Identifying Terbinafine Treatment in C. elegans Through Behavioral Phenotyping

Massimo BERARDI (345943), Eugène BERGERON (328786), Basile BULTEZ (407517)
CS-433 Machine Learning, Project 2, EPFL, Switzerland

***Abstract*—This study introduces a featureless machine learning framework to classify C. elegans nematodes treated with Terbinafine based exclusively on raw behavioral trajectories. By circumventing the limitations of manual feature engineering, we developed a custom Multiple Instance Learning architecture that integrates Convolutional Neural Networks with Gated Attention mechanisms. The primary contribution of this work lies in achieving superior predictive performance while simultaneously ensuring high model interpretability. Our approach attained an accuracy of 76% and an F1-score of 0.75, significantly outperforming baseline models reliant on handcrafted features which achieved only 68% accuracy. Furthermore, the embedded attention mechanism offers granular transparency, enabling the direct visualization of the specific trajectory segments that drive classification decisions.**

I. Introduction

Caenorhabditis elegans (*C. elegans*) is a fundamental model organism in aging research due to its short lifespan and well-characterized biology. This project focuses on the automated classification of *C. elegans* nematodes based on their longevity phenotypes. Specifically, we aim to classify worms treated with Terbinafine, a drug known to extend lifespan, and untreated control worms, using supervised learning based solely on behavioral patterns.

A. Problem Statement and Objectives

The core challenge is to distinguish drug-induced behavioral modifications from natural movement variability. Previous analysis established a baseline at 68% of accuracy, using manually engineered features such as aggregated speed and curvature statistics [1]. While predictive, this approach relies on expert selection and risks discarding subtle, non-linear patterns. To address this limitation, we investigate a featureless approach that leverages raw data representations via time-series embeddings and computer vision techniques.

Consequently, our work is driven by three key research questions:

1) Can machine learning models accurately predict the treatment group from behavioral data alone?
2) Can featureless architectures only using the time-series surpass the performance of traditional handcrafted feature baselines?
3) Do these models generalize to unseen worms without overfitting to individual movement quirks?

II. Exploratory Data Analysis

A. Dataset Overview

The dataset consists of longitudinal tracking data of adult nematodes moving on agar plates. The recording protocol captured the macroscopic worms' behavior throughout their entire adult lives with the following specifications:

- Sampling Rate: 1 frame every 2 seconds.
- Session Structure: 30-minute recording sessions (900 frames) repeated every 6 hours.
- Raw Features: Centroid coordinates (X, Y), instantaneous speed, and timestamps.

B. Challenges and Data Characteristics

Preliminary analysis revealed several critical characteristics that shaped our methodology:

1) Dataset Size: The dataset is composed of tracking data for 52 treated worms and 52 untreated worms in the control group. This limited quantity of samples may limit the ability of high-capacity models to generalize effectively to new populations. There is a heightened risk that complex architectures might memorize the idiosyncratic movement quirks of specific individuals rather than learning the robust, treatment-induced behavioral shifts.

2) Limitations of Macroscopic Behavioral Data: In this study, the analysis is restricted to macroscopic behavioral data derived from centroid-based tracking of *C. elegans*. While this level of observation captures global locomotion patterns and plate-scale dynamics, it inherently discards finer-grained biological information. In contrast, microscopic data can provide access to detailed phenotypic markers such as body posture and curvature dynamics, head foraging and pharyngeal pumping rates, muscle contraction patterns, or neuronal activity. The absence of such high-resolution information may limit the sensitivity of our approach to subtle drug-induced effects that manifest at the neuromuscular or cellular level before becoming detectable in coarse behavioral metrics.

3) Variable Length: Given that Terbinafine is expected to extend lifespan, treated worms may produce longer trajectories and more data points than control worms. This would create a potential variable sequence length problem, where simple concatenation could bias the dataset toward treated samples. However, we found that the Terbinafine treatment doesn't achieve significant lifespan extension on average (additional details in Appendix A.1).

4) Tracking Noise: The raw tracking data contain substantial artifacts arising from both initialization errors and tracking failures. These include loss of signal, instances where the worm temporarily disappears from the plate (e.g., due to dead angles), abrupt jumps between extreme positions, and misidentification of the worm by the tracker, leading to physically impossible displacements. Additionally, the system occasionally detects multiple worms on a single plate, despite only one being present, or incorrectly reports the worm as stationary at a fixed location while video inspection confirms that it is actively moving elsewhere on the plate.

5) Spatial Bias: We considered the possibility that worms might exhibit specific "spot preferences" on the plate (e.g., remaining in nutrient-rich or specific zones). This introduces a risk that models might learn to classify based on spatial location (X, Y) rather than motility dynamics, motivating the use of robust spatial augmentation as a precaution.

6) Data Leakage Risk: Since each worm contributes multiple sessions, a standard random split would place segments from the same individual in both training and validation sets, leading to identity-based overfitting.

III. Pre-Processing

To address the noise and bias identified in the EDA (II-B), we implemented a rigorous pipeline ensuring data quality and rigorous validation.

A. Cleaning and Reconstruction

We applied a multistage filter to the raw trajectories:

- Artifact Removal: First-row initialization artifacts were removed, and frames recorded after the annotated death were clipped.
- Trajectory Smoothing: To correct tracking jumps, we filtered frames where sudden displacement exceeded a threshold (> 16 pixels/frame). This value was determined empirically by visually comparing multiple cutoffs (additional details in Appendix A.2) and selecting the highest threshold that effectively filtered out evident tracking artifacts.
- Path Reconstruction: We reconstructed continuous trajectories by computing the cumulative sum of valid displacements, ensuring biologically plausible paths.

Additionally, we experimented with normalizing the data as done by the previous analysis, and standardizing as well; however, both methods achieved poor results when combined with the reconstruction pipeline.

B. Segmentation and Consistency

To handle the variable lifespan issue, continuous tracks were divided into fixed-length segments of 900 frames. This standardizes the input for sequential models and allows for analysis at specific life stages. Post-segmentation, we recalculated instantaneous speed.

C. Preprocessing for computer vision

Trajectories were segmented into short, clean temporal clips using a sliding window approach, ensuring that each clip contains continuous motion without missing values or tracking discontinuities. Global normalization statistics were computed across the dataset for locomotion speed and spatial scale. Each clip was then centered, spatially normalized, and encoded as a three-channel image representing speed, temporal progression, and trajectory occupancy, producing a compact and consistent input suitable for computer vision-based analysis.

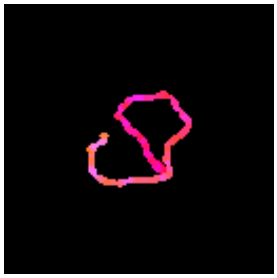

Figure 1. Example of a trajectory clip encoded as a three-channel image.

D. Validation Strategy

To prevent data leakage, we employed Stratified Group K-Fold Cross-Validation: Data was grouped by `WormID`, ensuring that all segments from a single worm reside exclusively in either the training or validation fold, while maintaining class balance inside the splits.

Upon completion of the preprocessing pipeline, the data is structured as a collection of multivariate time series segments. Each sample consists of multiple segments of 900 frames with 3 channels: `X`, `Y`, and `Speed`, labeled as treated or untreated. This structured dataset serves as the standardized input for all subsequent modeling pipelines except the computer vision one.

IV. Model design and selection

Our model selection came from 3 breakthroughs:

A. Feature extraction using convolution kernels

Our first major finding happened when trying the ROCKET architecture [2], which consists of random convolutions kernels producing an embedding vector for each variate. The embedding vector simply contains two values per kernel: a global max pooling and the proportion of positive values produced by the kernel.
Note: from here we denote as an "embedding" a vector that is a representation of the data in a latent space.

Coupled with Logistic Regression, the model gave immediately results above the given baseline. We made an analysis that lead us to two reasons for this results:

- Kernels differentiate the X and Y positions across time, making possible to access axis-wise velocities and accelerations. Additionally kernels extract high and low frequencies of the data. Those features can be essential when analyzing a trajectory.

- The convolution kernels increase the Signal to Noise Ratio (SNR). It has already been shown [3] and demonstrated [4] in previous works. In our case, as seen in II-B, the trajectory data present a lot of noise. The kernels creating matched filters [4] helps extract the real signal from those noises.

B. Data augmentation

As mentioned in subsection II-B5, there is a possibility for an overfitting at the dataset level, meaning that validation results could be biased and not report the real generalization capability of the model. This has also been reported by another group working on the same task, which achieved around 0.9 accuracy using k-Nearest Neighbors techniques. The accuracy collapses when performing rotations on the data.

To counter this effect, we found inspiration from self supervised techniques like SimCLR [5], that leverages data augmentation through transformations of samples. Following previous works [6] that validated the feasibility of spatiotemporal data augmentation, we augmented our data in a way that would not break the meaningfulness of the trajectories and not disturb the signals of the variates.

Therefore we implemented the following transformations:

- Rotations of X and Y by a random θ: $(X', Y')^{\mathrm{T}} = \mathbf{R}(\theta)(X, Y)^{\mathrm{T}}$ where $\mathbf{R}(\theta)$ is the standard rotation matrix by θ.
- Translations on X and Y by a random offset.
- Random scaling of every variate (random from 0.8x to 1.2x).

During the training phase, the models are exposed to original samples and augmented samples, which reduces the risk of overfitting by only looking at a specific spot in the plate, for instance. Utilizing those transformations, we increased the robustness of the model.

C. Attention is really all you need

Our task can be seen as a Multiple Instance Learning (MIL) framework: each worm is a bag and the instances are the segments [7]. We want to make a bag-level prediction by using multiple instances that can vary in number. Our literature review led to two architectures: TAIL-MIL [8] and TimeMIL[9]. Since TAIL-MIL achieved better performances across a larger set of time-series MIL tasks, we focused on this architecture to create a custom implementation.

Our model architecture consists of 3 steps:

- Feature extraction using CNNs, producing dimension reduced embedding for each variate in each segment.
- Two attention based pooling, one at the variate level, and another one at the time level. This produces a unique embedding that represents the whole sample.
- Prediction using a Multi-Layer Perceptron (MLP) head.

Details about the implementation can be found in the appendix, see VII-F.

Originally, TAIL-MIL is used to make predictions at the variate level, and then aggregate every prediction using attention based pooling (called *conjunctive pooling*) using positional encoding to make the attention time-aware. This implementation in our setup failed to converge, depicting clear underfitting, so we opted for an embedding-based model to augment the number of parameters in hidden layers. We kept the same idea, but instead of making predictions at variate level, we produced embeddings using CNNs (see IV-A) and then performed a weighted sum using Gated Attention [10] to aggregate them.

Classic attention led to pretty diluted attention weights, making them useful for the model but not as much as expected, especially for the interpretation part. Using gated attention forces the model to respect two conditions to produce high attention weights.

Formally, for an embedding $\mathbf{h}_k$, the attention weight is computed as: $a_k = \mathbf{w}^T(\tanh(\mathbf{V}\mathbf{h}_k) \odot \mathrm{sigm}(\mathbf{U}\mathbf{h}_k))$. The $\tanh$ component extracts the feature content, whereas the sigmoid gate makes a learnable filter. This multiplicative interaction ($\odot$) makes high attention rarer: a segment contributes to the final prediction only if it is feature-relevant and allowed by the gate. To respect the time awareness, a positional encoder value is added to embedding values when performing time pooling.

Attention comes with two very important and interesting features: **transparency** and **explainability**. Since we compute attention weights for each variate, and then for each time segment, we can easily visualize what part of the data has been more valued by the model to make its prediction. This allows to visualize directly which segments are more meaningful to the model, allowing for high-level interpretations of the predictions.

V. Training methods

A. Training pipeline

We trained our models using the Binary Cross Entropy Loss[11], as common for supervised binary classification tasks. Data augmentation is performed, then we split the data into a train and a validation split with ratio 80/20 of the worms. Since we perform the split at the worm level, the validation set contains only unseen worms by the model. Each model is fed with the training split and evaluated on the validation split, such that each model is trained and evaluated on the same data, assuring robust comparisons.

We perform this procedure 5 times for robustness and report the average accuracy and F1 score in VI.

B. Hyperparameters search

ROCKET only has one hyperparameter being the number of kernels used. With some testing, the optimal number of kernels we found was 1000.

For our custom model, we have tested several model architectures and modified each layer extensively. The presented architecture (details in Appendix VII-F) is the one that gave the best performances. For this architecture, there are still 3 hyperparameters:

- The embedding vector dimension: We have tested 4, 8, 16, 32 and 64. Dimension 4 fails to converge, while 32 and 64 showed quick overfitting even when maintaining a low learning rate. We therefore kept 8 and 16 for our testings and the former was kept as giving the best scores.
- Learning rate: The best learning rate found, similar as in the original TAIL-MIL architecture, has been $1e^{-4}$. We have tested values from $1e^{-2}$ to $1e^{-6}$ and $1e^{-4}$ achieved the good balance between convergence speed and stability.
- Batch size: Ranging from 1 to 64, the value giving the best stability has been 32.

VI. Discussion about our results

Table I
Performances comparison between the models performing worm-level predictions

Model	Accuracy	F1-Score
Our model	**0.76**	**0.75**
Baseline HC*	0.68	0.72
ROCKET + Log Reg	0.63	0.63
Computer vision	0.58	0.69
Baseline TS**	0.55	0.50
Naive Log Reg	0.46	0.45

*Previous work using hand-crafted features
**Using time series

It is worth noting that a decision made at random achieve 0.5 of accuracy, since the dataset is balanced. Our model inspired by TAIL-MIL clearly achieves better results than the baselines. Also the architecture is usable very easily in a regression setup to predict the lifespan of a worm, so we consider it a good progress towards this objective.

The computer vision model didn't perform very well, probably because data is lost in the transition between the raw time-series and the visualization. The model and preprocessing could be improved, but we decided to focus our efforts on our custom model which already achieved good performances.

Finally, the predictions from our model can be interpreted using the attention weights of the model, making the whole system transparent (see example in appendix A.4).

VII. Conclusion and future work

We believe that we came up with a strong base model that can be used later for lifespan prediction. Also, since the task is in a scope of understanding what behavioral data of *C. Elegans* lead to the predictions, the transparency of our model is a great advantage.

Further work needs to be done to validate the model on more data. We also believe that the model can be refined, with a better feature extraction layer, leveraging ROCKET for instance. Finally, interpretation of the predictions with experts should be performed to assess the quality and relevance of the predictions made.

Ethical risks

A. Risk Identification and Stakeholder Impact

While our project focuses on a model organism, the theoretical extension of behavioral phenotyping to humans poses an ethical risk regarding non-consensual surveillance. However, our model was trained exclusively on Terbinafine, a drug with no abuse potential, meaning it is unlikely to generalize to substances of higher ethical concern. Nevertheless, if transferable, "featureless" classifiers detecting drug influence via movement could theoretically lead to workplace surveillance, discriminatory insurance practices, or misuse in anti-doping screening.

B. Risk Assessment and Evaluation

We assess the likelihood of this risk as extremely low due to the vast biological gap between species. *C. elegans* movement is a direct readout of a simple, fully mapped nervous system of exactly 302 neurons [12] , whereas human behavior is governed by billions of neurons and layered with complex social and psychological factors. Consequently, features learned by our model (e.g., micro-turns in a Petri dish) are specific to nematode hydrodynamics and are not transferable to human kinematics.

C. Mitigation and Constraints

Given the negligible probability of transferability, the biological complexity gap acts as a natural safeguard, rendering algorithmic constraints like differential privacy unnecessary. Active mitigation would only be required if we identified evolutionarily conserved features detectable in humans. In such a scenario, we would consult an ethics committee and obfuscate high-risk features; however, the current biological barrier makes this unnecessary.

References

[1] Lysandre-c, "Lpbs-celegans: Analysis of c. elegans lifespan data," https://github.com/lysandre-c/LPBS-Celegans, 2023.

[2] A. Dempster, F. Petitjean, and G. I. Webb, "Rocket: exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, p. 1454–1495, Jul. 2020. [Online]. Available: http://dx.doi.org/10.1007/s10618-020-00701-z

[3] G. Yin, Z. Zhang, and B. Zhang, "Convolutions layers increase snr of shallow networks," *arXiv preprint arXiv:2402.14713*, 2024. [Online]. Available: https://arxiv.org/abs/2402.14713v2

[4] L. Stankovic and D. Mandic, "Convolutional neural networks demystified: A matched filtering perspective based tutorial," 2022. [Online]. Available: https://arxiv.org/abs/2108.11663

[5] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020. [Online]. Available: https://arxiv.org/abs/2002.05709

[6] Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu, "Time series data augmentation for deep learning: A survey," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, ser. IJCAI-2021. International Joint Conferences on Artificial Intelligence Organization, Aug. 2021, p. 4653–4660. [Online]. Available: http://dx.doi.org/10.24963/ijcai.2021/631

[7] J. Early, G. K. Cheung, K. Cutajar, H. Xie, J. Kandola, and N. Twomey, "Inherently interpretable time series classification via multiple instance learning," 2024. [Online]. Available: https://arxiv.org/abs/2311.10049

[8] Y. Yao, J. Wang, H. Wu, Y. Chen, J. Lv, and S. Wu, "Tail-mil: Transformer-based attentive interactive learning for multiple instance learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 9244–9251. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/33933

[9] Y. Ma, G. Zhou, Y. Zhao, J. Wang, C. Zhang, X. Li, Y. Zhang, and Y. Sun, "Timemil: A multi-instance learning framework for time series classification," *arXiv preprint arXiv:2405.03140*, 2024. [Online]. Available: https://arxiv.org/abs/2405.03140

[10] S. Hong, Y. Zou, and W. Wang, "Gated multi-head attention pooling for weakly labelled audio tagging," in *Interspeech 2020*, 2020.

[11] A. Mao, M. Mohri, and Y. Zhong, "Cross-entropy loss functions: Theoretical analysis and applications," 2023. [Online]. Available: https://arxiv.org/abs/2304.07288

[12] O. Hobert, "Neurogenesis in the nematode Caenorhabditis elegans," in *WormBook*, T. C. elegans Research Community, Ed. Pasadena, CA: WormBook, 2013, doi: 10.1895/wormbook.1.12.2. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK116086/

APPENDIX

D. *Terbinafine effect statistics*

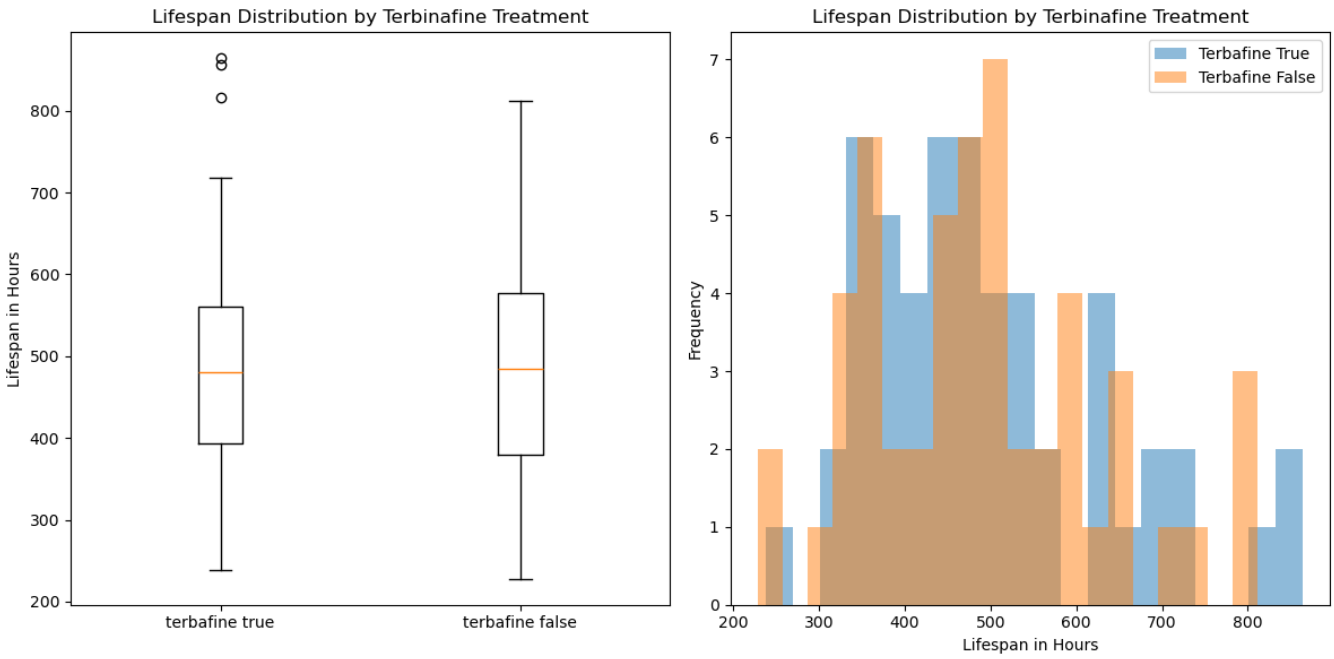

Figure A.1. Distribution of worms lifespan according to Terbinafine treatment. The boxplot (left) contains the median (orange line), first and third quartile represented by the box (called IQR), whiskers expands to the farthest data point within $1.5 \times IQR$, dots are outliers. The histogram (right).

E. Worm trajectories in function of the preprocessing

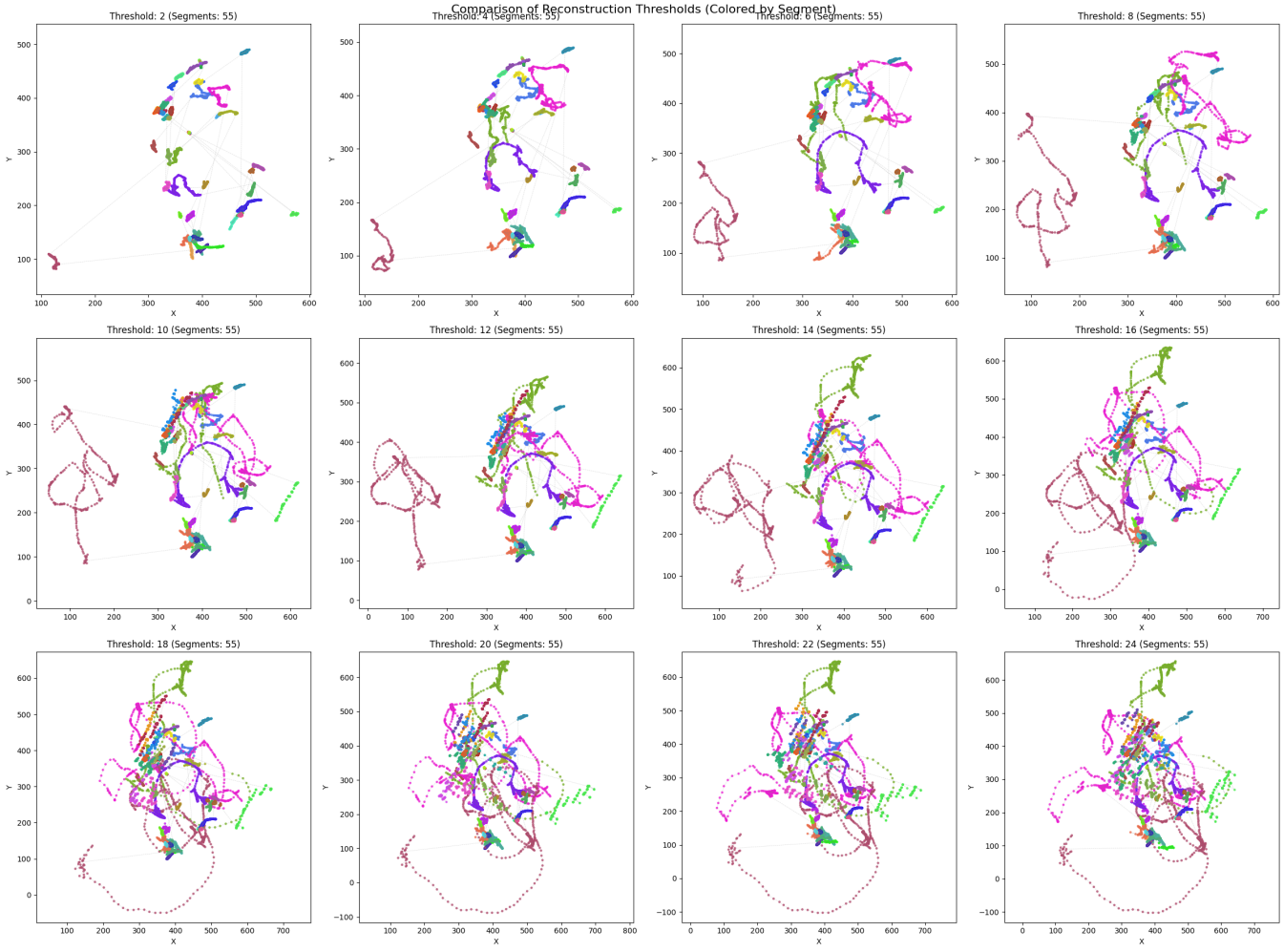

Figure A.2. Comparative visual trajectories of the same worm with various distance cutoff values.

F. Our model architecture

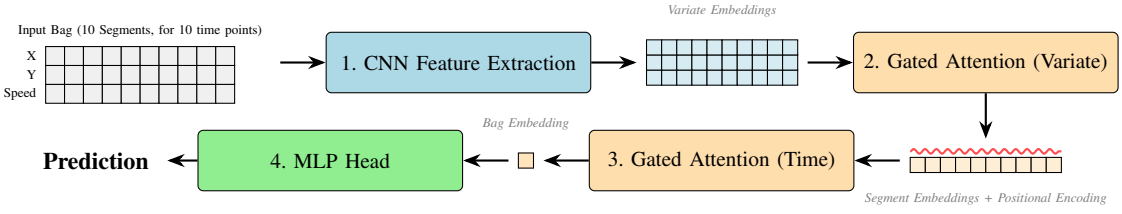

Figure A.3. TAIL-MIL Inspired Architecture

- **CNN Feature Extraction**: 3 convolution kernels of width 7, 5 and 3, with stride 2. They map the input from 1 to 32 channels, then from 32 to 64 then back to the embedding dimension (8 in our case). An adaptive max pooling is performed to reduce the output of the convolutions to a 1D vector. Finally, a fully connected layer is applied.
- **Gated Attention**: As explained in IV-C, for an embedding vector $\mathbf{h}_k$, the attention weight is computed as: $a_k = \mathbf{w}^T(\tanh(\mathbf{V}\mathbf{h}_k) \odot \text{sigm}(\mathbf{U}\mathbf{h}_k))$. Then we compute $\mathbf{h} = \sum_1^E \mathbf{h}_k a_k$ as the dimension (variate, or time) embedding,

with E the embedding dimension (in our example: 3 for the variate embedding, 10 for the segment embedding). The positional encoding is a learnable vector for the model, initialized at random.

- **MLP Head**: The head is composed of 2 fully connected layers with a ReLU function in the middle to create the non-linearity. We normalize the prediction at the end with a sigmoid function, but this was not really necessary and just an artifact of previous implementations. Formally: $\hat{y} = \sigma\left(\mathbf{W}_2 \cdot \text{ReLU}(\mathbf{W}_1\mathbf{x} + b_1) + b_2\right)$, with $\mathbf{x}$ the final embedding produced by the previous layers.

G. *Trajectories attention*

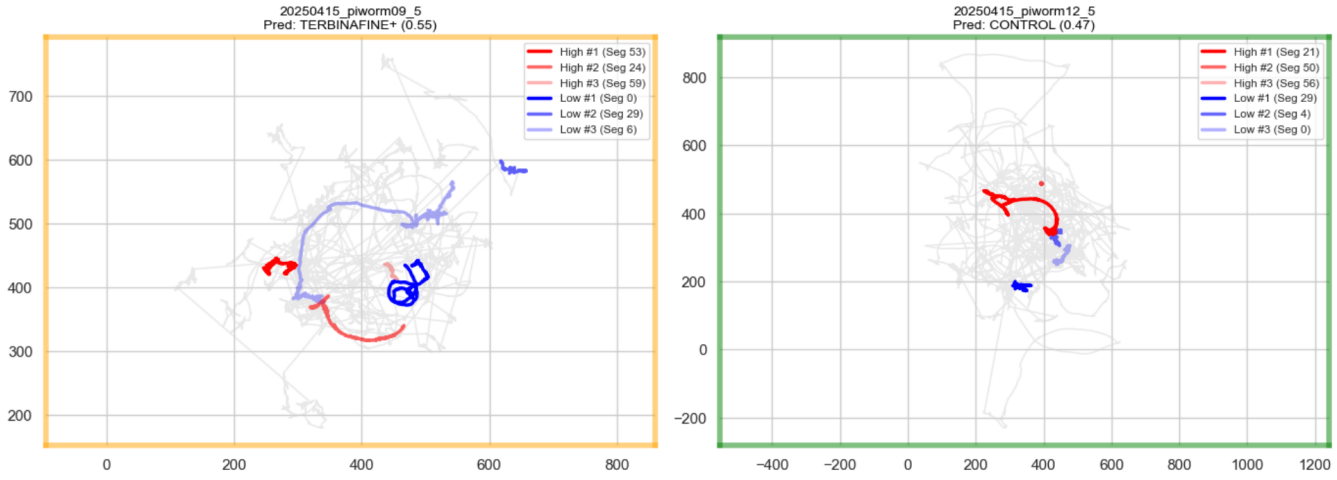

Figure A.4. Visualization of the trajectories of two worms. At the left, a drugged worm, predicted as such. At the right, a control worm predicted as non-drugged. **Highlighted in red the segments with the highest attention weights, and in blue smallest.** Low attention weight means that the segment embedding contributes less to the final decision, and vice-versa for high attention weights.