}$$

We can see from equation (3) that, thanks to the re-parametrization trick, we only have esperances, and in practice the variance of *Monte-Carlo* estimates of the gradient will be lower, and model training will therefore be more stable.

3.3.4 Assembling the concepts introduced

Now that we've introduced all the concepts, we can explain the aim of the course: to train VAEs on images of faces, and extract metrics of information processing *economy* and *efficiency*, to see if they correlate and predict beauty.
An additional objective is to provide an interface for re-training the models and obtaining the metrics on any new database. In this way, the mathematical model can be tested on animals.

4 Technical environment

4.1 Meso-LR computing cluster

The Meso@LR mesocenter is a specialized center offering shared resources and state-of-the-art infrastructures for High Performance Computing (HPC) and massive data processing.
Thanks to their CPUs with lots of RAM (3TB), we were able to finish running the code from last year's course, where ACPs on VGG16 activations were taking up a lot of memory.

4.2 Jean Zay supercomputer

Jean Zay represents the latest converged supercomputer recently acquired by the French Ministry of Higher Education, Research and Innovation, in collaboration with GENCI (Grand équipement national de calcul intensif).

This work has benefited from access to IDRIS computing resources through the allocation of resources by GENCI to the uzk68zl account.
Thanks to the V100 GPUs available at Jean Zay, we were able to run our models 4 times faster than with our local GPU, with the added advantage of being able to run several training sessions at the same time. We had access to them rather late in the course, so we only benefited from them for training models with attention mechanisms and ethnic and gender-specific models.

4.3 Language, libraries,GitHub pages

- Any implementation of Deep Learning architectures requires the use of an auto-differentiation library. Our choice was Pytorch [38].

- We chose the Python language because it's at a level of abstraction suited to our internship objectives (not so simple that we can't go into detail, but abstract enough).
  to save time).

- We've used code from various GitHub pages. Here's the list (all terms are defined later in the report):
  For LPIPS perceptual distance: GitHub link For
  AttGAN pre-trained model: GitHub link
  For a differentiable implementation of the SSIM similarity index: GitHub link

5 General method

5.1 Databases

In this course, we used two face databases.

- The CFD database [30] consists of around 800 standardized images of faces, divided by ethnicity and gender. For each face, a beauty score was assigned by a jury.
- The Fairface database is made up of a hundred thousand non-standardized images of faces collected on the Web, classified into ethnic and gender categories from size to height. approximately equal. [22]

The exclusive use of images of faces is questionable, as a better model of the visual cortex would be obtained by training our model on a database representative of natural images in general.

5.2 Neural processing fluency modeling (metrics)

5.2.1 Sparsity measurements

An intuitive way of measuring the simplicity of information processing in the brain is the sparsity of neuronal activations. Sparsity indicates the extent to which activations are localized to a small number of neurons. Typically, if a few neurons activate strongly and all the others remain at zero, the activations are sparse. On the contrary, if all neurons are half-activated, the activations are not sparse.
As a measure of sparsity, we used the following two metrics:

1. The gini index: a measure of wealth inequality in a population. A gini index of 1 corresponds to absolute inequality, and a gini index of 0 corresponds to absolute equality (see figure 6).
2. The L norm$_\epsilon$: this is calculated by counting the number of neurons that are activated above a threshold ϵ. If many neurons have activated above this threshold, the neuronal activation pattern is not considered sparse.

In [19], the authors propose a number of criteria that a measure of sparsity must meet. They compare a number of different measures, concluding that the only one that meets all their criteria is *the Gini index*.

5.2.2 Reconstruction quality

We used reconstruction quality as a measure of neural processing efficiency: the better the VAE reconstructs an image, the better the encoder (the visual cortex model) is judged to have extracted the necessary information. As a measure of reconstruction quality, we use the LPIPS perceptual distance [59] between the input image and the VAE output image. The smaller the distance, the more likely it is that the encoder has correctly understood the input image.

5.2.3 Sharpness Aware Minimization

Intuition :
Sharpness Aware Minimization (SAM) [7] is a modified version of gradient descent, where the aim is not only to minimize the cost function, but also to move towards a "flat" minimum, i.e. the values of the cost function in the neighborhood of the minimum remain low (see figure 8 for a visual explanation). The intuition is that a "sharp" (as opposed to "flat") local minimum indicates that the slightest change in the model's parameters would cause its knowledge to collapse. Avoiding this fragility of the model would guarantee a better capacity for generalization.

How it works :
Instead of trying to directly minimize the cost function $L(W)$ (W the model parameters), we try to minimize a modified version of the cost function :

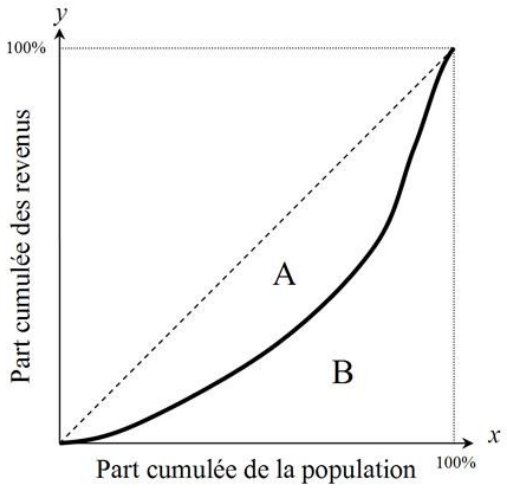


Figure 6: The gini index comes from economics, where it is used to measure inequality in a distribution. It is equal to the area A on the drawing.

$$L_{SAn}(W) = max_{\|\epsilon\|<\rho}(L(W+\epsilon))$$

In words, this expression means that we minimize the maximum of the cost function in a zone around W (the size of the zone being given by ρ, a parameter we choose).
However, a problem arises: the *max* operator used is not derivable, and to train our model, we need to derive the cost function. Let's see how we can approximate the following value:

$$_W \nabla[max_{\|\epsilon\|<\rho}(L(W+\epsilon))]$$

The trick is this: if we can find the $\hat{\epsilon}$ that maximizes the cost function in the area defined by ρ, we can simply replace the expression with :

$$_W \nabla[max_{\|\epsilon\|<\rho}(L(W+\epsilon))] = \nabla_W[L(W+\hat{\epsilon})]$$

We can then reason as follows:

1. The gradient of a function indicates the *local direction of maximum increase in the function's value*.
2. Provided ρ is small (and it is for us), our search for $\hat{\epsilon}$ can be considered local.
3. Conclusion: we can approximate $\hat{\epsilon}$ by :

$$\hat{\epsilon} \approx \rho * \frac{_W \nabla[L(W)]}{_W \|\nabla[L(W)]\|} \quad (4)$$

The modified gradient descent algorithm then becomes :

1. Run an image through the model to obtain the outputs.
2. For these outputs, calculate the cost function $[L(W)]$.
3. By backpropagation, find the gradient $\nabla_W[L(W)]$ of the cost function with respect to each of *the* model parameters *w.*
4. With this gradient, calculate $\hat{\epsilon}$ according to equation (4).
5. Add $\hat{\epsilon}$ to W and calculate the cost function $L(W+\hat{\epsilon})$ with these new parameters.
6. By backpropagation, find the gradient $\nabla_W[L(W+\hat{\epsilon})]$.

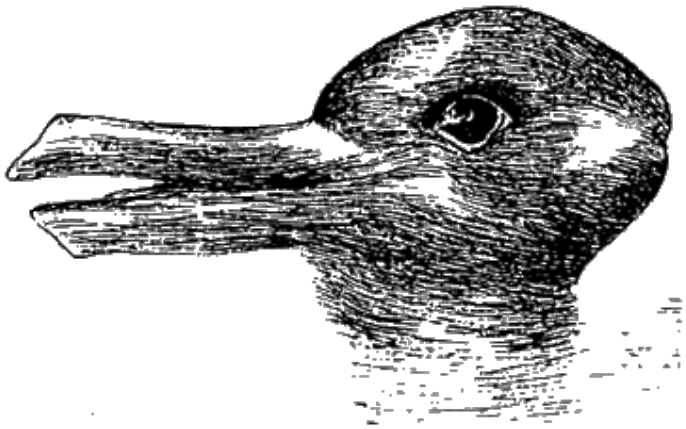


Figure 7: Demonstrating the importance of visual attention: if we pay attention to the pointy shapes on the left, we see a duck, but if we pay attention to the protuberance of the duck's skull on the right, we suddenly see a rabbit. In addition, when we finally understand the image, we feel pleasure, which is explained by fluency theory as the pleasure of information gain.

7. Remove $\epsilon\hat{}$ from the model parameters to return to W.
8. Change each of the weights w with the rule $w \leftarrow w - \nabla_w [L(W + \epsilon\hat{})] * \lambda$.
9. Repeat steps 1 to 8 until the cost function converges. A visual

representation of this algorithm is shown in figure 8.

Using SAM in our case :

We first tested SAM's ability to generalize models (see Fig. 9 for details of the training procedure). For two models trained on Fairface, one with SAM and the other without, there was no difference in terms of reconstruction quality on CFD. So, in the case of our VAEs, SAM made no contribution to training.

Nevertheless SAM did not interest us as a means of improving our models, but as inspiration for a measure of the model's understanding of images. If $L(W + \epsilon\hat{})$ is the local maximum of the cost function, and $L(W)$ is its local minimum, then the value $L(W + \epsilon\hat{}) - L(W)$ measures the amplitude of local change in the cost function - in other words, its *sharpness* local.
If, moreover, local acuity is a measure of the robustness of the model at point W and for an input image, we can use this local acuity measure as a measure of how *well the model has understood an input image*. We could therefore use this as a measure of the efficiency of neural processing and therefore of fluency, which could be a good predictor of beauty.

5.2.4 Attention

Attention in relation to aesthetics We are interested in the effect of attention on the experience of beauty. A telling example in terms of visual attention in relation to Fluence theory is that of optical illusions (see figure 7). According to Shaeffer [47], aesthetic attention is characterized by three features that separate it from "normal" attention:

- *Attentional densification*: for a given dimension of the stimulus (e.g. color), a subject in a state of aesthetic attention will perceive finer differences in this dimension.

Figure 8: If we represent the reconstruction error as a two-dimensional heightmap, which are two parameters of the model, a smoother curve (on the right) indicates greater robustness of the reconstruction to a change in the parameters, indicating a better understanding of the image, and thus a higher "efficiency" of information processing.

dimension, i.e. the number of values (colors) it can differentiate will be greater.

- *Attentional saturation*: the subject will pay attention to more dimensions of the stimulus (color, contour, physical presence, texture...) at once than would otherwise be the case.

- *Bypassing perceptual schemas*: perceptual schemas are abstract patterns according to which we categorize the objects we perceive. With the help of perceptual
ceptual attention, we can obtain all the information we need about a stimulus in a single glance (for example, to see a shoe, all we need to do is look at its general shape, and unless the sit- uation demands it, we don't look any further). In aesthetic attention, perceptual schemas would be bypassed, with attention focused on the perceptual experience rather than on object categorization.

These characteristics of aesthetic interaction with an object imply the abandonment of shortcuts and careful examination. They therefore seem contrary to the idea that the most beautiful stimuli would be the most easily processed (Fluence). [11] proposes a model of aesthetics that takes into account this additional dimension of aesthetics associated with effort and learning.

5.3 Development o f a Variational Autoencoder

In the state of the art, there are VAEs with very good reconstructions [3] [31]. For example, reconstructions of the nVAE model [53] can be seen in Figure 10. However, these are hierarchical models with residual connections and sampling at different levels, and we have doubts about their usefulness as models of the brain. The aim here was therefore to create a VAE that

- Has an encoder similar to the CNNs that have been tested as brain models (AlexNet [25], VGG16 [49]...).

- Produces quality reconstructions.

To our knowledge, there is no model in the state of the art that combines these two qualities.

We began by implementing and training on Fairface a Variational Autoencoder (VAE) with the architecture given in the original article [23]. To link reconstruction quality to the encoder's *understanding of* the image, reconstructions must be good. This was not the case with the basic architecture.

We have tested many techniques to improve the quality of our reconstructions. Here we list all these techniques with a brief explanation and figures showing their positive or negative effect.

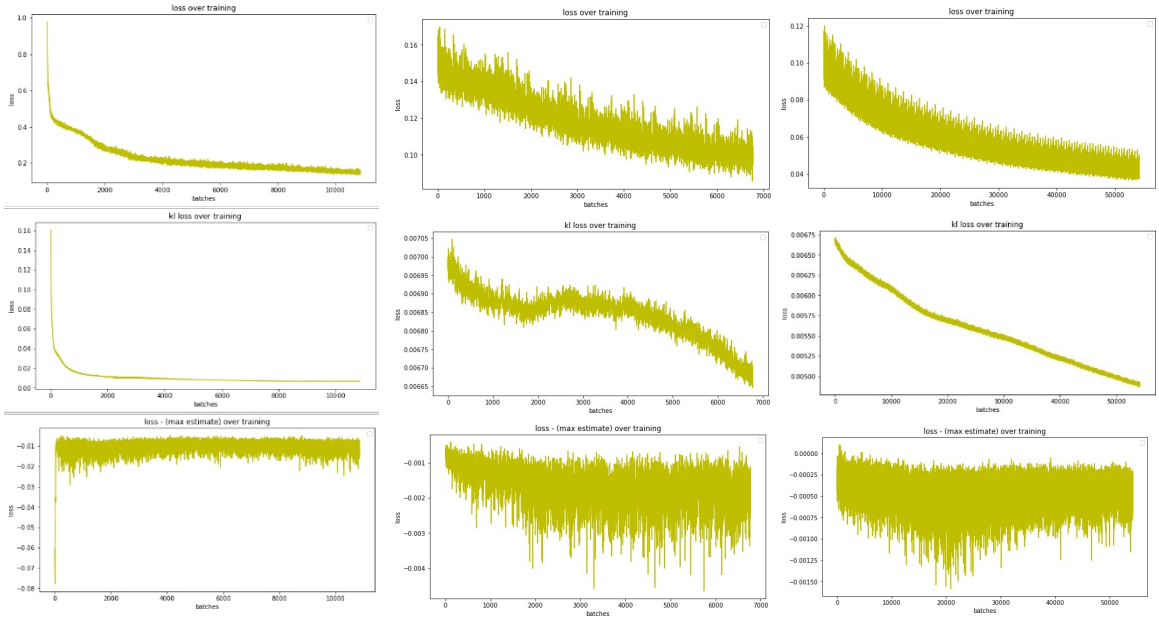


Figure 9: Cost functions during VAE training with Sharpness Aware Minimization - columns: Three training phases (the model was trained three times) - row 1: error of
reconstruction; line 2: value of the Kullback-Leibler divergence of the sparsity constraint; line 3: the value $L(\epsilon)$ - $L(\epsilon\hat{})$. We see that it remains negative, indicating that the gradient is still a good estimate of the direction o f maximum increase in the cost function.

Figure 1: 256×256-pixel samples generated by NVAE, trained on CelebA HQ [28].

Figure 10: Reconstructions given by the nVAE model. they are very good, but the architecture is far from brain models.

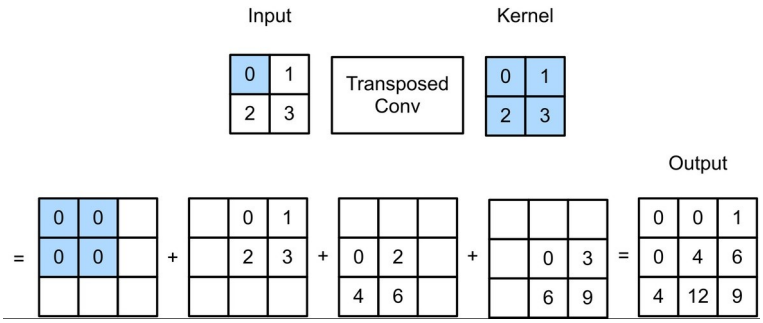


Figure 11: How ConvTranspose2d works: a kernel is multiplied by the value of one pixel at a time, and each time, the result is added to the output image to the pixels around the position of the pixel in question.

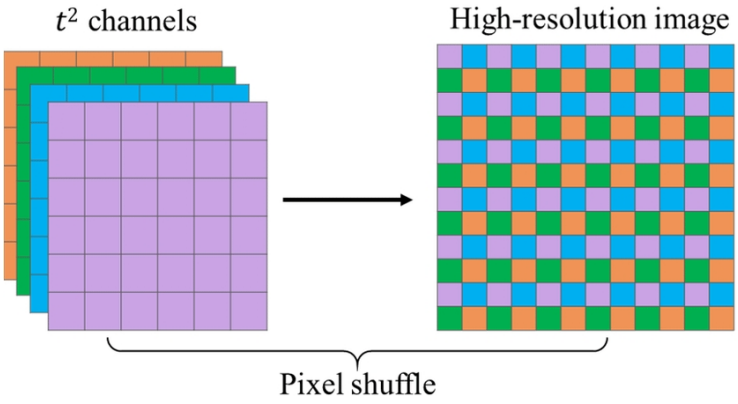


Figure 12: The principle behind PixelShuffle is to convolve many filters to obtain many different featuremaps. The featuremaps are then combined to form a larger image, as shown in the figure.

5.3.1 Upsampling methods :

see figure 16 for results
Convtranspose: This is the most widely used upsampling method in image recon- struction or generation methods. It resembles convolution, but works differently (see figure 11).

Traditional upsampling followed by convolution: ConvTranspose is known to cause undesirable checkerboard patterns on output images. One proposed solution [34] is to perform a traditional upsampling without parameters (e.g. by linear interpolation), followed by a classical convolution that allows parameters to be included in the operation.
Pixelshuffle: [48] is another alternative method to ConvTranspose. It involves applying a large number of convolution filters to obtain feature maps. These filters are then combined to form a larger image (see figure 12 for a visual explanation).

5.3.2 Differences in latent space :

Latent space by fully-connected or by convolution: In the architecture of the original article, latent space vectors are obtained by a fully-connected layer (a matrix multiplication whose parameters are all learned during training). We have tried this alternative, as well as the possibility of obtaining these vectors via a convolution layer (which allows us to take

information while reducing the number of parameters).
Latent space size: Choosing the size of the latent space is important: too small and the model is overly constrained, resulting in poor reconstructions. If it's too large, it makes the task easier for the model, which is no longer forced to learn well (see additional figure 21).

5.3.3 Cost functions :

see figure 17 for results
Simple distance measurements (L1 and L2 standards): L1 (sum of the absolute values of the differences between pixels) and L2 (sum of the squares of the differences between pixels) are the most commonly used standards for Autoencoder training.
SSIM: The SSIM structural similarity index[55] is a widely used measure of similarity between two images. It is based on the extraction of features from sub-parts of the two images and the comparison of these features.
Deep feature consistent VAE (DFCVAE) cost function[16]: The idea of this cost function is to mimic human similarity judgment with a neural network that serves as a model of the vi- sual cortex. To do this, we run the two images to be compared through the network to obtain the extracted representations, and then calculate an L2 norm between the two representataions. We have tried this technique with different networks as models of the visual system: attGAN [13], VGGFace[49], and a basic VAE.
LPIPS: the LPIPS distance [59] (Learned perceptual image patch similarity) is based on the same idea as the DFCVAE cost function, but with a few additions, including a multiplication of the representations on the different layers by scalars to better approximate human judgment.

5.3.4 Regularization methods :

Batch normalization: Batch normalization [21] consists in normalizing the output of each model layer (z-transform), then multiplying it by a mean and a standard deviation. Among other things, this addresses the *vanishing gradient* problem and reduces interdependency between layers. The *Normalization Instance* [52] does the same thing, but with a mean and standard deviation per image and per layer. We don't show the results here, as their use makes no difference.

5.3.5 Residual connections :

The main idea behind residual connections [12] in Deep Learning is to add the input of a layer to its output. If the input is x, and the non-idual layer gives the output $f(x)$, the residual layer will give $f(x)+x$. The primary intuition behind this mechanism is: convolution layers tend to learn the function $f(x) = 0$ more easily than the function $f(x) = x$. The addition of x in the layer output allows the network to focus on small differences rather than on
information. We have not used this technique in the encoder to keep it as similar to VGG-type models as possible.

5.3.6 Different models

Once we had decided which of the above techniques to keep, we experimented with different sizes for the encoder and decoder. We tested more architectures than are relevant to detail here; three examples are provided in Figure 13.

5.3.7 Study of representations

VAEs are able to extract, in an unsupervised way, structure between the different images they are given as input. In this way, it is possible to interpolate in the latent space between the images, and obtain a relevant non-linear interpolation in the output space (see figure 18). What's more, thanks to this structure, we might expect that in latent space, faces of a certain ethnicity

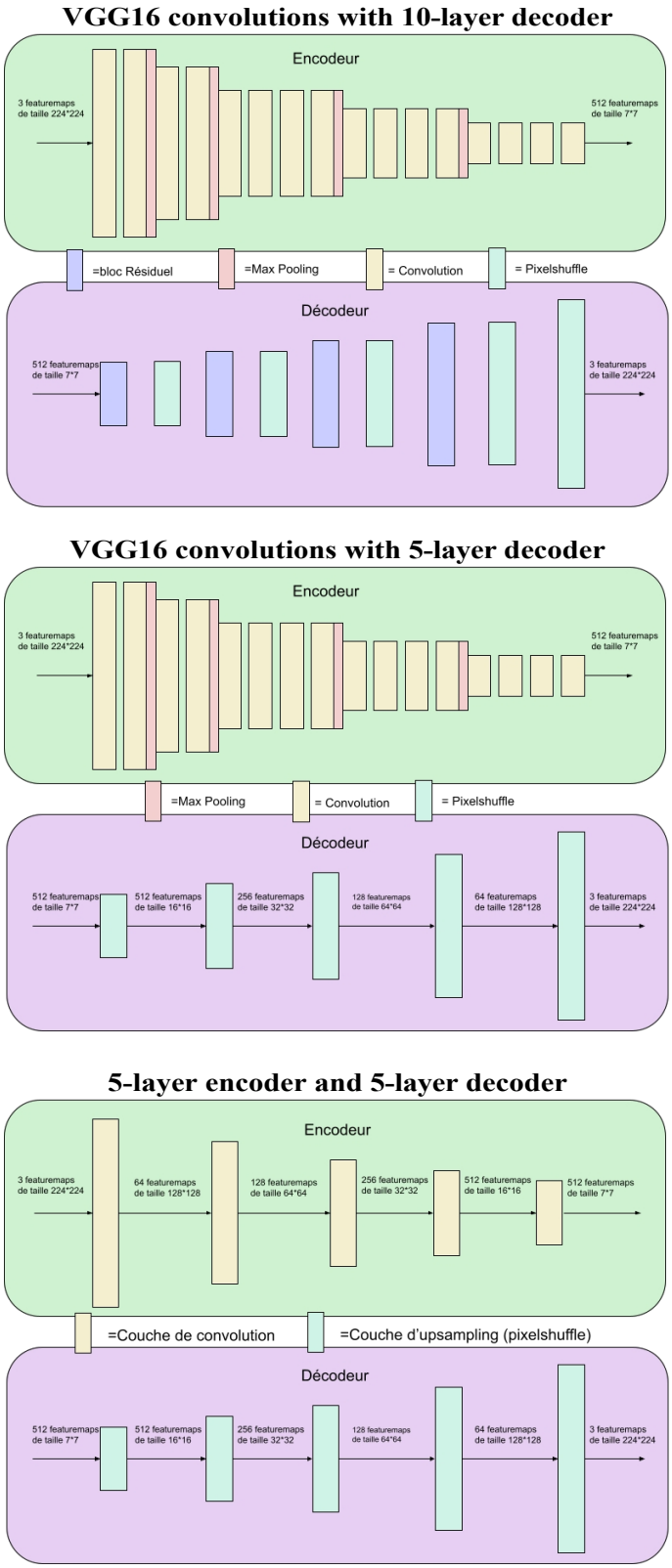


Figure 13: diagrams representing the 3 main architectures tested

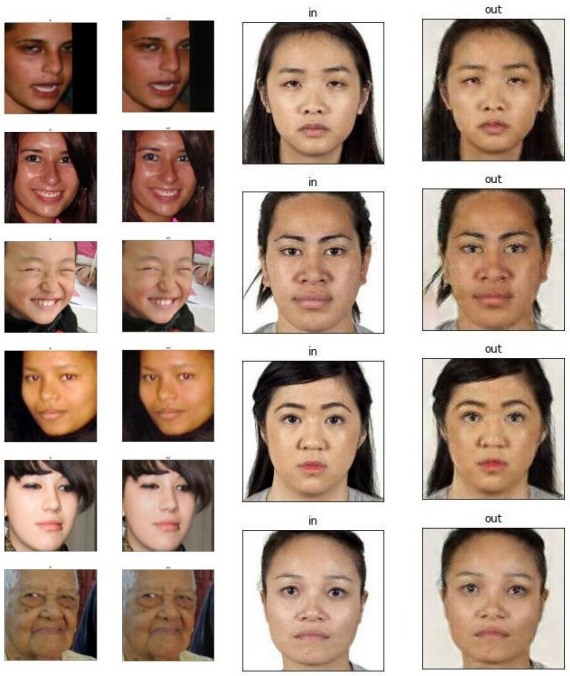


Figure 14: Reconstructions of the small model (5 layers a t encoder and decoder) on Fairface on the left, and CFD on the right. In each case, the original image is shown, followed by the reconstruction.

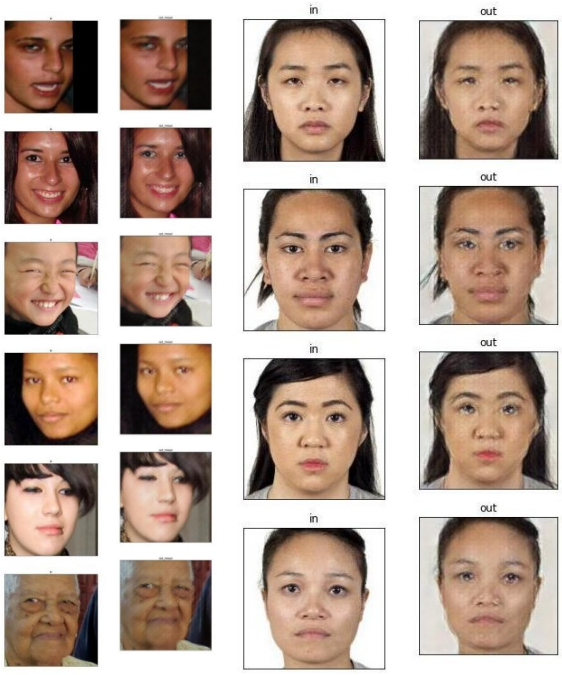


Figure 15: Reconstructions of the model with VGG encoder and 5-layer decoder on Fairface on the left, and CFD on the right.

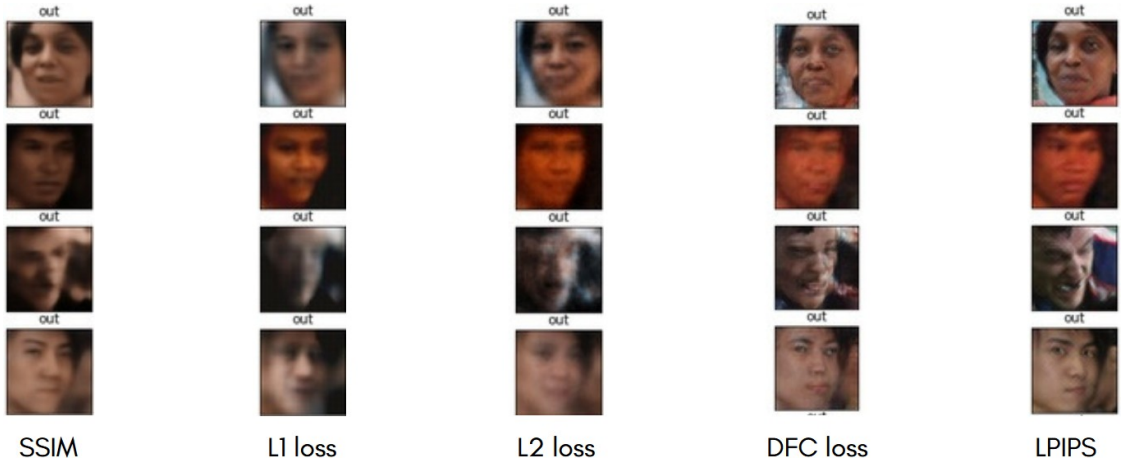


Figure 16: Effect of different upsampling methods on small model reconstructions. LPIPS seemed to be the best for our three architectures.

Figure 17: Effect of different training cost functions on small model reconstructions. We chose to use PixelShuffle in the end.

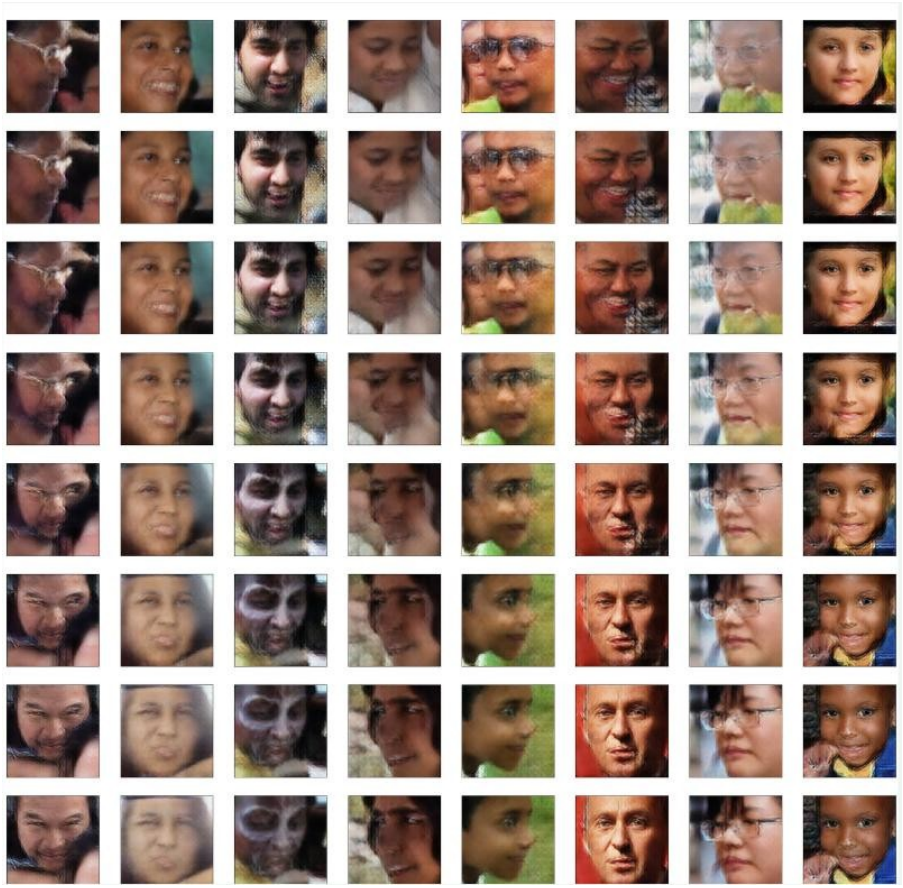


Figure 18: Linear interpolation in latent space between Fairface image representations and the small model. We decode each point of the interpolation to obtain the above transitions. It seems that the model always tries to represent a face, which shows a certain robustness of the model (even if there are better results in the state of the art).

are in the same zone, those in another are a little to the side, and so on. We therefore obtained figure 19 (left) to test this, and see if structure was not extracted in relation to beauty scores (beautiful faces being found more in a certain area than in another. No structure appeared in the latent space, but with a latent space four times smaller, we obtained separation between ethnicities and genders (see figure 19 (left)). However, no structure related to beauty emerges.

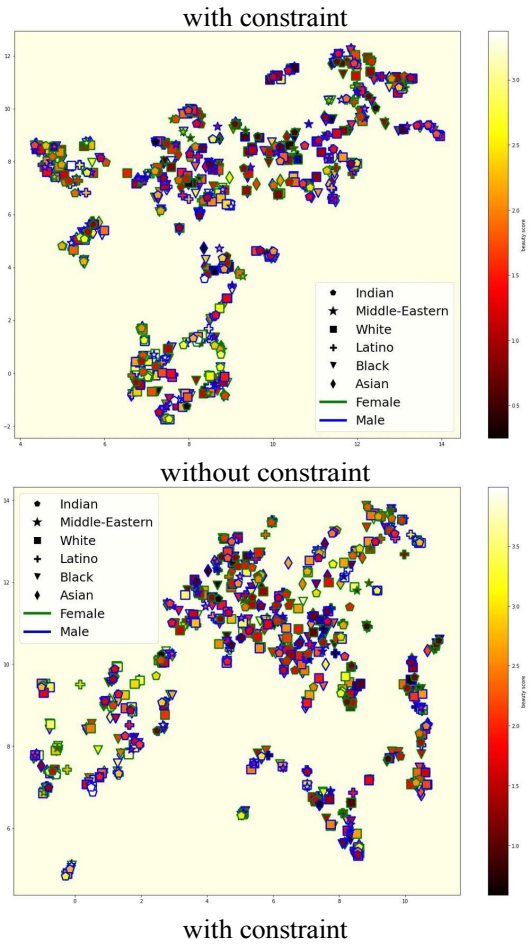


Figure 19: Structure extracted unsupervised by VAE with VGG as encoder, and 10 layers as decoder. The graph is obtained by 2-dimensional UMAP projection of activations in latent space in response to each CFD image. We compare with and without the constraint - there's nothing extracted in either case.

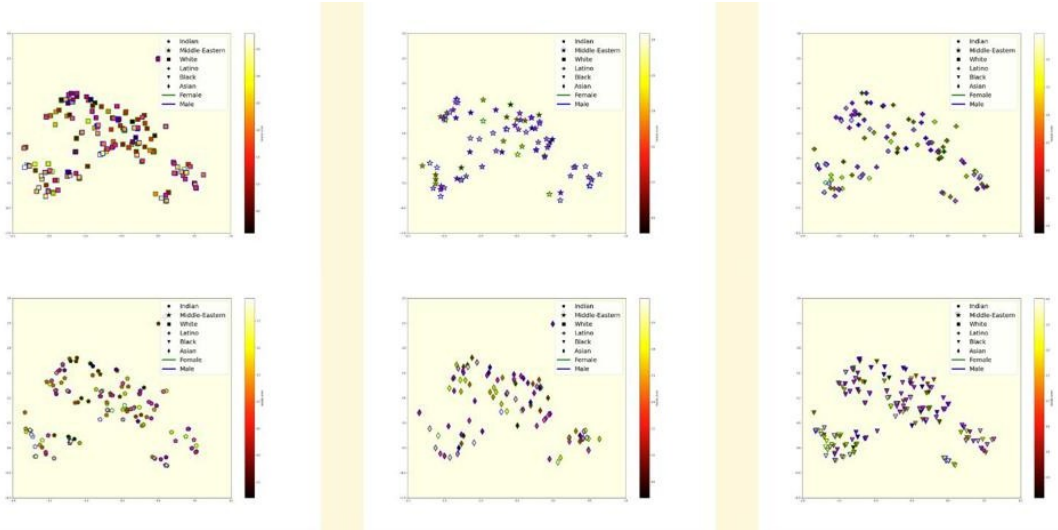


Figure 20: Using the same model and technique (without the constraint), we also check by ethnicity whether structure is extracted. This is not the case.

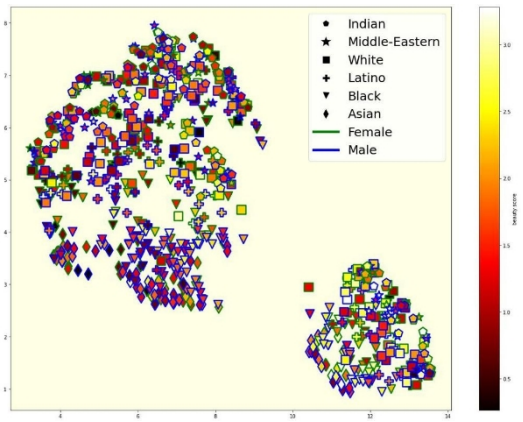


Figure 21: With a latent space 4 times smaller, this time we see a structure appear.

6 Tasks

6.1 Repeating the previous year's course

- Last year's intern, Melvin BARDIN, worked on modeling Fluence as the likelihood of CNN response to an input image: the more the CNN response is typical, the more fluent the image is considered to have been processed. He then used this measure of Fluency to predict beauty scores.

- There were a number of results missing from this course, which it was interesting to obtain my first task was to do just that. Normally, all I had to do was run a few scripts from the command line, but in reality you had to go and change things in the code, where there was no interface, so in the end you had to write extra lines and familiarize yourself with the thousands of lines of code that had been written before you could launch anything. This was very time-consuming.

6.2 Attention mechanisms

Deep Learning approach to attention:

Attention mechanisms are a major component of brain function. They enable us to "recycle" the functionality of certain brain areas, modifying their operation according to the task in hand. Attention mechanisms are also omnipresent in Deep Learning literature. In general, they provide models with an explicit means of ignoring information that is not important to them.
It would seem that there is a link, albeit imperfect, between attention mechanisms in the brain, and in artificial neural networks[27].

Experiment: Our idea was to model attention in the brain by attention in neural networks, and to see if the phenomena described by Schaeffer emerged for the most beautiful images.
Most state-of-the-art attention methods involve calculating attention maps on *the fly*, which are then applied to the image (or latent representation). It is difficult to relate these methods to Schaeffer's description, which implies cognitive feedback over time (as the symptoms of aesthetic attention are brought about by previous experience, either with the same stimulus or with another).
We chose to use the CBAM method [58] (see Figure 22) to model visual attention. In a first version, we used CBAM with depth (along feature maps) and spatial attention maps, with a sparsity constraint on spatial attention maps (see Figure 23 for results). Faced with these results, we chose to focus on in-depth attention, as it better represents the number of features to which the network had to give importance. We modified CBAM in two ways to extract these in-depth attention maps.

6.2.1 Attention to depth with sigmoid and sparsity stress (no residual)

For this first modification of CBAM, we removed spatial attention. Then, we added an L1 constraint (adding the sum of the values of the attention maps to the minimization objective) on these attention maps to induce them to be more sparse (the model is forced to be more selective in what it pays attention to). Results using this method are shown in figure 24.

6.2.2 Depth care with softmax at varying temperatures

Softmax is defined as follows: for a list of input values $a_1, a_2, a_3, \ldots$, it modifies each value as follows:

$$softmax(a_i) = \frac{exp(a_i)}{\sum_j exp(a_j)}$$

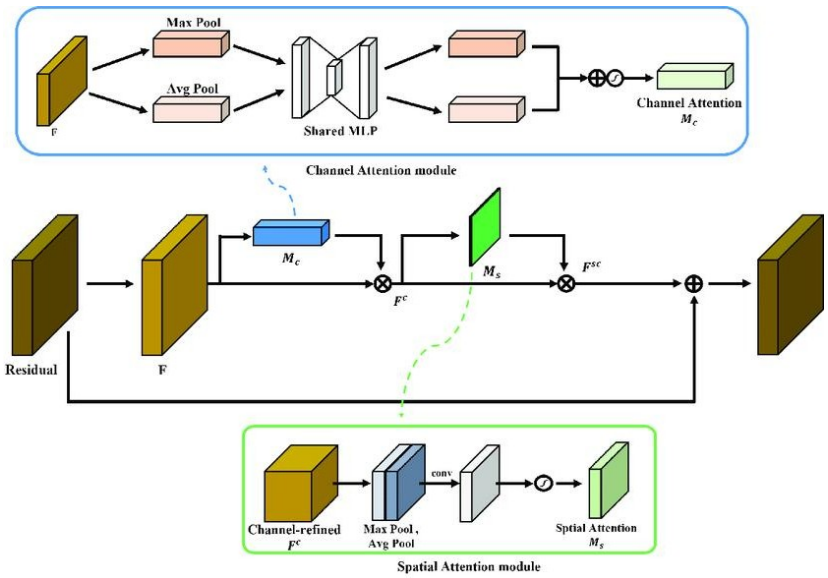


Figure 22: The essential principle of CBAM is to apply attention in depth and then attention in space. The attention scores assigned to each featuremap (in the case of depth attention) and to each pixel (in the case of spatial attention) are each calculated using pooling followed by a multi-layer perceptron (several matrix multiplications followed by a ReLU), and a sigmoid to bring the scores between 0 and 1.

Although the sigmoid in CBAM (and by extension the modified CBAM with L1 norm just described) gives values between 0 and 1, it would be interesting to use a Softmax instead of the sigmoid for the following reasons:

1. Softmax gives us activation cards whose sum is always 1, making them easier to interpret.
2. Softmax (thanks to the exponential) tends to force the sparsity of activations (hence the name *soft-max*, a derivable max operator).

A basic softmax seemed to us very constraining for the network, so we chose to add a *temperature* t to the softmax, which translates into :

$$sof\,tmax_t\,(a_i) = \frac{exp(ai/t)}{\sum_j exp(aj/t)}$$

In addition, we have chosen to put the softmax and attention maps only on certain layers of the network. Figure 25 shows the training results for different values of t and different layers on which to place attention.

6.2.3 Limits of our approach to attention :

Methods that calculate attention on the *fly* are not sufficient to capture biological attention. To improve on this, a global (like [54]) and temporal dimension should be added.

For the moment, our attention measure is rather similar to other approaches [46] for predicting image complexity by calculating the importance the model assigns to different features.

6.3 Specialization of visual models to ethnicity and gender

6.3.1 Origin of the idea

Last year's trainee (who measured fluency as the typicality of CNN activations in response to the stimulus) had tried to specialize his activations distributions (in which typicality is calculated as likelihood) on ethnic/gender groups. Immediately, his fluency measure became a better predictor of beauty.

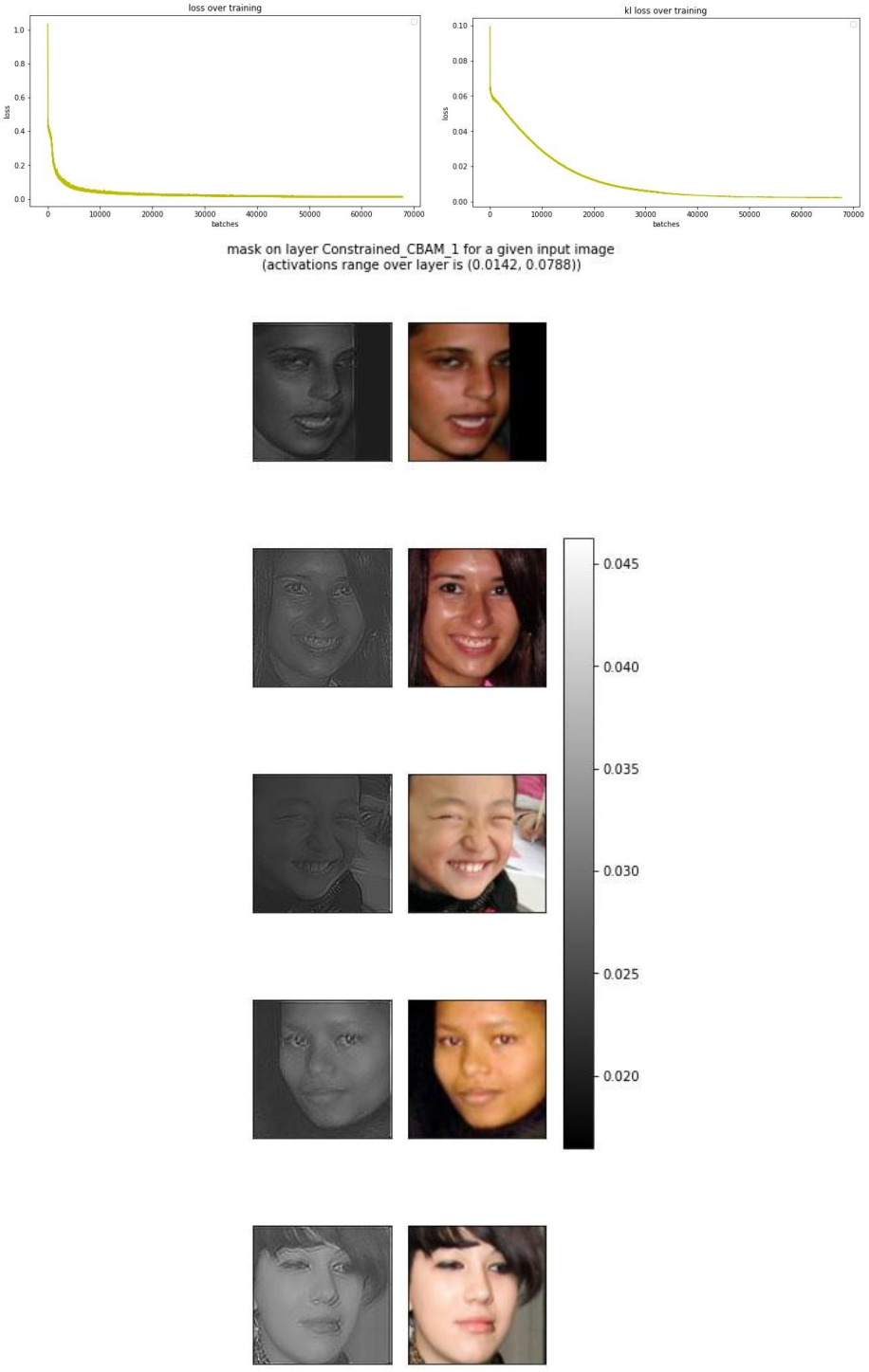


Figure 23: results with full CBAM with just a sparsity constraint on spatial featuremaps. In black and white, we see the spatial attention maps extracted for each image. These are not very interpretable, so we have chosen to use only depth attention. We also show the value of the cost functions (left reconstruction, and sparsity constraint on latent space) during training.

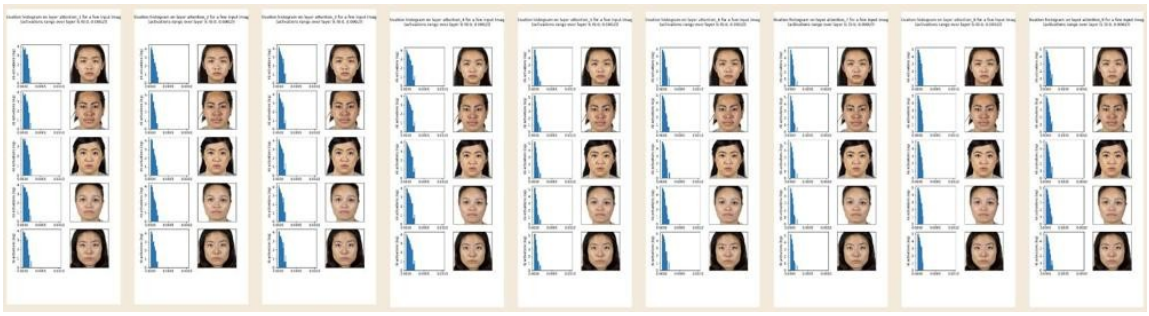


Figure 24: Results with just attention in depth and a sparsity constraint on it: we see that the sparsity constraint has pulled the histograms of activations towards 0, and the reconstructions are a little worse. Here we've used attention on 10 layers, and the histograms of attention values are given. As the reconstructions do not vary by layer, they are the same each time.

6.3.2 Biological interpretation

The interpretation of this phenomenon was that our perception of beauty would be conditional on the category in which we place the subject. Thus, the fluency of information processing would be influenced by an expectation towards the stimulus due to the latter's category, and by specifying the visual model according to this category, we better model fluency and by extension beauty.

6.3.3 Installation

We wanted to test this visual model specialization approach on our method. To do this, we trained a VAE per subgroup of the ethnicity*gender space, which we consider to be the model of the visual system in response to a category of images.

It should be noted that this approach is somewhat less relevant, since we are not comparing activations to a distribution representing **the** visual system's **expectation of** certain images. Nevertheless, our specialized models could represent the "path" that information takes in the brain conditional on the nature of the perceived object, or a direct modification of the visual system by cognitive feedback (attention or predictive coding [32]) that dynamically adapts it to the type of stimulus detected.

We don't yet have the results of training our models on each sub-class, as the models have been trained on Jean Zay, but we still need to generate the figures.

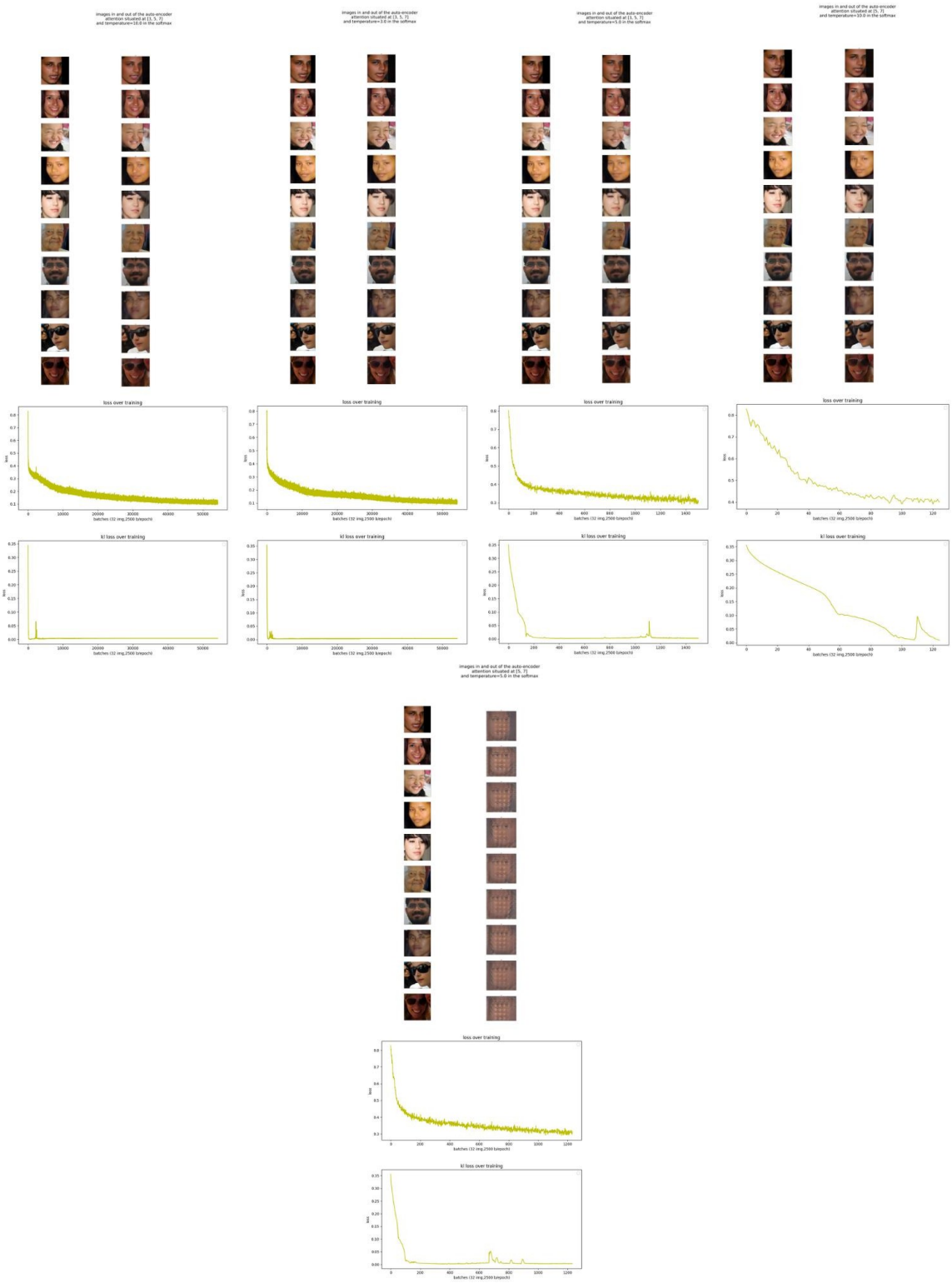


Figure 25: Reconstructions and cost functions (reconstruction above, and sparsity constraint on latent space below) with depth attention with softmax, located during training at different layers, and with different temperatures.
For many configurations not shown here, loss was NaN after a few epochs (as in the 57-5.0 case here, where we see that reconstructions are poor and the cost function stagnates above 0.3).

7 Additions to the method

7.1 Reconstruction attempts without decoder

7.1.1 Motivation:

As explained above, we aim to model neural processing fluency by two metrics: economy, and information processing efficiency. In particular, for neural processing efficiency, we want to know how much important information the encoder (our model of the brain) has managed to extract. One problem with measuring this with the recon- struction error is the bias introduced by the decoder. It would be interesting to have a way of obtaining reconstructions without introducing a decoder into the balance.

7.1.2 The method :

The method we have adopted for reconstructing an image processed by the VAE encoder without using the decoder is as follows:

1. Pass the input image X through the encoder to obtain the latent representation Z_X.
2. Pass a noise image N through the encoder to obtain the latent representation Z_N.
3. Calculate the distance L2 between (Z_X and Z_N), denoted D. Using PyTorch's self-differentiation, calculate $\frac{\delta D}{\delta N}$.
4. Modify N so that its latent representation approximates that of X: $N \leftarrow N - \lambda * \frac{\delta D}{\delta N}$.
5. Repeat steps 2 to 4 until D converges.

7.1.3 The problem :

In practice, when we implement this method, we obtain the results shown in Figure 26. D becomes very small, indicating that the representation of N becomes the same as that of X in the VAE latent space. Nevertheless, in the image space, N remains very different from X. What causes this behavior?

7.1.4 Adversarial examples:

One of the biggest problems with CNNs is their gigantic size, more than enough to learn the entire training database by heart. A related problem is that CNNs rely on imperceptible patterns in input images to make their decisions. If we artificially introduce these patterns into an image, as they are imperceptible, we can end up with a CNN's interpretation of the image completely different from that of a human. For example, an image of a cat can be altered in a way that is imperceptible to the human eye, so that a CNN believes it to be a helicopter. Such an image is called an *adversarial example* [10]. In the case of an Autoencoder, this misinterpretation of the image results in what is shown in figure 27.

Clever Hans:

To better explain the problem of adversarial examples, Ian Goodfellow compares this to the case of the donkey "Clever Hans". *Clever Hans was a donkey who seemed to be able to count, for when he was put on stage and told a basic arithmetical calculation, he would clap his hoof for the number of strokes corresponding to the calculation's solution. However, one day when the calculation was whispered in his ear, he was suddenly unable to solve it. Clever Hans' strategy became apparent: he tapped his hoof until the audience held its breath, indicating that he had arrived at the correct number of strokes.* Clever Hans relied on an accidental cue to perform well, completely ignoring the nature of the problem at hand. Are adversarial examples proof that CNNs behave in the same way? There is no consensus on this. For example, in [20], the authors show that these imperceptible patterns on which CNNs are based are the same from one database to another, indicating that they are not so *accidental*.

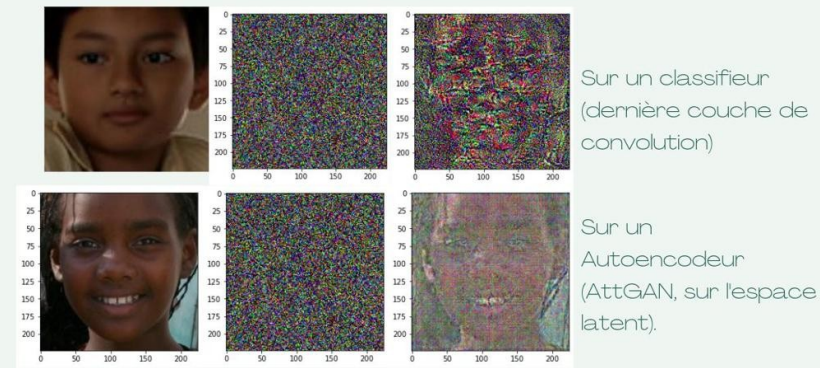


Figure 26: reconstruction by gradient descent, on VGGface and AttGAN - for each line, in order: original, starting point, image after convergence. It is interesting to note that the reconstructions, although insufficient, are better for an autoencoder than for a classifier, indicating a more complete latent representation.

In any case, in our case, this flaw in the CNNs makes our decoderless reconstruction method a failure.

7.2 Use of a sparsity constraint

One of the tasks carried out during the internship was to implement a constraint on the model's training to force its activations to be more sparse.

7.2.1 Biological motivation:

Barlow's Efficient Coding Hypothesis [1] states that the brain, under a constraint of energy expenditure, would tend to encode information sparsely. This is not the case for our CNNs. It has been shown [35] that adding an L1 constraint (by adding the L1 norm of activations to the minimization objective) brings out filters similar to those found in the visual cortex. Since we're interested in modeling the brain through CNNs, it's interesting to add such a constraint to potentially amplify the resemblance to the brain. In addition, the L_ϵ standard we use as a measure of sparsity requires that the activation distributions be constrained around 0 and with the left tail much shorter than the right tail.

7.2.2 The constraint in place

We have implemented the constraint of minimizing the Kullback-Leibler distance between the distribution of activations, and a localized uniform distribution over a value ρ (constraint taken directly from [37]). Specifically, we add to the minimization objective the value

$$\sum_n (\rho log(\frac{\rho}{\rho_n}) + (1 - \rho) log(\frac{(1-\rho)}{(1-\rho_n)}))$$

Where n corresponds to a neuron (we loop over all the neurons in a layer), ρ to the uniform law localization, and ρ_n to the average activation of neuron n for a certain number of input images (in our case 32 in general). In addition, we use a parameter β by which we multiply this constraint in the minimization objective, in order to give it more or less importance (if β is very large, we will minimize the sparsity constraint more than the reconstruction error, and vice versa).

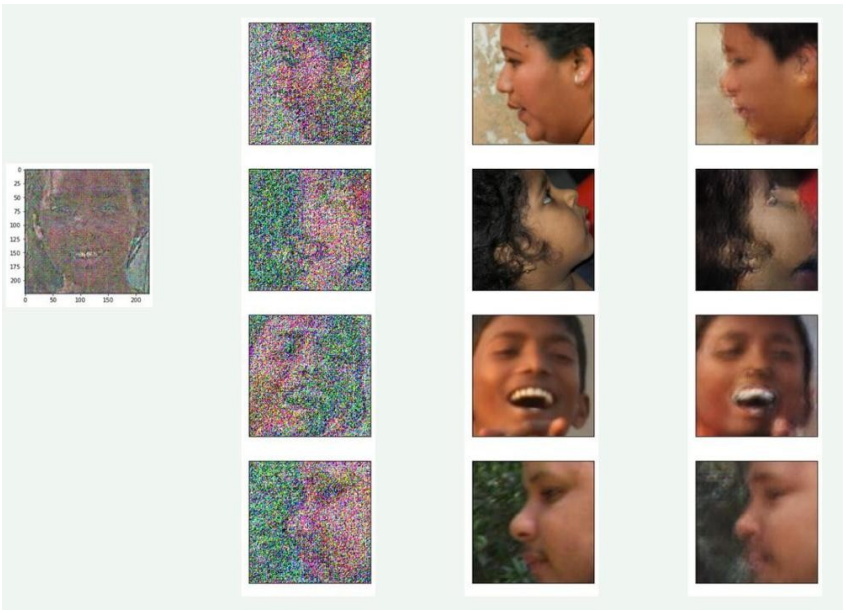


Figure 27: Demonstration of adversarial example: left: recall of reconstructions with attGAN. Second column: decoderless reconstruction with VAE. Third column: original images. Last column: VAE reconstruction of images in second column.

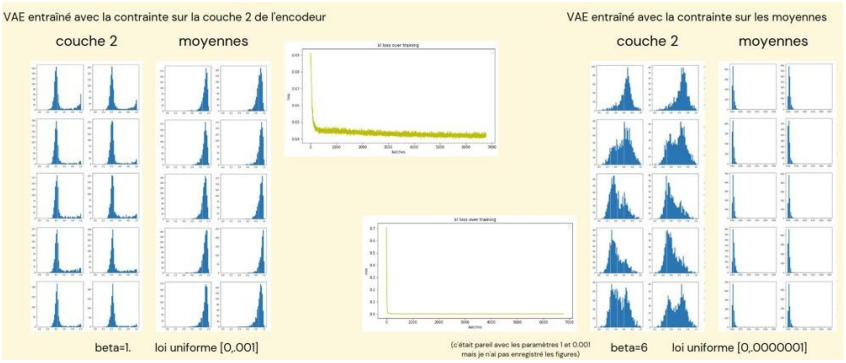
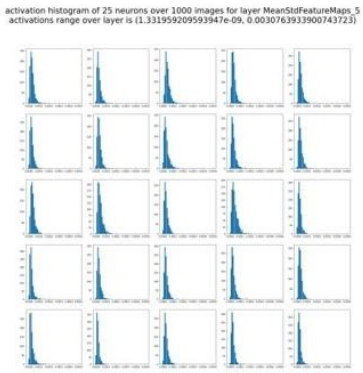
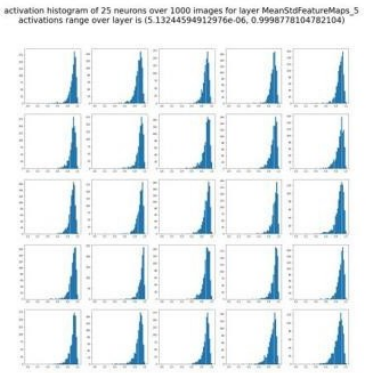


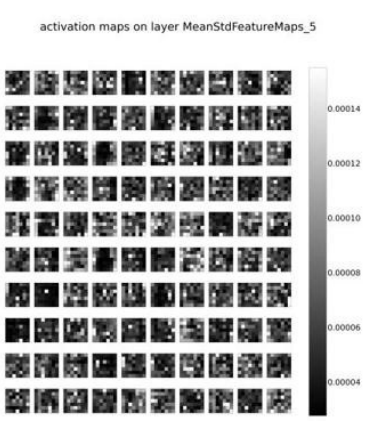
Figure 28: The histograms shown are the activation frequency histograms for all CFD images, for a single neuron each time. We can see here that the constraint is having an effect (distributions tend towards 0) on the averages layer, but not on layer 2 (on the small model). This was in fact due to the Normalization Instance, which was located on the first 4 layers and prevented the constraint from taking effect. This seems difficult to explain.



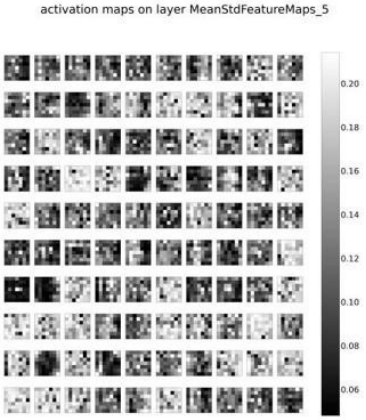
(a) histogramme des activations sur les moyennes



(b) histogramme des activations sur les moyennes (entraînement sans la contrainte)



(c) featuremaps des activations sur les moyennes



(d) featuremaps des activations sur les moyennes (entraînement sans la contrainte)

Figure 29: Effect of constraint on activations: it greatly reduces amplitude by pulling distributions towards a constant, but also increases sparsity. In terms of Featuremaps, with constraint we see less white and more black, indicating that activations are more localized.

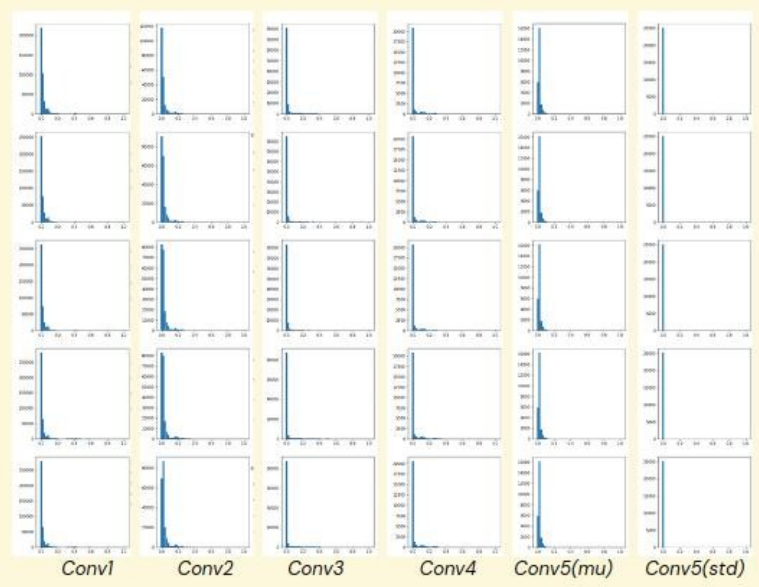


Figure 30: Histograms of neural activations on all CFD images, on the model with 5 layers at the decoder and encoder, without the normalization instance, and with the sparsity constraint. We see that the constraint works to bring activations back towards 0, but that it doesn't create any activation/non-activation separation quantization (as we'd hoped).

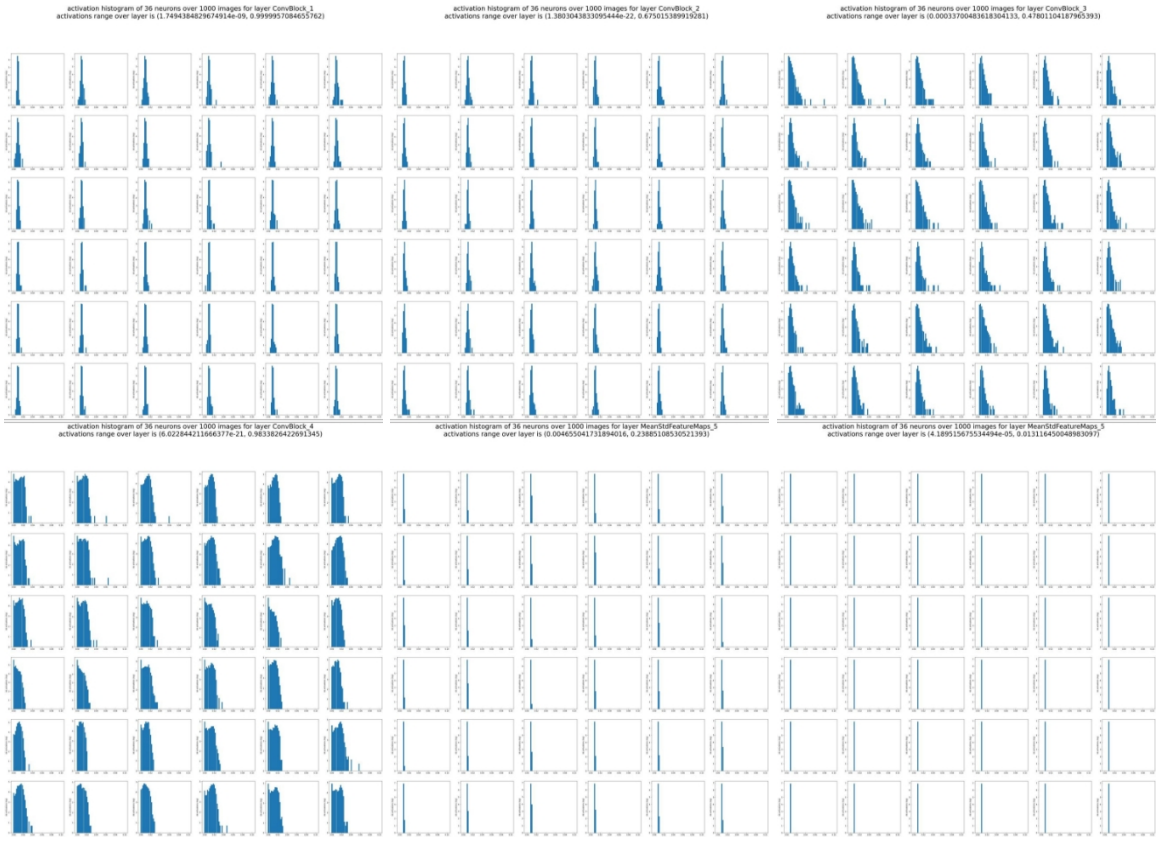


Figure 31: Histograms of activations on the small model when the constraint is 0.000001. We show histograms of activations for 25 neurons, (5x5 matrix), on all CFD images. These are activation frequency histograms for all CFD images, for a single neuron each time. We see that bringing the uniform distribution closer to 0 has no positive effect on the distributions. There are 6 matrices of 25 activations, to be read in the reading direction: layer 1, 2, 3, 4, then averages and finally standard deviation.

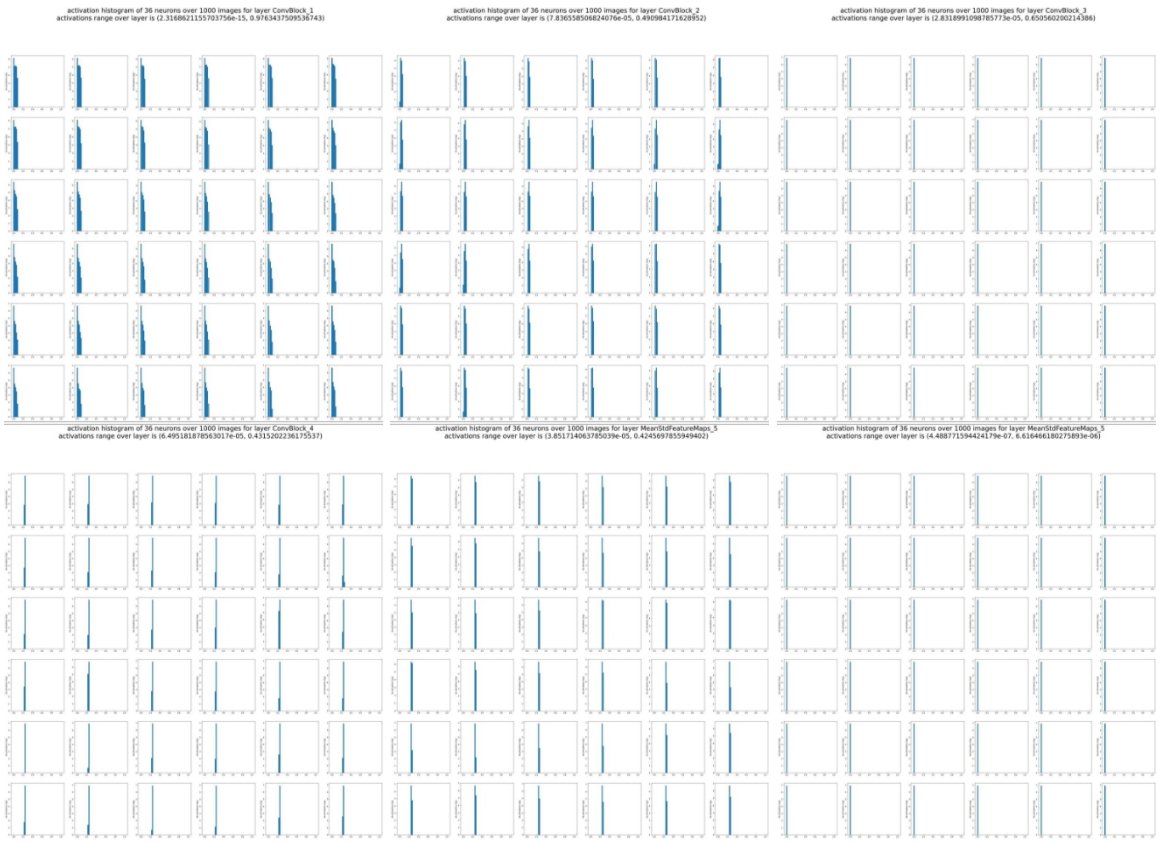


Figure 32: same as figure 31 when the constraint is multiplied by 3 in the final cost function. We still don't see the expected separation.

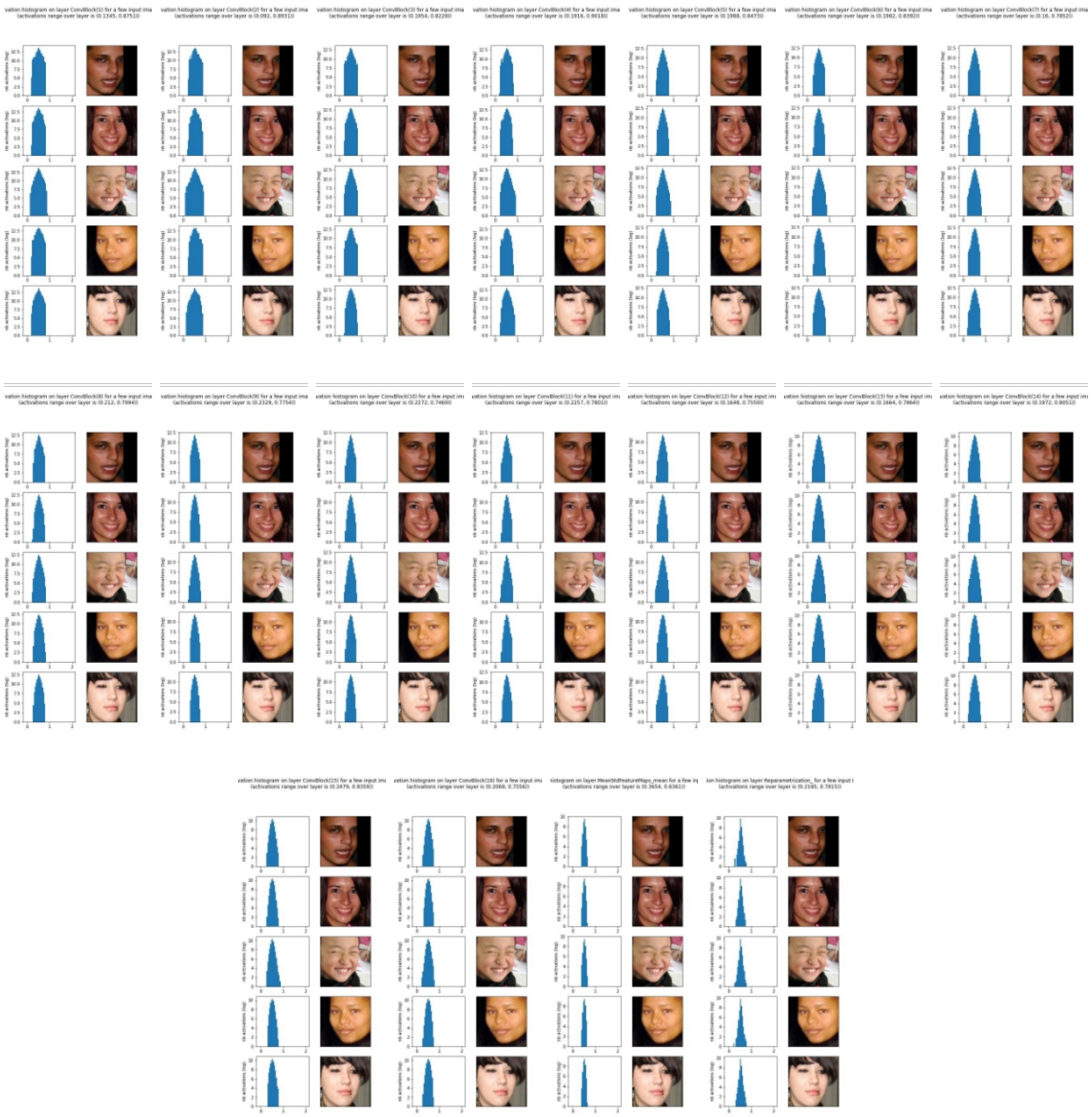


Figure 33: Histograms (this time histograms for the entire layer, on a single image, instead of on a single neuron for all images) on the VAE with VGG in the encoder and 10 layers in the decoder, without the constraint. There are 5 histograms per encoder layer, and they are given in layer order here.

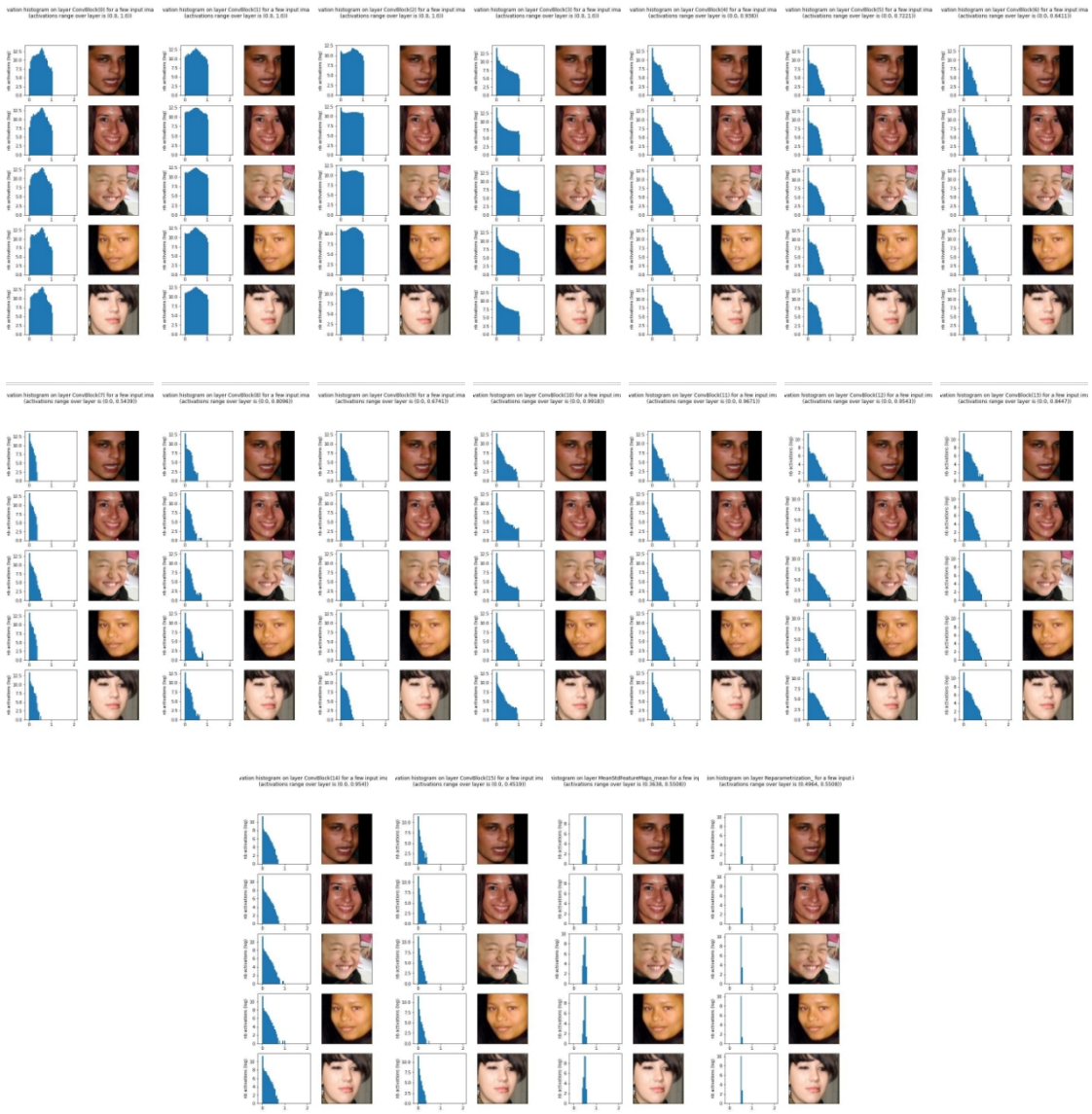


Figure 34: Histograms on the VAE with VGG at the encoder and 10 layers at the decoder, with the constraint. It seems that the constraint works a little less well here, perhaps because the encoder, being longer, needs to retain more information, and can afford less sparsity. Here too, the histograms are given in the order of the encoder layers.

8 Abandoned tracks

8.1 Abandoned tracks

Here we briefly present a number of avenues that we considered in greater or lesser depth (sometimes even tested) during the course, only to abandon them in the end, either for lack of time or because they didn't work.

8.1.1 Spectral normalization

Spectral normalization idea

As the gradient descent algorithm relies on the validity of the gradient as the local direction of maximum increase in the cost function, it would be interesting to be able to ensure that the gradient is stable and not too large. One way of formulating this is :

$$\|L(x) - L(y)\| \leq K * \|x - y\|$$

Where x and y are any vectors of real numbers, and K is a constant. If L satisfies this condition, it is said to be K-*Lipschitz*. The Spectral Normalization heuristic is as follows: *if the cost function is 1-Lipschitz, the training will be more stable.*

How it works

The *spectral norm* of a function is the maximum value of all singular values of its Jacobian at all points in space. In practice, we will apply the spectral normalization in-dividually to each layer of our model, without the activation function, which therefore represents a linear application that will have the same Jacobian for all points in space.
Consider the following two results:

- The spectral norm N_s of a linear application is equal to its Lipschitz constant (i.e. the application is N_s -Lipschitz). (proof not provided)
- It is quite intuitive that the spectral norm of the linear application multiplied by a scalar λ is equal to λ times its spectral norm.

We can then conceive that by calculating the spectral norm of the layers in our network and dividing the weights by it, we'll have 1-Lipschitz layers. Since we're using ReLUs as activation functions, since the ReLU is 1-Lipschitz, and since the composition of 1-Lipschitz functions is also 1-Lipschitz, by performing this operation of dividing the weights by their spectral norm, we render the entire network 1-Lipschitz.
The final element is how to calculate the spectral norm. For this, *the iterated power method* is commonly used.

Usefulness

After its invention, Spectral Normalization was first applied to Generative Antagonistic Neural Networks [9], where it was used to diversify the images generated by the generator. We have used this technique in our VAE (applying normalization to all layers) because the authors of nVAE [53] claim that it is useful for VAEs. Nevertheless, as can be seen in Figure 35, in our case it made no difference.

8.1.2 Other sparsity constraints

Here, we have chosen a sparsity constraint from a number of possibilities. In particular, a constraint designed for the sparsification of neural networks in order to com- press them [29], and an L1 norm [35] as discussed above, were considered as alternatives during the course.

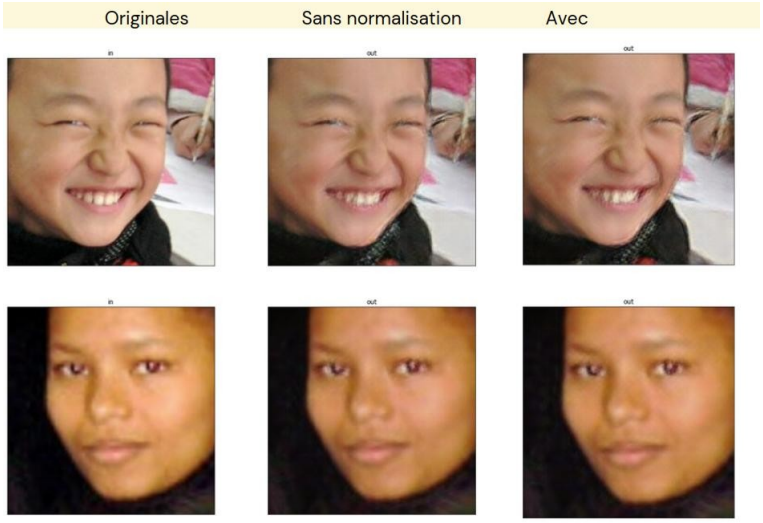


Figure 35: Reconstructions with spectral normalization are neither better nor worse than without (small model here).

8.1.3 Generative Invertible Flows

Generative Invertible Flows [24] is a generative latent variable model (like VAE) whose principle is to create an invertible transformation to the latent space. Inversion is not achieved by means of a decoder, but by means of the properties of the operations used to pass from the input to the latent space (see figure 36 for results of this type of model).
Generative Invertible Flows would have been an interesting alternative to VAEs insofar as they would allow us to bypass the inherent bias of the decoder. However, in the course of our research, we came across a number of results that deterred us from using them:

- The invertible operations used are a far cry from the convolution operations of our VAEs, which would take us further away from our biological model.
- Latent space tends to have a high dimensionality in Generative Invertible Flows compared to VAEs (so potentially less information extracted explicitly).
  (see figure 21 compared with 19)).

8.1.4 Mutual information

The principle of mutual information :
The mutual information between two random variables represents the extent to which one distribution depends probabilistically on the other. You can get an intuitive idea by looking at the formula. The mutual information between the random variables X and Y is :

$$I(X;Y) = \sum_{x,y} P(x,y) * log(\frac{P(x,y)}{P(x)P(y)})$$

We see that mutual information is in fact the Kullback-Leibler divergence between the joint distribution of the two variables, and the product of the two marginal distributions. Remembering that if two events are independent, we have $P(x, y) = P(x)P(y)$, we can see that mutual information will increase the more x and y are dependent events.

Usefulness in our case :
With the same aim of measuring the efficiency of the encoder's information processing without the need for a decoder, we wanted to use the mutual information between the input image and the latent representations. An additional advantage of this method would have been to be able to obtain a measure for each layer rather than having a single reconstruction for the whole model. Having this mutual information measure between the latent representation at the output of each layer, and

Figure 36: Samples from the Glow model (not reconstructions, but artificial samples randomly drawn from latent space).

the input image, would have enabled different analyses of the effect of different layers on the perception of beauty.

Problem:
Since mutual information is calculated between two random variables, it is not possible to calculate it on a single image (several images and several corresponding latent representations are needed to create distrubutions). Even with multiple samples, the estimators are not perfect [50]. For our purposes, we would have needed one measurement per image, so we abandoned this avenue.

8.1.5 Depth-separated convolutions

The principle: In image processing, two-dimensional convolution filters are often used to smooth images or extract contours. One way of optimizing the convolution operation is to express the filter as the outer product between two vectors. For example, the basic edge detection filter

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

can be written as :

$$\begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}$$

The convolution operation with one of the small vectors and then with the other is equivalent to convolution with the whole filter, but takes up less memory and less computing time.
Depth Separable Convolutions is a technique from the MobileNet architecture [17], which involves expressing the three-dimensional filters used in our convolution layers (height, width, features) as a convolution operation with a two-dimensional filter (height,width) and a one-dimensional filter (features) (see 37). The advantage is a sharp reduction in computation time and memory requirements. The price to pay is that separable filters are limited in terms of expressiveness, but in practice this doesn't seem to bother us much.

In nVAE [53], the authors recommend the use of depth-separable convolutions, as they allow the size of convolution filters to be increased without taking up more memory space. We have tested this technique (which enabled us to go from 3x3 to 7x7 filters), but it does not improve the results in our case (see figure 38).

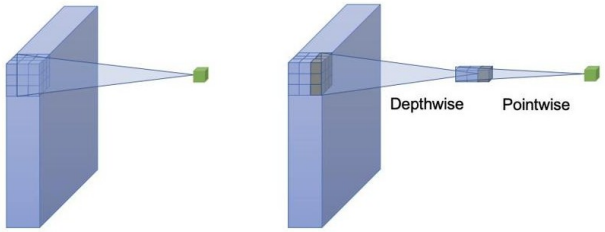


Figure 3: Standard convolution and depthwise separable convolution.

Figure 37: The principle of depth-separable convolutions is to transform a convolution with a 3D filter (left) into a convolution with a 2d filter followed by a convolution with a 1D filter (right).

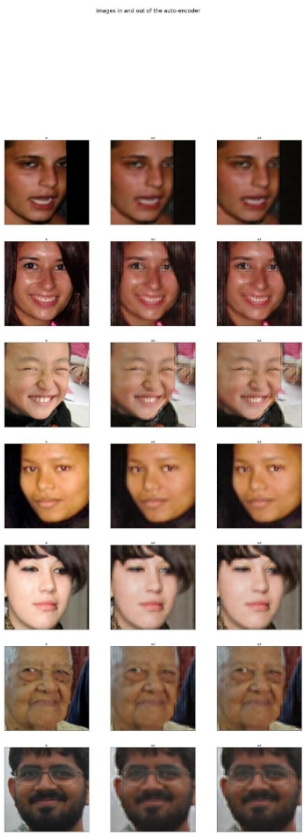


Figure 38: Reconstructions of the small model with filters of size 7x7 instead of 3x3, taking up the same memory space, thanks to depth-separable convolutions (it's worse).

8.1.6 Tracks for reconstruction without a decoder

After the first month and a half of the course, we had abandoned the idea of reconstructing images without the decoder, by optimizing an image so that it had the same latent representation as the target image. During the remainder of the course, we came across a number of possible improvements to this technique, which it would have been interesting to explore (in particular, perhaps a combination of these techniques could have greatly improved the reconstructions).

Anti-adversarial approaches

As adversarial examples can present a security problem in Deep Learning-based applications, various methods have been created to combat this weakness.
It is possible, for example, to generate adversarial examples *during training* and teach the network to nevertheless classify them correctly. Databases augmented with adversarial examples are available [14].
An approach that seems more elegant and a little less *ad-hoc* is the *Local Winner Takes All* (LWTA[36]) approach, where a mechanism of local competition between CNN neurons, inspired by the way the brain works, is added, giving it robustness to adversarial examples without the need to add any in the training base.

Distribution output detection

In VAE, a recurring problem is that VAEs will create a distribution of images from the training base in latent space, and place images in the middle of this distribution that have nothing to do with it (for example, an image of Gaussian noise ends up right next to an image of a face). This could be a good explanation of what happens when we try to do a reconstruction without the decoder. There are methods for making the latent space robust to these "distribution outputs" [6] [40].

Restricting image space :

Another way of approaching the problem is in image space rather than in latent space. The reason why our decoderless optimization results in non-realistic images is that our encoder is not injective: for the same latent representation, there are several input images. Only a subset of these images will be realistic. If we could add an a-priori (e.g. in the form of a constraint on the optimized image) on the image space that restricts it to realistic images, our optimization should result in better reconstructions. The best-known version of this method is DeepDream [33] (see figure 39).

Firstly, it's doubtful that we'd get the perfect reconstructions we need, and secondly, with this approach, we'd be taking the risk of introducing a new bias into the a priori, and thus getting rid of the decoder bias to introduce a new bias elsewhere.

Figure 39: Deep dream makes an adversarial attack with gradient descent, only with an apriori on the image space. What would this have looked like on an autoencoder, and one that specializes in faces?

9 Results

9.1 Beauty-related statistical analysis

9.1.1 Correlations

List of metrics

Over the course of the course, various avenues were explored and different fluency metrics developed. Here's a list:

- SAM acuity
- The LPIPS reconstruction error
- The gini index of layer activation
- The L_ϵ standard for layer activation (insofar as the constraint was applied to the drive)
- The L1 standard for attention cards.

Expectations

We explain here, in the context of our topic on fluency and beauty, what our expectation is of the correlation that each metric should have with beauty scores on CFD.

- SAM acuity: as we use it as a measure of the point at which the network has *understood* the image, it would be a measure of information processing efficiency and therefore positively correlated to beauty scores.
- The LPIPS reconstruction error: this allows us to test the quality of the latent representation extracted by the encoder, by checking whether the decoder reconstructs the image correctly. Thus, it is a measure of information processing efficiency that should be positively correlated with beauty scores.
- The gini index of layer activations: we use this as a measure of the economy of effort during information processing. A high gini index indicates high sparsity, and easy information processing. It would be expected that the gini index would be positively correlated with beauty scores.
- The L norm$_\epsilon$ of layer activations (insofar as the constraint was applied to training): the higher the L norm$_\epsilon$, the more neurons are considered to have activated, and therefore the greater the information processing effort. The norm L_ϵ would be therefore negatively correlated with beauty scores within the framework of Fluency theory.
- The L1 standard for attention maps: since it measures the amount of different features the model had to take into account to process the information, a high value would indicate, in the case of the Fluence, an image that's difficult to process. This metric would therefore be negatively correlated with beauty.

Method

To obtain the correlations between beauty scores and metrics, we ran each of the images in the CFD database through the encoders of our various networks, keeping track of the intermediate representations at the output of each layer. We calculated the gini and L metrics$_\epsilon$ on these intermediate representations. Next, we obtained the reconstructions and calculated the LPIPS value, and finally the SAM acuity. For models where we had added attention me- canisms, we extracted the attention maps and calculated their L1 norm.

In this way, we obtained as many values for each metric as there were beauty values in the CFD database. We could then calculate a correlation between the vector of each metric and the vector of beauty scores.

In the case of gini, L_ϵ and attention maps, we could calculate a correlation for each latent representation.

We also looked at the P-values of our correlations (see figure 40 for an example), and in general they are less than .05 when the correlation has a value greater than .1.
In addition, we looked at the Spearman correlations (Pearson correlation after replacing the values by their index in the sorted list of values), to check that we weren't missing non-linear relationships we looked at the Spearman correlations each time at the same time as the Spearman correlations. We didn't find any additional information this way.
Finally, as Spearman correlation can also miss certain relationships (for example, a bivariate distribution that forms a circle in space will have a Spearman correlation of 0 between the two axes), we also looked at graphs of the joint distributions between metric values, and beauty scores (see figure 45 for an example). Once again, we found nothing special in this way.

Results

Figure 47 shows Pearson correlations between activation gini index and beauty scores, for all encoder layers of the VGG19 model with 10 layers at the decoder, and with an attention mechanism (the softmax version) on layer 5.

Figure 48 shows the correlation between the gini index calculated on all layers at once, as well as the SAM acuity measure, and the beauty scores on CFD (on the VGG19 model with 10 layers at the decoder).

Figure 49 shows the Pearson correlations between the gini and L metrics$_\epsilon$, and the beauty scores, for the 5 layers of the small model encoder. The correlation between beauty scores and reconstruction error is also given. Two versions of each metric are shown: one on a model trained with the constraint, and another with a model trained without.

Figure 50 shows the same thing as figure 49, but without the reconstruction error (this model didn't reconstruct well enough), and on the VGG19 model with 5 layers at the decoder. The layers are arranged in reading order (top left, layer 1, next to it, layer 2, etc.).

Interpretation
It seems that correlations are generally quite low. Even among those above 0.1, it's difficult to detect any particular pattern. Here we review the metrics and comment on their effect.

- The LPIPS reconstruction error: this metric, too, seems to give no
- SAM acuity: this metric seems to give no information on beauty scores. Since it's a measure of the acuity of the reconstruction error, we could have used it as an indicator of beauty scores. wait.
- The gini index of layer activations: Across the three models, it's hard to say whether it's generally positively or negatively correlated with beauty. One thing potentially interesting is that on the small model and on the VGG19 model with 5 layers at the decoder, it would appear that the sign of the correlation alternates systematically from one layer to the next. However, it's hard to say what this would mean in the context of Fluency theory. The gini index on all layers at once, on the
- The L_ϵ norm for layer activations: as with the gini index, there is no tendency to be positively or negatively correlated, even though this pattern is also present. from one layer to the next.
- Standard L1 attention cards: do not seem to carry any information on beauty.

Effect of constraint: *On vgg19 with 5 layers at the decoder*: the presence of training constraint seems to reduce the correlation between gini and beauty scores.

On the small model: The constraint changes the correlation completely. There are more correlations with than without the constraint, but with the constraint, the correlations are negative, which is the opposite of our expectation. We did not train the VGG19 model with 10 layers at the decoder with the constraint. Effect of different models: The vgg with 10 layers at the decoder seems to stand out from the others in its correlation patterns. Perhaps this is caused by the addition of the attention mechanism.

9.1.2 Linear models

Method

Correlations let us know whether our metrics and beauty tend to vary together. However, they do not allow us to directly verify their ability to predict beauty scores. It would indeed be interesting to know, among the variations between beauty scores for different images, what proportion could be predicted thanks to our metrics. For this purpose, we use linear models.
A linear model is a set of weights that form a linear combination of vectors, the result of which is as close as possible to a target vector. The vectors in the combination are formed from the samples of our *predictor* (or independent) *variables*, and the target vector is formed from the samples of our *target* (or dependent) variable. In more compact form, a linear model consists of a set of weights $\theta_1, \theta_2, \ldots, \theta_n$ such that the distance between

$$\theta_1 * V_1 + \theta_2 * V_2 + \ldots + \theta_n \text{ and } V_c$$

will be minimal (where V_c is the target vector, and $V_1, V_2, \ldots, V_{n-1}$ are the predictive vectors).
If $V\hat{c}$ is the vector produced by our linear model, we measure the predictive power of our model linear as follows :

- $SR = \Sigma_i (V\hat{c} - V_c)^2$ is a measure of the variability between predictions and the true target vector.
- $SS = \Sigma_i (V_c - \bar{V}_c)^2$ is a measure of the variability within the true target vector.
- $\frac{SR}{SS}$ is then the proportion of variability between beauty scores that is not explained by the predictions.
- $R^2 = 1 - \frac{SR}{SS}$ is the proportion of variability between beauty scores that is explained by the predictions.

We use this $R\text{-}score^2$ to measure the ability of metrics to explain beauty.
The metrics we're interested in here are those that can be extracted per layer: it's not very interesting to do a linear model on just reconstruction error, or just SAM acuity. So we train our linear models on the Gini, L_ϵ and L1 norm metrics of the attention maps.

Expectations

If our metrics are good predictors of beauty, which would be in line with the Fluency theory hypothesis, we would expect to have higher R^2 scores. For example, in [43], an R^2 score of 0.17 was found between beauty scores and a measure of fluency. In Melvin BARDIN's work (internship last year), on different CFD subclasses, scores ranged from 0.04 to 0.3.

Results

Figure 52 shows, for the small model, the average scores on 10 linear models trained for different versions of the small model, and the L-epsilon and gini metrics (there are therefore 5 independent vectors, for the 5 layers of the encoder).
For attention metrics, when we train the VGG19 model with 10 layers at the decoder, and attention maps after each layer between layer 4 and 9 of the encoder, and use them as independent vectors in our linear model, we find an R^2 of around 0.108 with the first version of attention that has only deep attention and an L1 constraint and 0.0942 with the second that had a softmax.

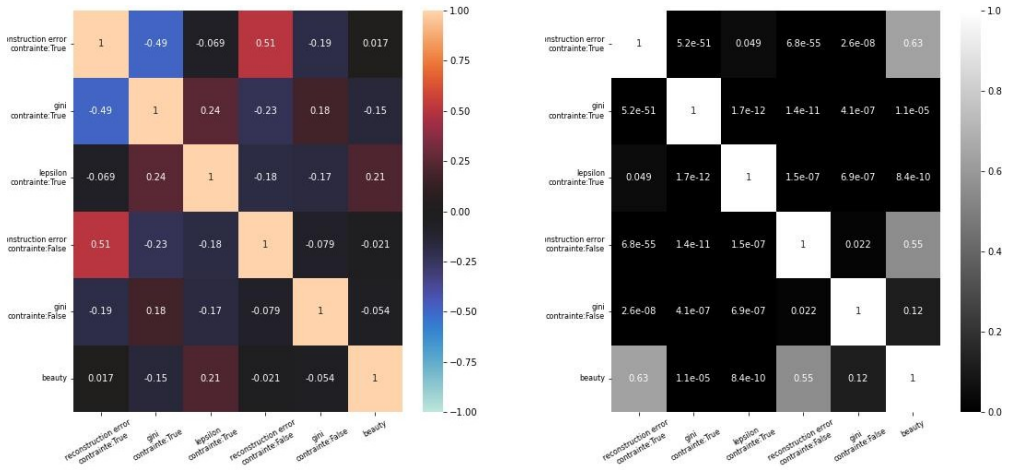


Figure 40: For each correlation matrix, we also look at the Pvalues matrix to ensure that the results are significant.

Interpretation

It would appear that our metrics do not contain much information on beauty scores. Several hypotheses can be put forward as a result:

- The networks were not good models of the brain,
- Metrics are not advised as measures of fluency,
- The databases do not measure the type of beauty that would be predicted by fluency,
- Fluency predicts about 10 percent of beauty in general.

Conclusion

Conclusion on results

To conclude, in this internship we established a methodology to study the link between fluency metrics and beauty scores, using autoencoders as brain models. The main tasks were the development of 5 different metrics, testing the use of a sparsity constraint, attempting to dispense with the decoder, creating and training a new VAE architecture, and statistical analysis. The internship's contribution is primarily methodological, given the statistical results: we didn't find a very clear pattern despite correlations with pvalues below 01, but the study wasn't exhaustive enough to reject the hypothesis. Thus, the method developed will be re-used in the future and the objective of the last month of the internship is to develop, in collaboration with people who could use it, a practical interface for training models on new databases, and extracting metrics. It would be interesting, for example, to make videos showing how to use the interface, or to add documentation using the Sphinx tool.

Skills developed

- More experience in Pytorch implementation and training
- Better understanding of the theoretical side of Deep Learning
- Introduction to a wide range of previously unknown DL techniques

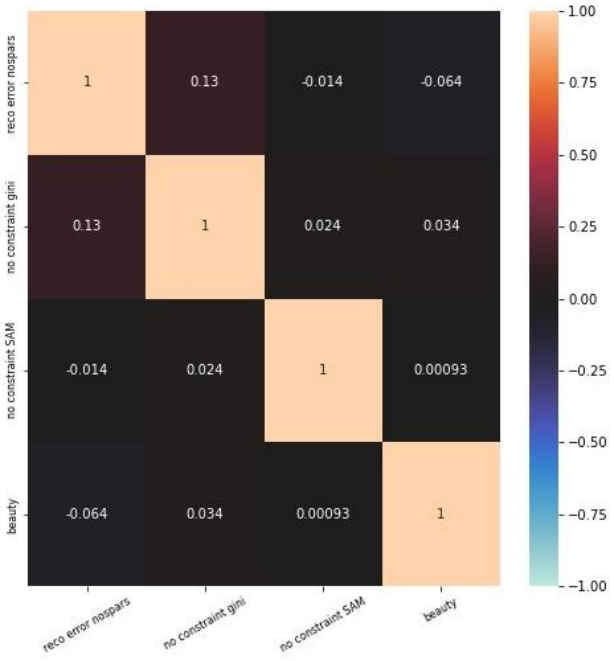


Figure 41: Correlation of beauty (last row of the matrix) with the SAM and gini metrics on all layers of the small model. The correlations are almost zero, so there is no link between these metrics and the beauty score.

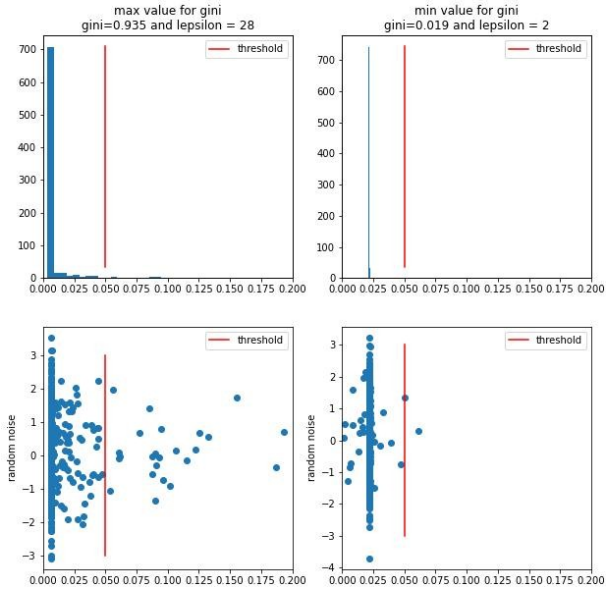


Figure 42: Counter-intuitive behaviour of the gini and L metrics$_\epsilon$: in some cases, gini and L are the same metric.
L_ϵ can be positively correlated, which is counter-intuitive.

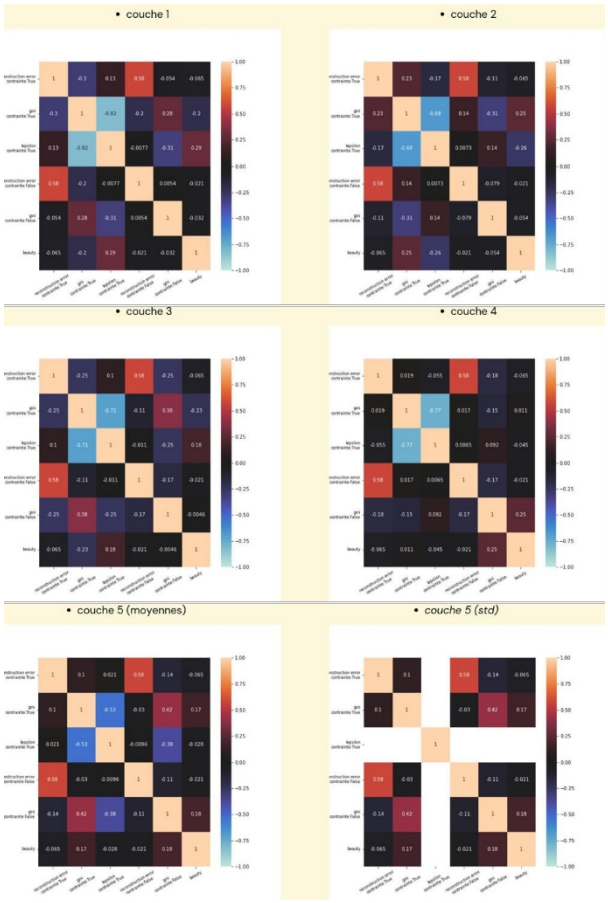


Figure 43: Pearson correlations between fluence metrics, and beauty scores on CFD, for the small vae with 5 layers at the decoder. We give the correlations for the metrics on a trained model with and without the constraint. We observe alternating correlations on the first layers, which is not very interpretable. Otherwise, we have no correlation with beauty scores. Constraint seems to bring out correlations with higher magnitude, perhaps because it reinforces the brain model.

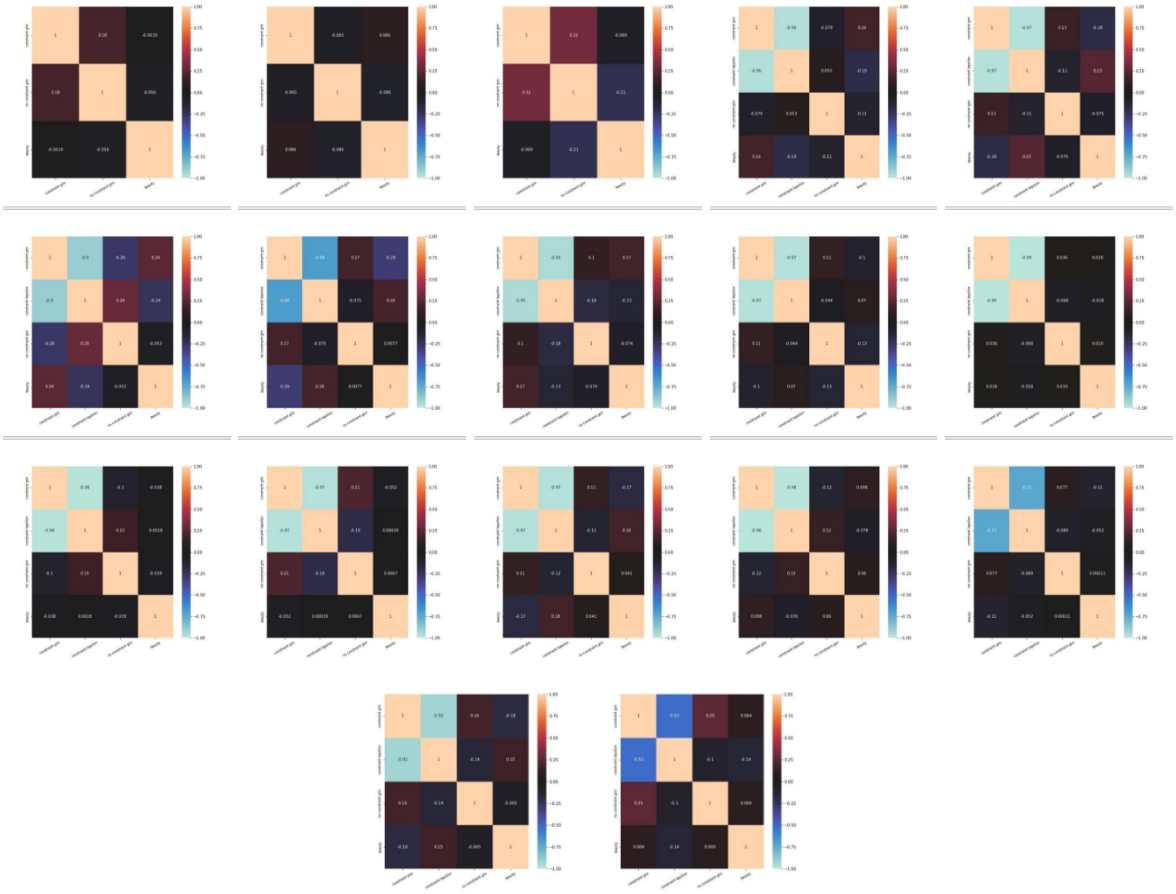


Figure 44: Correlations between fluency metrics and beauty scores on CFD, for vgg vae with 5 layers at the decoder. We see the same oscillation in correlation scores, surrounded by zero correlations.

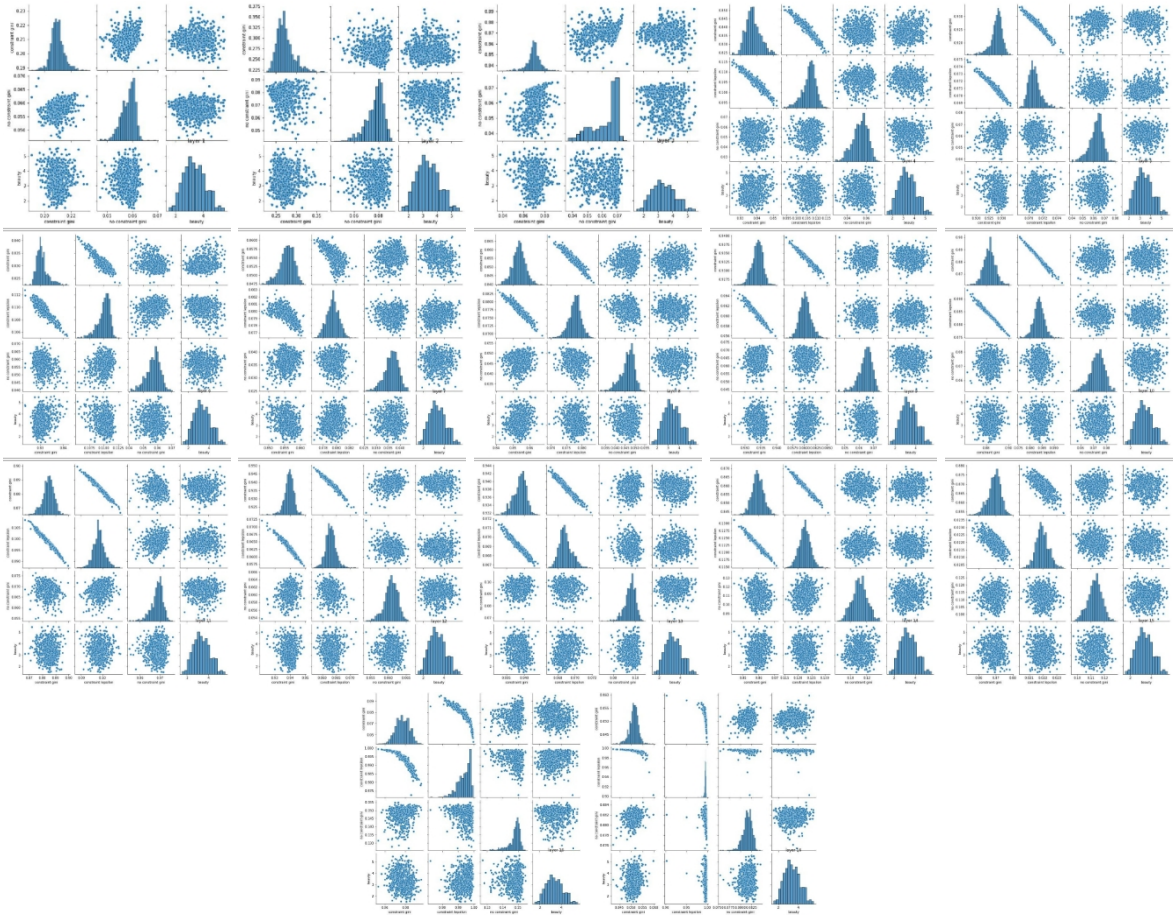


Figure 45: joint distributions between fluence metrics and beauty scores on CFD, for vgg vae with 5 layers at the decoder (to check for non-linear relationships)

R^2 ridge regression	sparsité partout	sparsité partout sauf conv1	sparsité partout sauf conv1 et moyennes	pas de sparsité
lepsilon	0.106	0.103	0.0538	intolérable
gini	0.0740	0.0759	0.0925	0.0948

Figure 46: Results of linear regressions with l2 normalization (mean r2 over 10 models), for the small model trained with stress at different locations.

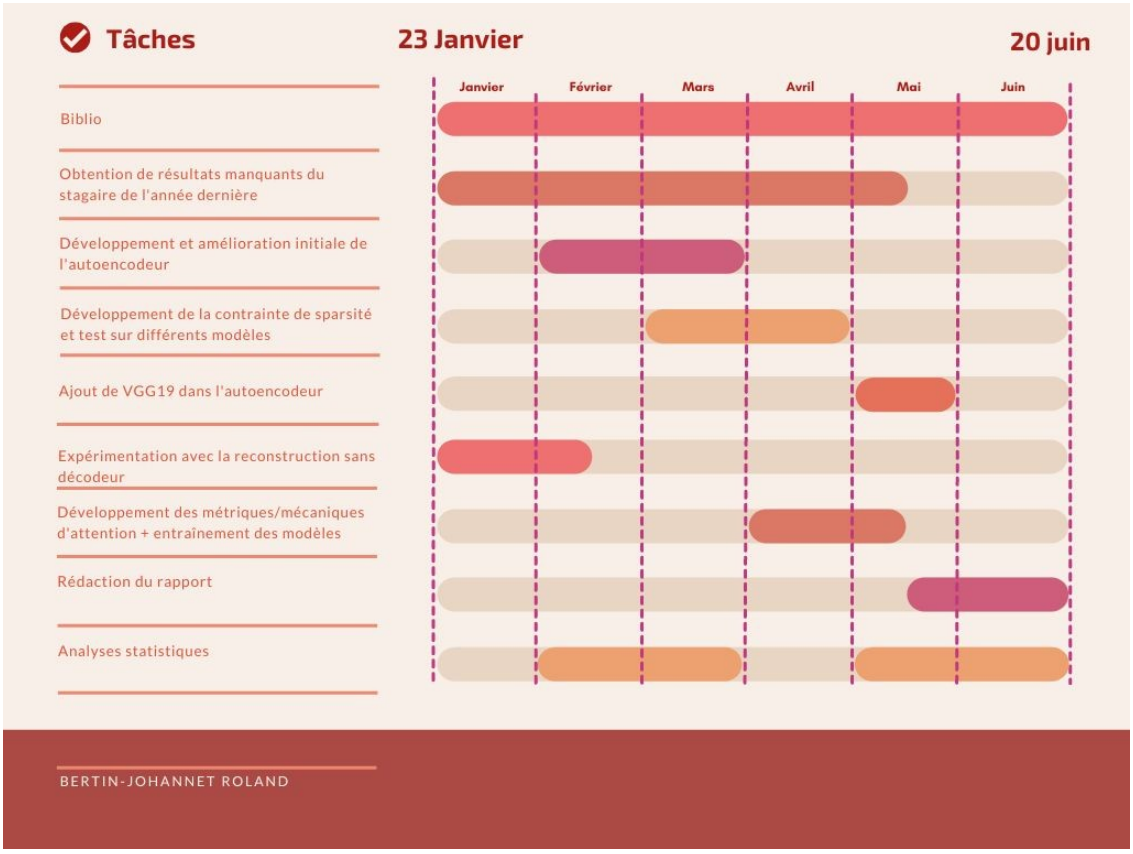


Figure 47: Allocation of time to the various course tasks.

- Acquaintance with some of the bibliography in the field of com- putational neuroscience
- Awakening to various theories of aesthetics
- Practice the scientific method
- Apply statistics to answer new questions
- Collaboration and discussion with a variety of profiles (mathematicians, ecologists, neuroscientists, Deep Learning specialists).

Future prospects for this topic

It would be interesting in the future, even if it goes beyond the scope of this internship, to add a temporal dimension to our metrics. This could lead to more realistic and biologically advanced attention, and could be achieved with reinforcement learning.

References

[1] Horace B Barlow et al. Possible principles underlying the transformation of sensory messages. *Sensory communication*, 1(01):217-233, 1961.

[2] Aenne A Brielmann and Peter Dayan. A computational model of aesthetic value. *Psychological Review*, 2022.

[3] Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images. *arXiv preprint arXiv:2011.10650*, 2020.

[4] Denis Dutton. *The art instinct: Beauty, pleasure, & human evolution*. Oxford University Press, USA, 2009.

[5] Ronald A Fisher. The evolution of sexual preference. *The Eugenics Review*, 7(3):184, 1915.

[6] Griffin Floto, Stefan Kremer, and Mihai Nica. The tilted variational autoencoder: Improving out-of-distribution detection. In *The Eleventh International Conference on Learning Representations*, 2023.

[7] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[8] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193-202, 1980.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[11] Laura K. M. Graf and Jan R. Landwehr. A dual-process perspective on fluency-based aesthetics: the pleasure-interest model of aesthetic liking. *Personality and Social Psychology Review: An Of- ficial Journal of the Society for Personality and Social Psychology, Inc*, 19(4):395-410, November 2015.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[13] Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. Attgan: Facial attribute editing by only changing what you want. *IEEE transactions on image processing*, 28(11):5464-5478, 2019.

[14] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262-15271, 2021.

[15] Alexander Hepburn, Valero Laparra, Raul Santos-Rodriguez, Johannes Ballé, and Jesús Malo. On the relation between statistical learning and perceptual distances, 2022.

[16] Xianxu Hou, Linlin Shen, Ke Sun, and Guoping Qiu. Deep feature consistent variational autoencoder. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 1133-1141. IEEE, 2017.

[17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[18] David H Hubel and Torsten N Wiesel. Receptive fields of single neurons in the cat's striate cortex. *The Journal of physiology*, 148(3):574, 1959.

[19] N. Hurley and S. Rickard. Comparing Measures of Sparsity. *IEEE Transactions on Information Theory*, 55(10):4723-4741, October 2009.

[20] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features, 2019.

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448-456. pmlr, 2015.

[22] Kimmo Karkkainen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1548-1558, 2021.

[23] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[24] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84-90, 2017.

[26] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278-2324, 1998.

[27] Grace W Lindsay. Attention in psychology, neuroscience, and machine learning. *Frontiers in computational neuroscience*, 14:29, 2020.

[28] Grace W Lindsay. Convolutional neural networks as a model of the visual system: Past, present, and future. *Journal of cognitive neuroscience*, 33(10):2017-2031, 2021.

[29] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*, 2017.

[30] Debbie S. Ma, Joshua Correll, and Bernd Wittenbrink. The Chicago face database: A free stimulus set of faces and norming data. *Behavior Research Methods*, 47(4):1122-1135, December 2015.

[31] Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. Biva: A very deep hierarchy of latent variables for generative modeling. *Advances in neural information processing systems*, 32, 2019.

[32] Beren Millidge, Anil Seth, and Christopher L Buckley. Predictive coding: a theoretical and experimental review, 2022.

[33] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks, 2015.

[34] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[35] Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607-609, 1996.

[36] Konstantinos P. Panousis, Sotirios Chatzis, and Sergios Theodoridis. Stochastic local winner- takes-all networks enable profound adversarial robustness, 2021.

[37] Johanna Pasquet, Jérôme Pasquet, Marc Chaumont, and Dominique Fouchez. Pelican: deep architecture for the light curve analysis. *Astronomy & Astrophysics*, 627:A21, 2019.

[38] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch, 2017.

[39] Richard O Prum. *The evolution of beauty: How Darwin's forgotten theory of mate choice shapes the animal world-and us*. Anchor, 2018.

[40] Xuming Ran, Mingkun Xu, Lingrui Mei, Qi Xu, and Quanying Liu. Detecting out-of-distribution samples via variational auto-encoder with reliable uncertainty estimation. *Neural Networks*, 145:199-208, 2022.

[41] Rolf Reber, Norbert Schwarz, and Piotr Winkielman. Processing Fluency and Aesthetic Pleasure: Is Beauty in the Perceiver's Processing Experience? *Personality and Social Psychology Review*, 8(4):364-382, November 2004.

[42] Julien P Renoult. The evolution of aesthetics: A review of models. *Aesthetics and Neuroscience: Scientific and Artistic Perspectives*, pages 271-299, 2016.

[43] Julien P. Renoult, Jeanne Bovet, and Michel Raymond. Beauty is in the efficient coding of the beholder. *Royal Society Open Science*, 3(3):160027, March 2016.

[44] Julien P Renoult and Tamra C Mendelson. Processing bias: extending sensory drive to in- clude efficacy and efficiency in information processing. *Proceedings of the Royal Society B*, 286(1900):20190165, 2019.

[45] Michael J Ryan. A taste for the beautiful. In *A Taste for the Beautiful*. Princeton University Press, 2018.

[46] Elham Saraee, Mona Jalal, and Margrit Betke. Visual complexity analysis using deep intermediate-layer features. *Computer Vision and Image Understanding*, 195:102949, 2020.

[47] Jean-Marie Schaeffer. *The aesthetic experience*. Editions Gallimard, March 2015. Google-Books-ID: JsDmBgAAQBAJ.

[48] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network, 2016.

[49] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, April 2015. arXiv:1409.1556 [cs].

[50] Jiaming Song and Stefano Ermon. Understanding the limitations of variational mutual information estimators. *arXiv preprint arXiv:1910.06222*, 2019.

[51] Michele Svanera, Andrew T Morgan, Lucy S Petro, and Lars Muckli. A self-supervised deep neural network for image completion resembles early visual cortex fmri activity patterns for occluded scenes. *Journal of Vision*, 21(7):5-5, 2021.

[52] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[53] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *Advances in neural information processing systems*, 33:19667-19679, 2020.

[54] Rufin VanRullen and Andrea Alamia. Gattanet: Global attention agreement for convolutional neural networks. In *Artificial Neural Networks and Machine Learning-ICANN 2021: 30th Inter- national Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14-17, 2021, Proceedings, Part I*, pages 281-293. Springer, 2021.

[55] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600-612, 2004.

[56] Piotr Winkielman, Jamin Halberstadt, Tedra Fazendeiro, and Steve Catty. Prototypes are attractive because they are easy on the mind. *Psychological Science*, 17(9):799-806, September 2006.

[57] Piotr Winkielman, Norbert Schwarz, Tedra Fazendeiro, and Rolf Reber. The Hedonic marking of Processing Fluency: implications for evaluative judgment. In *The Psychology of Evaluation: Affective Processes in Cognition and Emotion*. Psychology Press, January 2003.

[58] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3-19, 2018.

[59] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586-595, 2018.