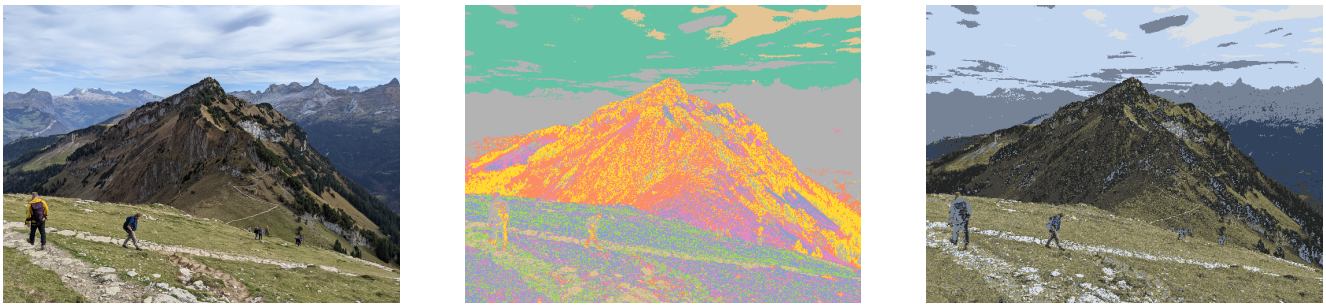
Game Theory for Image Segmentation

Clément JAMBON
cjambon@student.ethz.ch

June 13, 2023



(a) (b) (c)

Figure 1: Segmentation obtained with our implementation of *Pure Infection and immunization dynamics*: (a) is the original image, (b) shows the resulting clusters and (c) the average colors over the segments.

1 Introduction

Image segmentation has widely been explored in the image processing community. As a consequence, there exists a significant amount of methods including k-means, graph-based methods [5], histogram clustering [9] or more recently neural approaches that build on neural networks to extract priors from large amount of data [6] and allow to cluster pixels beyond color-space similarities.

In this project, motivated by the material of "Controversies in Game Theory"[4], we choose to focus on another approach that builds on Game Theory and Evolutionary Dynamics. More precisely, inspired by the work of Shen et al. [11] and the PhD dissertation of Samuel Rota Bulò [2], we propose to formulate image segmentation as a clustering game. In order to find the equilibrium of such a game, we lift it to a discrete-time evolutionary dynamics formulation where we make mixed strategies evolve in search for an *Evolutionary Stable Strategy* (ESS). As images are very high-dimensional and lead to slow convergence when using standard algorithms like *Best response dynamics* and *Replicator dynamics*, we follow [11] and [2], and use the *Infection and immunization dynamics* with a pure strategy selection function as introduced by Bulò in [10]. Fig. 1 provides a glimpse at the results of our implementation.

We start by reviewing the notion of clustering games with the material from the course [4] in section 2. We then introduce discrete-time *Evolutionary dynamics* and the algorithms we use to find such strategies in section 3. This allows us to define image segmentation as a clustering game and we enumerate in section 4 our assumptions and experimental setup. Finally, we present our results on RGB similarities in section 5 and propose to extend the method to higher-dimensional features from state-of-the-art deep learning models, namely DINO features [3] in section 6.

Before getting any further, we would like to emphasize that all the results presented in this report were obtained from scratch. Both [2] and [11] provide a methodology for the described algorithms but we are not aware of any existing implementation. For reproducibility and educational purposes, we make all of our code in the form of documented *notebooks* available at `https://github.com/clementjambon/evolutionary-segmentation`.

2 Clustering games

In the following, we introduce the notion of clustering games and try to stay as close as possible to the notions introduced by Heinrich Nax and Jonny Newton in their lectures from course [4]. At every step, we try to connect the formalism we introduce to the problem of segmentation.

We start by defining $S = \{1, \dots, n\}$ a set of *pure strategies* which represent objects to be clustered, in our case image pixels. We connect strategies with a payoff/utility matrix A where $A_{i,j}$ specifies how much one can expect from choosing strategy i against a strategy j. In the clustering case, we can interpret this matrix as a similarity matrix that quantifies how close pixel i is to pixel j. Note that in the following, we will only deal with symmetric payoffs (which is consistent to our image-space clustering formulation).

In practice, we will consider linear combinations $\mathbf{x}$ of these pure strategies as *mixed strategies*. In order to be consistently comparable, they live in the n-dimensional simplex:

$$\Delta = \left\{ \mathbf{x} = (x_1, \dots, x_n) \mid \sum_{i=1}^{n} x_i = 1 \right\} \tag{1}$$

Every mixed strategy comes with a set of non-zero components $\sigma(\mathbf{x})$ which is called its *support*. Intuitively, this will represent a set of pixels belonging to the same cluster.

With these definitions, we can now see that playing a pure strategy i gives an expected payoff $\pi(\mathbf{e}_i|\mathbf{x}) = \mathbf{e}_i^T A\mathbf{x}$ where $\mathbf{e}_i$ is the i-th vector of the canonical basis of $\mathbb{R}^n$. More generally, the expected payoff of a mixed strategy $\mathbf{y}$ against another mixed strategy $\mathbf{x}$ is given by $\pi(\mathbf{y}|\mathbf{x}) = \mathbf{y}^T A\mathbf{x}$.

As they will be necessary to understand some of the concepts presented next and in section 3, we also introduce the following notations:

- the expected payoff of $\mathbf{x}$ against itself is $\pi(\mathbf{x}) = \mathbf{x}^T A\mathbf{x}$
- $\pi(\mathbf{y} - \mathbf{x}|\mathbf{x}) = \pi(\mathbf{y}|\mathbf{x}) - \pi(\mathbf{x})$
- the *best response* strategies $\beta(\mathbf{x})$ against $\mathbf{x}$ are the strategies that give the maximum payoff when played against $\mathbf{x}$, namely $\beta(\mathbf{x}) = \arg\max_{\mathbf{y}} \pi(\mathbf{y}|\mathbf{x})$. Note that these may not be unique!

We can now intuitively define the notion of Nash equilibrium with these notations: $\mathbf{x}$ satisfies a *Nash equilibrium* if no other strategies improve its expected payoff against $\mathbf{x}$ than itself or more formally and equivalently (with all the definitions we have):

$$\forall \mathbf{y} \in \Delta, \quad \pi(\mathbf{y}|\mathbf{x}) \leq \pi(\mathbf{x}) \iff \forall \mathbf{y} \in \Delta, \quad \pi(\mathbf{y} - \mathbf{x}|\mathbf{x}) \leq 0 \iff \mathbf{x} \in \beta(\mathbf{x}) \tag{2}$$

Let's now come back to our clustering game. Intuitively, the goal will be, given a pixel similarity matrix A, to find a mixed strategy $\mathbf{x}$ that satisfies a Nash equilibrium and the support of such a strategy will give the pixels classified as part of the same cluster and the remaining pixels will be assigned to another cluster. You may then ask: what if we want to perform clustering with more than 2 clusters? We can simply slice-off the pixels belonging to the first cluster from S and similarity matrix A and find a new Nash equilibrium. In fact, this scheme provides a hint at the strength of clustering games. As emphasized by Bulò in [2], we need not know the relevant number of clusters in advance, which is a challenging task in image segmentation and clustering in general! Instead, we can just run this procedure recursively until S is a singleton or we cannot find a Nash equilibrium (this latter point should not happen with a real symmetric matrix though!). Each of these iterations is called in both [2] and [11] a *segment*. In practice, as we shall present in section 4, we define a maximum number of segments in our segmentation case.

3 Evolutionary dynamics

With the rules of our *clustering game* set, we now need to find how to play the game! More formally, the challenge lies in **efficiently** finding an equilibrium. To this extent, we follow three approaches from

Evolutionary Dynamics covered by [2] and [11]. For conciseness and due to our lack of extensive knowledge of the field, we will only cover the main definitions and give the intuitions behind the corresponding algorithms.

Evolutionary Dynamics provides a framework to model the evolution of populations, or in other words mixed strategies, as a function of time. Given an initial mixed strategy $\mathbf{x}^{(0)}$, its evolution can be described as a general differential equation:

$$\dot{\mathbf{x}}^{(t)} = g(\mathbf{x}^{(t)}, t) \tag{3}$$

In practice, under some assumptions (notably a proper renormalization of $\mathbf{x}^{(t)}$), we can discretize this continuous process using discrete time steps $t = 1, 2, \ldots$ and express the population $\mathbf{x}^{(t+1)}$ at time $t+1$ as a function of the population $\mathbf{x}^{(t)}$ at time t.

Best response dynamics

Best response dynamics builds on the intuition that we can inject in the population, at each time step t, one individual that would yield the best response w.r.t. the payoff matrix. This gives the following discrete-time dynamics:

$$\mathbf{x}^{(t+1)} = \frac{\mathbf{r}^{(t+1)} - \mathbf{x}^{(t)}}{t+1} + \mathbf{x}^{(t)} \quad \text{where } \mathbf{r}^{(t+1)} \in \beta(\mathbf{x}^{(t)}) \tag{4}$$

As the population "grows", the newly added individuals should have less and less of an impact: this is captured by the term $\frac{1}{t+1}$ in the equation above.

Replicator dynamics

Replicator dynamics is a subset of the more general *imitation dynamics*. The key idea is to rescale each entry i of the mixed strategy vector $\mathbf{x}^{(t)}$ at time t depending on how well playing the pure strategy i against itself (namely $\mathbf{x}^{(t)}$) performs:

$$x_i^{(t+1)} = x_i^{(t)} \frac{\pi(\mathbf{e}_i|\mathbf{x}^{(t)}) + C}{\pi(\mathbf{x}^{(t)}) + C} \tag{5}$$

The constant C is simply introduced to ensure that we do not divide by zero. In our case, as we shall present in section 4, the similarity matrix has only stricly positive entries and we can just choose C to be zero.

Infection and immunization dynamics

Unfortunately, *Best response dynamics* and *Replicator dynamics* tend to be particularly slow at approaching an equilibrium (especially when the set of strategies S becomes large which will be a concern to us considering the large number of pixels in an image). See section 3.10 of [2] for a quantitative discussion of this point. To address this, Bulò and Bomze introduced in [10] *Infection and immunization dynamics* (which we further denote as InImDyn for brevity).

The key takeaway of this more advanced mechanism is, starting from $\mathbf{x}$, to progressively infect the population with another population against which it is not immune. This entails to define the notion of infection and we can naturally assume that a population $\mathbf{y}$ is *infective* for another population $\mathbf{y}$ if it stricly improves its payoff, namely $\pi(\mathbf{y}|\mathbf{x}) > \pi(\mathbf{x}) \iff \pi(\mathbf{y} - \mathbf{x}|\mathbf{x}) > 0$. Following [2], we denote the corresponding set of infective strategies as $\Upsilon(\mathbf{x}) = \{\mathbf{y} \in \Delta \mid \pi(\mathbf{y} - \mathbf{x}|\mathbf{x}) > 0\}$.

However, to actually put into practice such a strategy, two points need to be clarified:

- The first is: "which strategy should infect population $\mathbf{x}$?" More formally, we need to define an infective *strategy selection function* $\mathcal{S}(\mathbf{x})$. In [10], a rather simple idea is proposed: find a pure strategy $\mathbf{e}_i$ that maximizes $\pi(\mathbf{e}_i - \mathbf{x}|\mathbf{x})$. Actually, this can be extended further to maximizing $|\pi(\mathbf{e}_i - \mathbf{x}|\mathbf{x})|$ and in the case where this maximum corresponds to a negative value of $\pi(\mathbf{e}_i - \mathbf{x}|\mathbf{x})$ (only allowed when i is in the support of $\mathbf{x}$ in order to be able to define the *costrategy* thereafter),

we can choose what is called the *costrategy* $\bar{\mathbf{e}}_i^{\mathbf{x}}$ of $\mathbf{e}_i$ w.r.t $\mathbf{x}$ that is obtained by projecting $\mathbf{e}_i$ on the opposite side of the simplex Δ while passing through $\mathbf{x}$. This can be written analytically as $\bar{\mathbf{e}}_i^{\mathbf{x}} = -\frac{1}{x_i - 1}\mathbf{e}_i + \frac{x_i}{x_i - 1}\mathbf{x}$.

- The second point is "how much of this selected population $\mathbf{y} = \mathcal{S}(\mathbf{x}^{(t)})$ we should inject within the existing population $\mathbf{x}^{(t)}$ at time t. This can be thought of as choosing a nice scalar $\delta_{\mathbf{y}}(\mathbf{x})$ (similar to $\frac{1}{t+1}$ in eq. 4 describing best response dynamics) such that we can write the discrete update as:

$$\mathbf{x}^{(t+1)} = \delta_{\mathbf{y}}(\mathbf{x})(\mathbf{y} - \mathbf{x}^{(t)}) + \mathbf{x}^{(t)} \tag{6}$$

  By building on the properties of the simplex (we omit the introduction of the notions of *score function* and *invasion barriers* to keep the discussion simple) and in the case of the clustering game scenario introduced in section 2, one can derive that we can choose (see [2]):

$$\delta_{\mathbf{y}}(\mathbf{x}) = \begin{cases} \min\left(\frac{\pi(\mathbf{x}-\mathbf{y}|\mathbf{x})}{\pi(\mathbf{y}-\mathbf{x})}, 1\right) & \text{if } \pi(\mathbf{y}-\mathbf{x}) < 0 \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

We now mention two results from [2] that hint at the convergence of InImDyn under the pure selection strategy:

- First, Proposition 4 of section 3 of [2] ensures that there exists an infective strategy for a mixed strategy $\mathbf{x}$ if and only if the strategy given by the pure strategy selection scheme described above is injective.
- Second, Theorem 2 of section 3 of [2] guarantees that if the set of infective strategies $\Upsilon(\mathbf{x})$ for a population $\mathbf{x}$ is empty, then $\mathbf{x}$ is a fixed point of the InImDyn dynamics and more interestingly a Nash equilibrium.

4 Playing the segmentation game

With all this nice formalism at our disposal, we now expose how we can use it to practically perform image segmentation. Additionally, we clarify our experimental setup.

First of all, we need to define the actual payoff matrix A. Similar to [2] and [11], we choose to compute similarities using a Gaussian Kernel:

$$A_{i,j} = \exp\left(-\frac{\|C(i) - C(j)\|^2}{\sigma^2}\right) \tag{8}$$

where $C(i)$ corresponds to a 3-dimensional vector representing the value of pixel i in the chosen color space. As suggested in [2], we convert initial values in *RGB* space to the *Lab* color space using the implementation provided by the OpenCV library[1]. We choose σ to match the dynamic range of standard images, namely $\sigma = 250$.

As computations scale quadratically with the side-length of images (because we have $n = H \times W$ pixels), we follow again [2] and subsample pixels at different rates (see next section), run our clustering and then assign out-of-sample pixels using the dominant-set technique from [8]. In all cases, we initialize the population with $\mathbf{x}^{(0)}$ a constant value (i.e. $\frac{1}{n}$). In other words, all pixels start with an equal probability of belonging to the cluster. We run clustering on at most 8 segments, which yields at most 9 clusters (because we need to account for the potentially non-classified samples at the end of the last segment). For all experiments, we set the maximum number of iterations in a segment to 5000 and stop the dynamics if the current strategy reaches an *ε-neighborhood* of a Nash equilibrium defined in [2] as

$$\sum_{i=1}^{n} [\min(x_i, \pi(\mathbf{x} - \mathbf{e}_i|\mathbf{x}))]^2 \leq \varepsilon^2 \tag{9}$$

and we choose $\varepsilon = 10^{-3}$.

Finally, as we try to reproduce the results of [2] and [11], we evaluate our algorithms on the BSDS300 dataset[7].

(a) Best Response dynamics (b) Replicator dynamics (c) Pure InImDyn

Figure 2: Segmentations obtained with our implementations of the three dynamics presented in section 3 at sampling rate $p = 0.01$.

5 Results

All the results provided thereafter can be reproduced by using the notebook provided with this report.

In Fig. 2 we compare the three different dynamics on three images from the test set of the BSDS300 dataset. These pictures were obtained by averaging the colors over each segment in *Lab* space. We can clearly see that with a fixed maximum number of iterations, Pure InImDyn clearly outperforms Best Response dynamics. The difference is not so clear for Replicator dynamics. However, as we shall see below, it is significantly more efficient than the later!

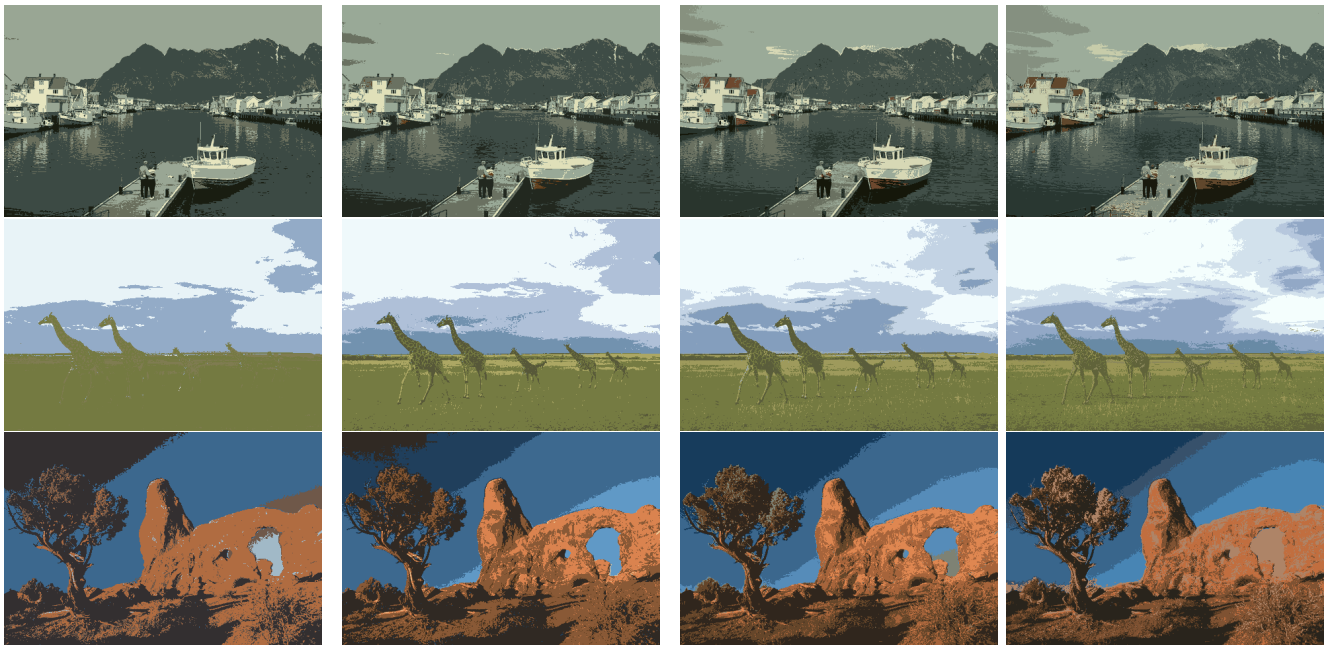
Next, we investigate the impact of the sampling rate p. For this, we focus on the InImDyn dynamics. As shown in Fig. 3 and unsurprisingly, higher sampling rates give perceptually more consistent clusters. However, higher sampling rates require more iterations to converge. This explains the slight decrease in quality between $p = 0.005$ and $p = 0.01$ in Fig. 3.

Furthermore, to motivate the advantage of Pure InImDyn over Best Response and Replicator dynamics, we execute our three dynamics on the test set of the BSDS300 dataset and record the execution time. Fig. 4 shows the significant gain that InImDyn provides us with.

Finally, we show in Fig. 5 examples of the distribution of strategies as they evolve following the discrete-time dynamics of each of the investigated approaches. These results are consistent with the ones presented in [11]. We can see in the right column that Pure InImDyn reaches an equilibrium faster than the other dynamics.

6 Going deeper: *DINO* features for semantic segmentation

Motivated by the advent of deep learning and the more interesting task of *semantic segmentation*, we propose an attempt to extend the method to tackle this problem using *DINO* features. These high-dimensional features were introduced by Caron et al. in [3]. They result from a self-supervised training



(a) $p = 0.001$ (b) $p = 0.0025$ (c) $p = 0.005$ (d) $p = 0.01$

Figure 3: Segmentations obtained at different sampling rates.

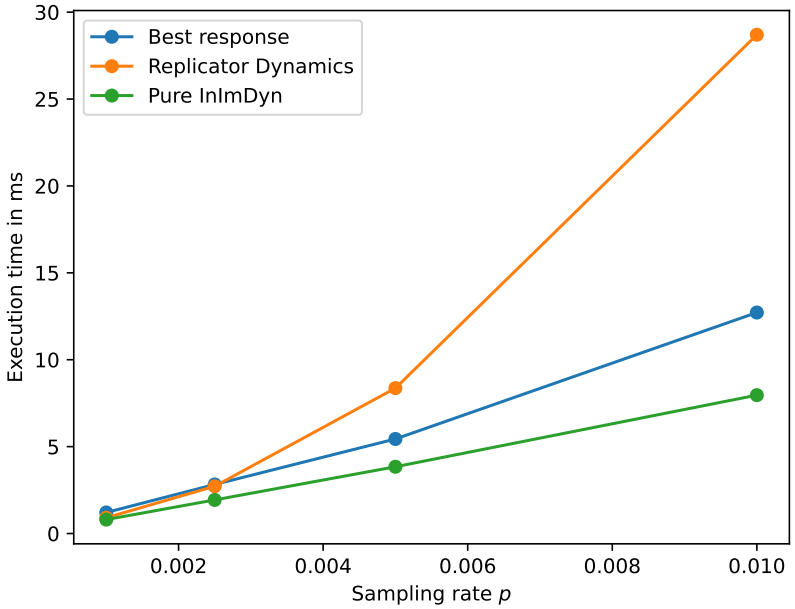


Figure 4: Average execution time on the test set of the BSDS300 dataset at different sampling rates and across the multiple dynamics

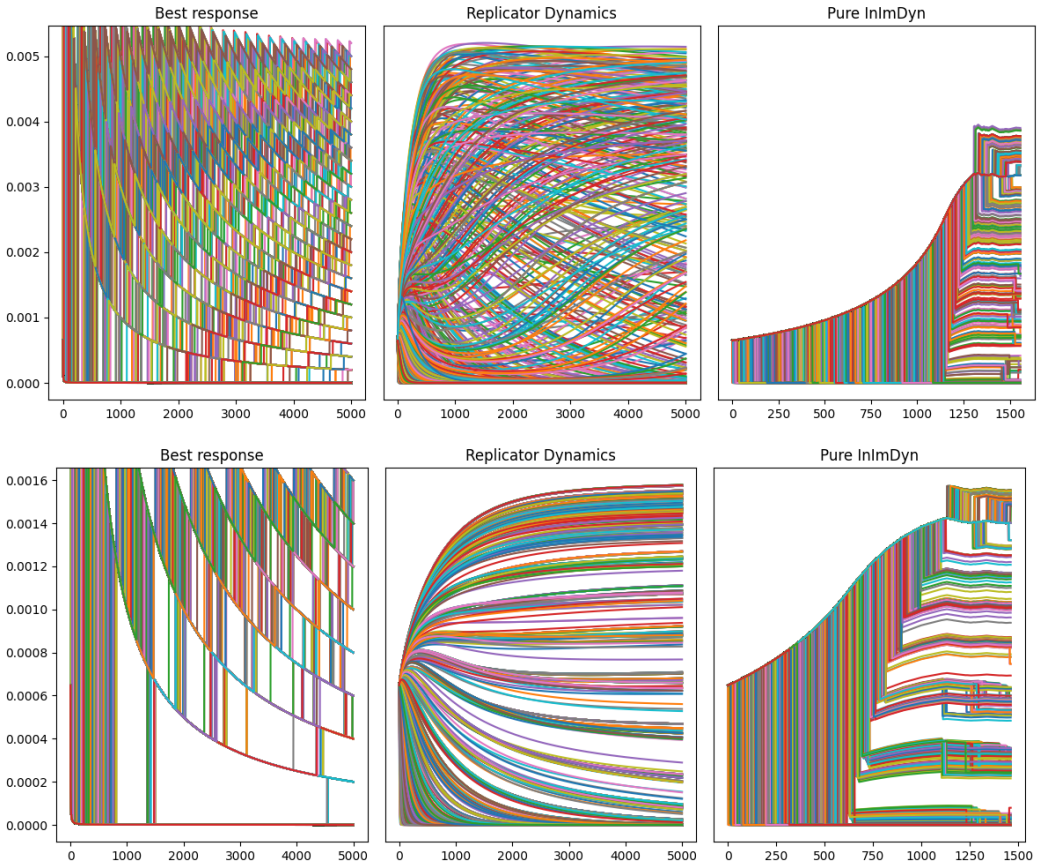


Figure 5: Trajectories of the mixed strategy entries on the first segment of two images from the BSDS300 dataset

procedure with Vision Transformers [12]. Most importantly to us, they were shown to embed semantic information. We thus investigate whether we can repurpose our segmentation algorithm from RGB space to this high-dimensional space.

First of all, we extract DINO features using the facebook/dino-vitb8 model available on huggingface. This model assumes images of size 224×224 and produces features of dimension 768 by considering patches of 8 pixels. This results in 28×28 images, which is significantly low resolution but still provides us with a coarse semantic segmentation in the end. Additionally, we propose alternative results where we tile the image into 4 blocks of 224×224 thus doubling the size of the result. Yet due to the receptive field of the network, this results in artifacts at the junction of the tiles. As 768 is a very high dimension, we run a PCA on the features to reduce their dimensionality to 64 only.

In order to run our segmentation algorithm, we modify the similarity matrix to account for the high dimensionality of DINO features. Instead of a L2 loss, we consider cosine similarities, thus redefining A with $A_{i,j} = \exp\left(-\frac{(1-\text{sim}(\phi_i,\phi_j))}{T}\right)$ where sim stands for the cosine similarity between the features ϕ_i (resp. ϕ_j) from pixel i (resp. j) and T is a temperature parameter. In our experiments, we chose $T = 25$.

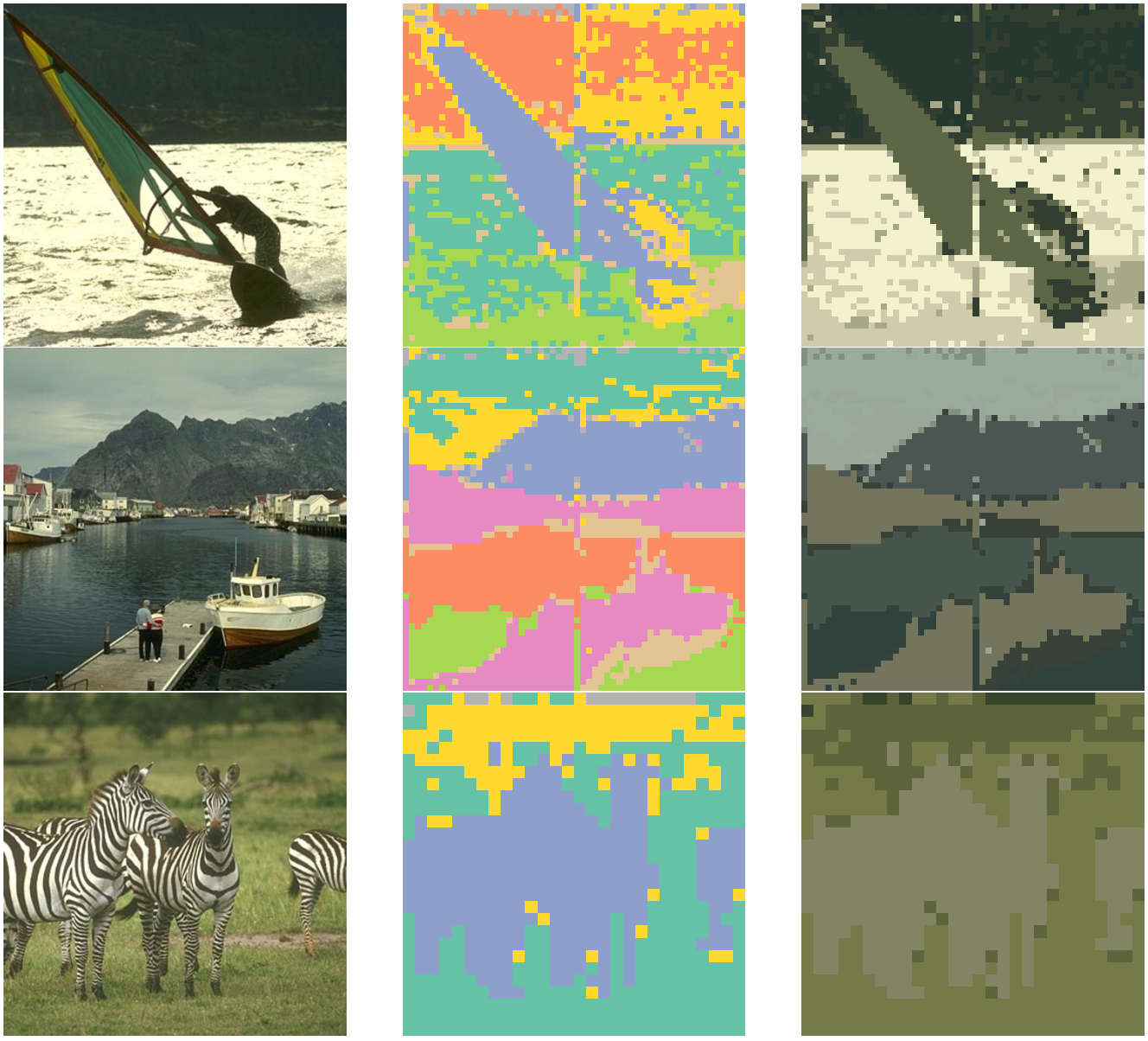
Fig. 6 (resp. Fig. 7) shows the results of segmentation with 1 (resp. 4 tiles). Despite the rather low image size of the DINO feature maps, we can clearly segment semantically the input images. I find it particularly interesting that we managed to adapt a decade-old algorithm to state-of-the-art methods in a conclusive manner.

7 Conclusion and discussion

In this project, we investigated a game theoretic approach to Image Segmentation. To do so, we built on the previous works [2] and [11], and proposed to study three different evolutionary dynamics: *Best Response dynamics*, *Replicator dynamics* and *Infection and immunization dynamics* with a pure selection

(a) Original (b) Clusters (c) Average colors

Figure 6: Segmentations obtained with DINO features on 28×28 feature maps.



(a) Original (b) Clusters (c) Average colors

Figure 7: Segmentations obtained with DINO features on 4 tiles of 28×28.

strategy. Consistent with their results, we showed that InImDyn provides a good tradeoff between efficiency and quality (in terms of equilibrium).

Our clustering game is very simple and only rely on pairwise similarities between pixels without taking into account spatiality or any form of aggregated payoff. As a consequence, it would be interesting to study higher-order similarities as introduced later in Bulò's PhD dissertation[2].

References

[1] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[2] Samuel Rota Bulò. A game-theoretic framework for similarity-based data clustering. jan 2009. PhD Thesis. Available at `https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=6f2491d53e273343ba3d6374c4e7309c7d68fcd1`.

[3] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers, 2021.

[4] H. Rauhut D. Helbing, H.Nax. Controversies in Game Theory IX: Cooperative and Non-Cooperative Game Theory. Spring Semester 2023.

[5] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[6] Alexander Kirillov, Kaiming He, Ross B. Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. *CoRR*, abs/1801.00868, 2018.

[7] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[8] Massimiliano Pavan and Marcello Pelillo. Efficient out-of-sample extension of dominant-set clusters. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004.

[9] J. Puzicha, T. Hofmann, and J.M. Buhmann. Histogram clustering for unsupervised image segmentation. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 2, pages 602–608 Vol. 2, 1999.

[10] Samuel Rota Bulò and Immanuel M. Bomze. Infection and immunization: A new class of evolutionary game dynamics. *Games and Economic Behavior*, 71(1):193–211, 2011. Special Issue In Honor of John Nash.

[11] Dan Shen, Erik Blasch, Khanh Pham, and Genshe Chen. A clustering game based framework for image segmentation. In *2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, pages 818–823, 2012.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.