A Modular and Interpretable Pipeline for Unsupervised Learning on Scientific Spatiotemporal Imaging Datasets

Timothée Levilly [1] **Zhen Yuan Yeo** [1] **Ying Chen Lim** [2] **Ngoc Thi Nguyen** [2 3] **Alexandre Thiéry** [1]
N. Duane Loh [2 3]

[1]*Department of Statistics and Data Science, National University of Singapore, Singapore 117546, Singapore* [2]*Department of Physics, National University of Singapore, Singapore 117551, Singapore* [3]*Centre for Bio-imaging Sciences, National University of Singapore, Singapore 117557, Singapore*. Correspondence to: duaneloh@nus.edu.sg.

1. Abstract

We present a modular software package that readily generalizes to many unsupervised vector-quantization of complex spatiotemporal patterns in imaging time sequences. This modularity comprises four core stages – *segmentation*, *featurization*, *tokenization*, and *motif learning* – each with interchangeable implementations. The separation into these four stages allows clear checkpoints to interrogate the decisions and vector-quantization learned by machine learning tools for each stage, regardless of whichever implementations were chosen for each stage.

Furthermore, this modularity allows users to easily A/B test different choices to identify configurations best suited to their data and scientific questions. By exposing these options in a structured and machine-readable form, it enables LLM prompted, AI assisted analysis of new scientific datasets, allowing experimentalists to focus on hypothesis generation and scientific interpretation rather than pipeline engineering.

2. Introduction

Unsupervised learning has become central to the analysis of spatiotemporal imaging data across biology and physics. Examples include extracting features from images using various self-supervised methods [1] [2] [3], and novel non-linear dimensionality reduction techniques for visualization such as Uniform Manifold Approximation and Projection (UMAP) [4]. However, existing scientific workflows are often task specific, tightly coupled, and difficult to interrogate. This limits reuse, comparison, and interpretability across datasets.

We present a modular framework for unsupervised analysis of imaging time series built around four explicit stages: segmentation, featurization, tokenization, and motif learning. Each stage is decoupled, interchangeable, and inspectable, enabling systematic exploration of modeling choices while maintaining a consistent data flow.

Our contribution is a general data-analysis pipeline design that emphasizes robustness, flexibility, and accessibility for scientific discovery.

3. Framework Overview

The framework decomposes unsupervised spatiotemporal analysis into four conceptually distinct stages. This separation reflects common analytical steps already used implicitly in many studies, but makes them explicit and configurable.

Here is a high-level view of the 4 protocol steps as illustrated in Figure 1:

Segmentation Extract subregions of the original data that capture meaningful local behavior.

Featurization Create a latent space representation of meaningful feature vectors for the regions of interest.

Tokenization Discretizes the feature space into a finite set of states using dimensionality reduction and vector quantization. The outputs are a codebook in latent space and reconstructed representative states in image space.

Motif Learning Interpret spatial and/or temporal relationships between persistent states.

4. Results and Discussion

Interchangeable modules: Each pipeline stage supports multiple interchangeable implementations with a shared interface. These 4 steps have to be tailored for the needs of individual domain-specific datasets. Users may assemble pipelines programmatically through a structured API or declaratively using a TOML configuration file.

Unified data structures across pipeline stages: All stages of the pipeline operate on a shared set of standardized yet flexible data structures, enabling consistent interaction between segmentation, featurization, tokenization, and motif learning. Image data, intermediate representations, and learned tokens follow a common shape convention that accommodates diverse modalities, including 2D and 3D data, grayscale and multichannel images, and both static and time resolved datasets. This is shown in Figure 2. This unified representation ensures interoperability between interchangeable modules, simplifies custom method integration, and supports both programmatic and configuration driven pipeline construction.

AI assisted pipeline design: The configuration driven approach enables rapid experimentation without direct code modification. Configuration options and documentation can also be exposed to a large language model, allowing AI assisted generation of valid analysis pipelines from natural language descriptions. A demonstration of such usage is shown in Appendix

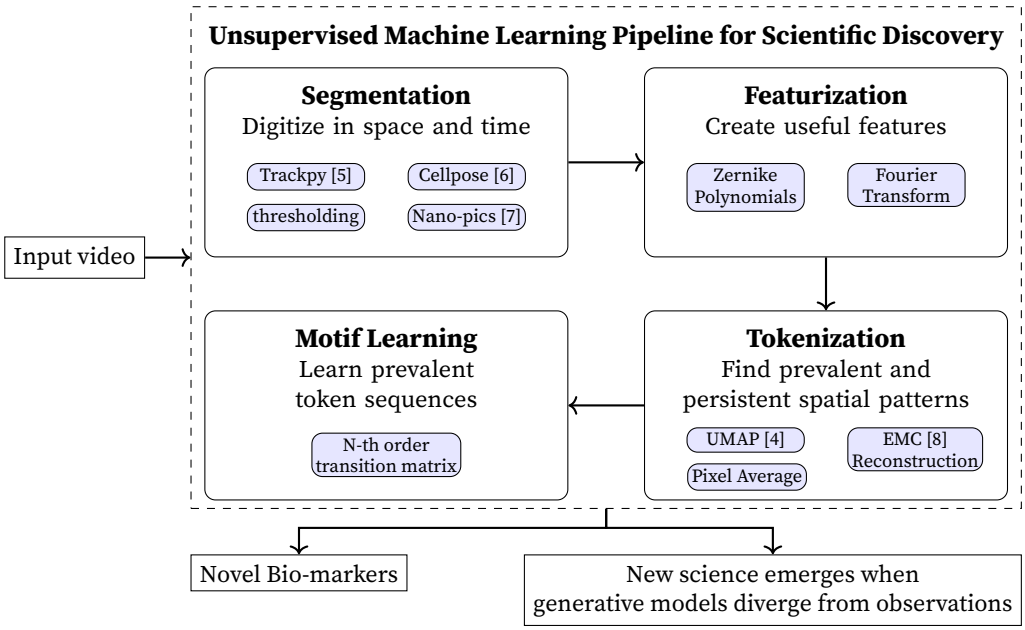

Fig. 1: High-level view of the unsupervised machine learning pipeline for scientific discovery. Input video data are processed through modular stages of segmentation, featurization, tokenization, and motif learning to identify persistent spatial patterns and recurrent token sequences. The blue rounded boxes indicate interchangeable method options within each stage, allowing different algorithms or representations to be swapped and compared without altering the overall pipeline structure. Outputs include the discovery of novel biomarkers and the identification of new scientific insights when generative models diverge from experimental observations.

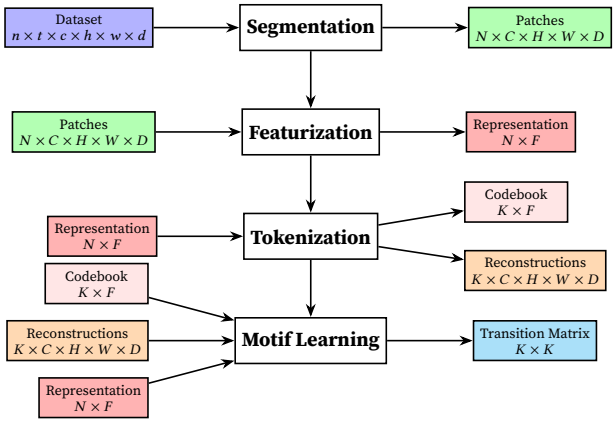

Fig. 2: Pipeline Architecture

A2. This lowers the barrier to exploration, reduces user error, and allows domain experts to focus on scientific interpretation rather than software implementation details.

Multi-domain datasets and tutorials: We provide a curated collection of example datasets spanning multiple application domains, including both experimental and synthetic data, each paired with validated configuration files tailored to the corresponding analysis requirements. For every reference dataset, step-by-step tutorials are provided to illustrate recommended pipeline choices and parameter settings. In addition, we supply a chatbot-ready tutorial and complete API documentation for the four core pipeline stages, enabling users to understand, reproduce, and adapt analyses across domains with minimal prior familiarity.

5. Conclusion

We introduced a modular and interpretable pipeline for unsupervised analysis of scientific spatiotemporal imaging data. By explicitly separating segmentation, featurization, tokenization, and motif learning, the framework enables systematic interrogation of modeling choices while preserving flexibility in algorithm selection. The use of unified data structures ensures interoperability across modules and supports reproducible comparison of alternative configurations. Together, these design choices lower the barrier to applying unsupervised learning to complex scientific datasets, promote interpretability, and facilitate scalable and AI assisted scientific discovery across domains.

Acknowledgments

The authors gratefully acknowledge the National University of Singapore (NUS) and the Institute for Digital Molecular Analytics and Science (IDMxS).

References

[1] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15979–15988, 2022.

[2] Alireza Nasiri and Tristan Bepler. Unsupervised object representation learning using translation and rotation group equivariant vae. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc.

[3] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning, 2020.

[4] John Healy and Leland McInnes. Uniform manifold approximation and projection. *Nature Reviews Methods Primers*, 4(1):82, 11 2024.

[5] John C. Crocker and David G. Grier. Methods of digital video microscopy for colloidal studies. *Journal of Colloid and Interface Science*, 179(1):298–310, 1996.

[6] Carsen Stringer, Tim Wang, Michalis Michaelos, and Marius Pachitariu. Cellpose: a generalist algorithm for cellular segmentation. *Nat. Methods*, 18(1):100–106, January 2021.

[7] Zhen Yuan Yeo, Eun Ho Song, Kyeongmee Lee, Jwa-Min Nam, and N Duane Loh. Biosensing using CNNs to detect noisy but persistent nanoparticle binding events on supported lipid bilayer systems. In *AI4X 2025 International Conference*, 2025.

[8] N Duane Loh. A minimal view of single-particle imaging with X-ray lasers. *Philos. Trans. R. Soc. Lond. B Biol. Sci.*, 369(1647):20130328, July 2014.

Appendix A. Human-readable pipeline configuration structure

```toml
[[load_data]] # creates dataset object from files
data_path = "data.npz" # data attribute
metadata_path = "metadata.csv" # metadata attribute
array_name_if_npz = "data"

[[preprocess]] # modifies data attribute in place
func = "hist_match_by_video" # histogram matching for each video
video_id_values="filename" # _values is a special suffix; it retrieves the values in video_id metadata column
frame_id_values="frame_id"

[[preprocess]]
func = "crop"
x1 = 20 # from 20
x2 = 70 # to 70
axis = 2 # along height axis

[[preprocess]]
func = "crop"
x1 = 20
x2 = 70
axis = 3 # along width axis

[[preprocess]]
func = "normalize_min_max" # along axes 2 and 3 by default

[[core]]
subtype = "create_representation" # creates a new representation
name = "fourier" # representation generated using fourier on the original dataset
source = "data" # the original data after pre-processing
func = "fft2" # also available: zernike
use_log1p = true # extracts the log from

[[core]]
subtype = "create_representation" # creates a new representation
name = "vae" # representation generated using fourier on the original dataset
source = "fourier" # the original data after pre-processing
func = "shallow_vae" # also available: zernike
verbose = true
compression_rate = 16

[[core]]
subtype = "create_representation" # creates a new representation
name = "pca"
func = "pca" # principal component analysis
source = "vae"
n_components = 32 # number of principal components

[[core]]
subtype = "create_metadata" # inserts a new metadata column
name = "cluster"
func = "k_means" # clustering function
source = "pca" # from the pca representation
random_state = 0
n_clusters = 32 # k=32

[[core]]
subtype = "create_representation" # creates a new representation
name = "umap"
func = "umap"
source = "pca"

[[core]]
subtype = "create_reconstruction" # reconstructs an image to illustrate each cluster
name = "avg" # reconstructs by averaging images in the cluster
func = "avg"
source = "data"
cluster = "cluster"

[[core]] # another technique using the expand maximize compress algorithm
subtype = "create_reconstruction"
name = "emc"
func = "emc_translation"
source = "data"
cluster = "cluster"
iter_count = 20
padding = true

[[viz]]
func = "two_dim_plot" # creates a figure
source = "umap" # from the umap representation
label = "width_nm" # what to highlight in color from the metadata
label2 = "cell_number" # what to represent with different shapes from the metadata
```

Fig. A1: TOML configuration extract

Each instruction block is referencing of one of the 4 core steps (or an I/O operation). The actual implementation varies depending on the function called and its parameters. Data structures are managed in the background, as such, *preprocess* instruction blocks apply to the input Dataset implicitly. *Core* instructions can either:

- create a new *Representation* by applying a transform to an existing Representation or the original Dataset
- create a *Codebook*, stored as a new metadata column
- create a *Reconstruction* from a Representation and a Codebook

Additionally, visualization functions *viz* handle representing low dimensional spaces with a variety of options, and handle the Motif Learning step.

Appendix B. AI-assisted pipeline generation

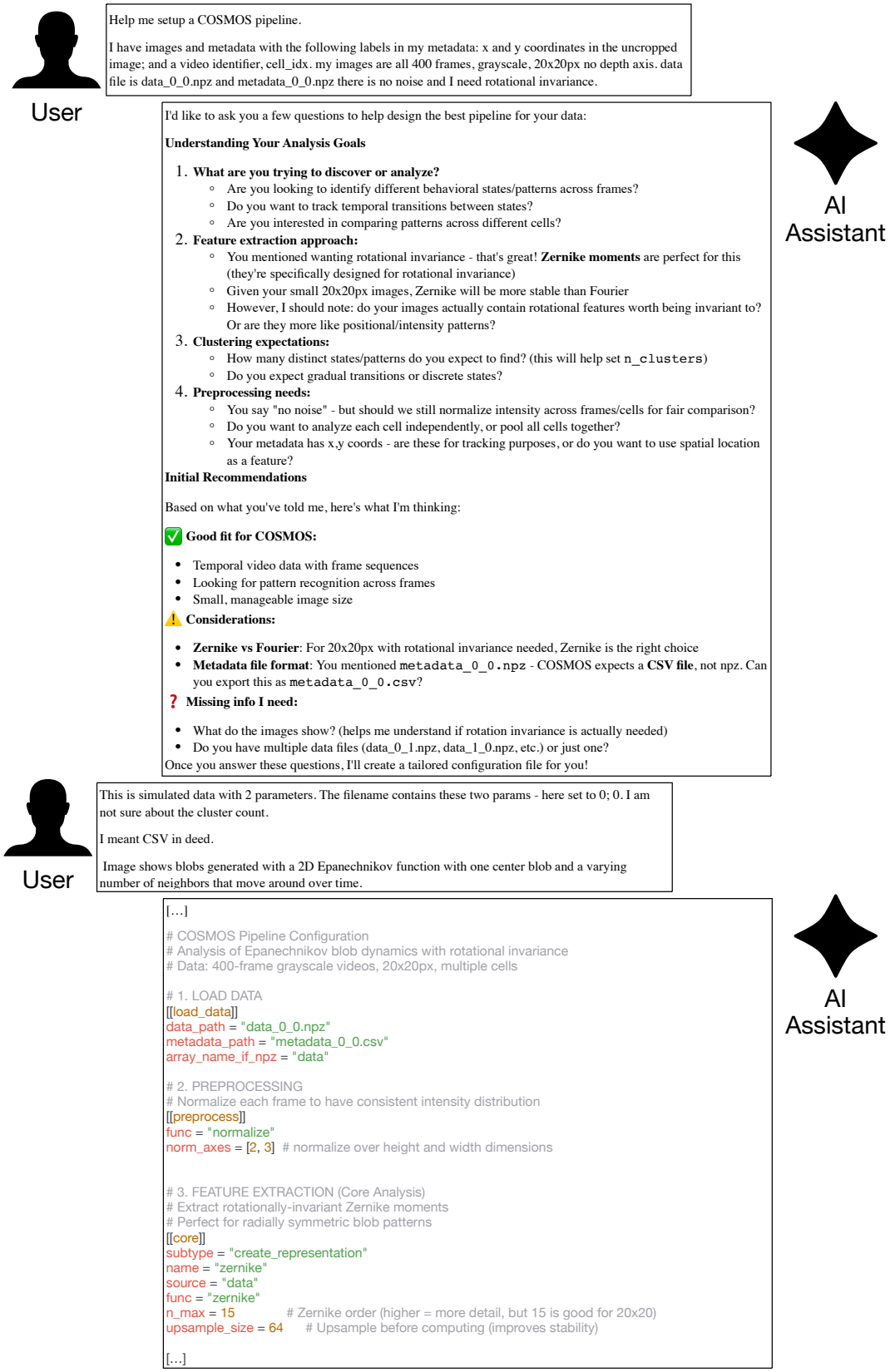

Fig. A2: AI-assisted pipeline generation transcript

From the instructions provided, the AI assistant first asks a set of questions to better understand the specificities of the user's dataset, then generates a configuration file for them. The configuration output has been cropped in this illustration.