Learning and Explaining Semantic Concepts in Deep Time Series Models

Matilde Silva[1], **André Carreiro**[1,2], **Duarte Folgado**[1,2]

[1]Fraunhofer Portugal AICOS, Porto, Portugal
[2]Comprehensive Health Research Center (CHRC), Porto, Portugal
{matilde.silva, andre.carreiro, duarte.folgado}@fraunhofer.pt

Abstract

Symbolic representations have a long history in time series analysis, offering an implicit form of interpretability. However, modern deep learning approaches for time series classification have largely moved away from symbolic methods in favor of end-to-end modeling. Yet both symbolic representations and, more importantly, high-level semantic concepts, remain central to how humans reason about temporal patterns. In this work, we revisit the role of symbolic representation in time series and propose a framework that integrates semantic, high-level concepts into deep learning models via concept-augmented training. These concepts describe segment-level trends, such as *rising*, *falling*, or *plateau*, and enables the model to learn internal representations that align with human-understandable concepts. Additionally, we extend an attribution method to produce post-hoc concept explanations, mapping neuron-level relevance to atomic and composite temporal concepts, providing insight into which concepts influenced model decisions. To evaluate our approach, we implemented a controlled experimental setup using semi-automatically annotated synthetic signals and demonstrated strong generalization in identifying high-level concepts. By offering concept-supervised learning and concept-level post-hoc explanations, our framework provides a comprehensive approach for interpretable time series modeling and contributes a step toward enabling the community to explore concept-level reasoning for interpretable time series models.

1 Introduction

Human reasoning often employs *concept-based thinking*, whereby individuals abstract commonalities from multiple examples and organize them into coherent categories (Armstrong, Gleitman, and Gleitman 1983; Tenenbaum 1999). The ability to distill shared structure is central to understanding and navigating the world; indeed, as Gregory Murphy noted in *The Big Book of Concepts*, "concepts are the glue that holds our mental world together" (Murphy 2002). Translating this cognitive process into interpretable machine learning could greatly enhance model transparency, especially in high-risk domains.

Despite the advances in model interpretability for time series, existing methods indicate *where* a model focuses in time but not *what* those regions represent. Yet, time series often carry inherent semantic structure, and their patterns naturally convey interpretable concepts. For instance, two classes (1)  and (2) differ because one class has a long *rising* pattern and the other has a long *falling* pattern. This observation suggests that concept-level reasoning could provide a natural bridge between human understanding and model explanations.

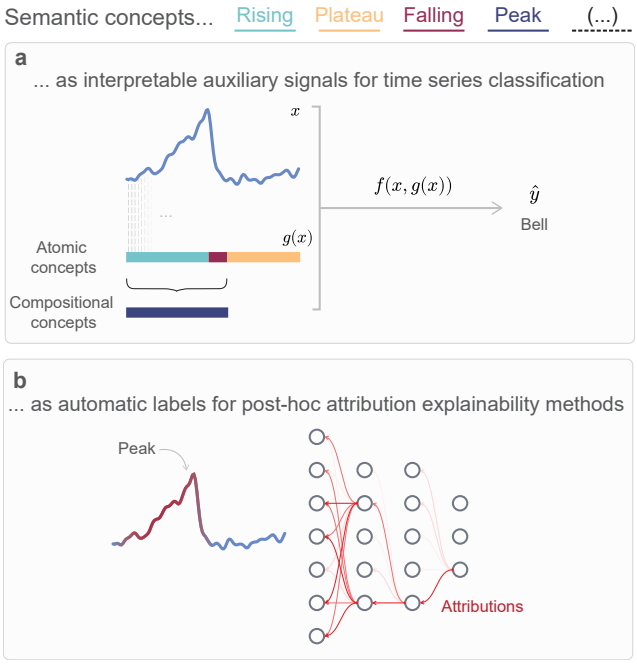

Figure 1: Concept-based framework for time series explainability. (a) A dense-labeling model g automatically extracts high-level semantic concepts from raw time series, producing interpretable auxiliary signals for a downstream classifier f. High-level concepts are organized hierarchically. At the base we consider rising, falling, and plateau, from which more complex compositions, such as peak, can emerge. (b) The same learned concepts serve as automatic labels to annotate post-hoc attribution maps, aligning model explanations with human-understandable patterns. Red arrows indicate backward relevance propagation, highlighting concept-relevant regions in the signal.

We propose a concept-based framework for time series explainability (Figure 1). We wondered: *Can high-level, domain-agnostic concepts be integrated to better understand deep time series models?* To explore this question, we investigate two complementary pathways: (1) inspired by Concept Bottleneck Models (CBMs) (Koh et al. 2020) from the computer vision domain, we propose a hybrid framework that fuses raw input representations with semantically meaningful concept-level annotations, and (2) we develop human-understandable post-hoc explanations that reveal what internal units of deep models have learned. Our proposed framework first extracts high-level concepts from time series using a dedicated concept-based dense-labeling model. Then, these extracted concepts (i.e., concept annotations) are either integrated with the raw time series into a second predictive model or used to interpret a black-box model through post-hoc analysis. Our approach is domain-agnostic and applicable to a broad range of sequence learning tasks. In summary, our contributions are as follows:

- **A concept-annotated synthetic dataset for benchmarking concept annotation in time series.** We introduce a controlled signal generator that produces time series labeled with intuitive temporal concepts, namely *rising*, *falling*, and *plateau*. This dataset enables quantitative evaluation of concept-based learning approaches and supports reproducible comparisons.
- **A concept-based dense-labeling model for high-level semantic labeling in time series.** We adapt the U-Time architecture for dense labeling of intuitive temporal concepts, introducing architectural modifications tailored to per-timestep concept labeling. We validate its performance against symbolic and change-point detection baselines.
- **A hybrid, two-branch deep model that integrates learned concepts with raw time series inputs.** We demonstrate that our proposed architecture (combining high-level semantic concepts representations with H-InceptionTime) achieves comparable predictive performance to the baseline across the UCR archive.
- **An automatic post-hoc concept labeling framework for neural units.** We combine our dense-labeling model with an attribution method to automatically assign human-interpretable concept labels to individual channels and neurons, providing insight into what each component of a black-box model has learned.

2 Related Work

2.1 Interpretability and Explainability in Time Series

Symbolic and Dictionary-based Representations. Symbolic and dictionary-based methods encode temporal patterns as sequences of symbols or short motifs, providing an interpretable abstraction of the signal. Early approaches (Horowitz 1975; Giese, Bourne, and Ward 1979; Udupa and Murthy 1980; Trahanias and Skordalakis 1990) relied on primitive extraction, linguistic representations, or grammar-based formulations to describe waveform structures. While conceptually appealing, these methods were difficult to scale, sensitive to noise, and highly dependent on domain knowledge.

Later methods introduced more scalable and data-driven symbolic representations. Methods such as SAX (Lin et al. 2003, 2007), SFA (Schäfer and Högqvist 2012), and MrSEQL (Le Nguyen et al. 2019) convert time series into compact symbolic sequences, enabling efficient comparison and classification. Dictionary-based methods such as BOSS (Schäfer 2015) and WEASEL (Schäfer and Leser 2017) further capture recurring local patterns through symbolic word histograms. Shapelet-based methods similarly identify discriminative subseries as symbolic motifs (Ye and Keogh 2009). Although recent variants such as WEASEL V2 (Schäfer and Leser 2023) achieve higher accuracy and efficiency, these models still rely on low-level tokens (such as short symbolic motifs or frequency-based patterns), which are symbolic and not inherently semantically interpretable.

Post-Hoc Explainability in Deep Time Series Models. Post-hoc methods aim to explain black-box models by attributing importance to input features (Bordt et al. 2022; Lipton 2018). Layer-wise Relevance Propagation (LRP), which computes backpropagated relevance scores, has been extended in several directions, including DFT-LRP (Vielhaben et al. 2024), which operates in the frequency domain, and Concept Relevance Propagation (CRP) (Achtibat et al. 2023), originally proposed for images but also applicable to time series. CRP introduces conditional relevance maps, which disentangle internal representations and reveal which channels contribute most to a given prediction. Moreover, CRP can further identify the channels most relevant to a target class and retrieve representative reference samples with the highest relevance for those channels. While these post-hoc methods provide valuable attribution insights, they typically operate at the level of raw input features or latent activations, without explicitly linking these patterns to textual representations of human-understandable concepts.

Concept-based Intrinsic Interpretability. CBMs (Koh et al. 2020) introduce intermediate, human-defined concepts through which predictions are made, offering intrinsic interpretability. While widely adopted in computer vision, CBMs often trade off predictive performance for interpretability. Recent works relax the bottleneck constraint by treating concepts as auxiliary signals via skip connections, improving performance while maintaining interpretability (Opsahl and Antun 2024). However, their application to time series remains limited, likely due to the lack of concept annotations and the variable nature of temporal signals.

Recent works have started to explore this gap. van Sprang, Acar, and Zuidema (2024) extended CBMs to time series Transformers by enforcing concept representations within attention or feed-forward layers. Yet, these latent features are not inherently interpretable, and attention mechanisms themselves can be unreliable, with redundant heads (Michel, Levy, and Neubig 2019; Voita et al. 2019).

Literature Gap. While these recent efforts demonstrate the potential of concept representations for time series classification, they are mostly confined to domain- or task-specific concepts. As a result, a general, domain-agnostic framework for integrating interpretable concepts into time series classification remains largely unexplored.

2.2 Timestep-wise Segmentation and Labeling

To incorporate semantic concepts into deep learning, time series must be segmented and labeled into meaningful intervals. Segmentation methods include symbolic, rule-based, change-point detection, and deep learning approaches.

TSAX (Zhang et al. 2018) attempts to encode higher-level trend changes through the subsegmentation of SAX representations, enabling more transparent reasoning. Symbolic Search in Time Series (SSTS) (Rodrigues et al. 2019) leverages symbolic representations to identify meaningful segments based on user-defined regular expression queries. However, these methods are constrained by handcrafted thresholds and a focus on local patterns, which limits their scalability and generalization to diverse time series.

Change-point detection algorithms include binary segmentation (Bai 1997), which recursively identifies the most significant change-point but can be computationally intensive; bottom-up segmentation (Keogh et al. 2001), which merges fine-grained segments iteratively; and PELT (Killick, Fearnhead, and Eckley 2012), which optimizes segmentation efficiently via pruning.

Among Deep Learning (DL) models, U-Time (Perslev et al. 2019) is a fully convolutional neural network adapted from the U-Net framework for time series data, originally developed for sleep-stage classification.

3 Methods

This section details the proposed framework, comprising (1) the generation of a concept-annotated time series dataset, (2) methods for automatic concept annotation, (3) classification architectures that integrate raw time series with concept information, and (4) concept-based explanations.

3.1 Concept-Annotated Synthetic Dataset

We adapted the synthetic time series generator of Miot and Drigout (2020) to produce time series with ground-truth concept annotations derived from the full temporal context. This setup captures global patterns of interest while suppressing local fluctuations dominated by noise.

The generator produces piecewise-linear time series with configurable parameters such as segment slope and noise level. This flexibility enables the creation of diverse signal combinations and controlled ambiguities, allowing systematic evaluation of how models handle uncertain transitions between concepts.

We focused on three high-level atomic concepts, motivated by reasoning abstractions found in many real-world applications. Coarse directional trends such as rising, falling, and plateau frequently correspond to meaningful events. For instance, a rising heart rate can indicate physical exertion or stress, while a falling respiratory rate may suggest relaxation. We generate coherent trend labels by smoothing local slope fluctuations, applying adaptive thresholds, and merging short or inconsistent segments to enhance temporal continuity (see Appendix A for details).

3.2 Automatic Concept Annotation

We formulate concept annotation as a per-timestep labeling task, assigning a semantic label to each point in the time series. Using the synthetic dataset with ground-truth annotations, we benchmark three categories of methods under a supervised setting:

- **Change-point detection methods:** We evaluated Pruned Exact Linear Time (PELT), Bottom-Up, and Binary Segmentation, assigning labels by computing the slope within each detected segment and applying thresholding.
- **Rule-based segmentation methods:** We evaluated SSTS (Rodrigues et al. 2019) and TSAX (Zhang et al. 2018), which partition time series into segments defined by directional trend patterns (rising, falling, plateau). SSTS segments using smoothed derivatives, whereas TSAX captures trend transitions through symbolic subsegmentation of SAX representations.
- **U-Time:** We modified U-Time (Perslev et al. 2019) for fine-grained predictions by removing average pooling, adding padding to align input and output lengths, incorporating dropout, and reducing model depth.

For both rule-based and the change-point detection methods, we included signal preprocessing using smoothing windows of varying sizes and post-processing to remove spurious segments (i.e., smaller than a predefined threshold).

3.3 Univariate Time Series Classification

Rather than adopting the original CBM architecture, we treat concepts as complementary signals to the raw time series, following the principle of CBM-Res and CBM-Skip, which integrate concept embeddings through skip connections at the prediction layer in vision tasks (Opsahl and Antun 2024).

To quantify the impact of concept information on classification, we evaluate four model variants (Figure 2):

- **Baseline:** A H-InceptionTime (Ismail-Fawaz et al. 2022) model trained directly on raw time series[1], without concept supervision.
- **Concept-Augmented Baseline (CAB):** An H-InceptionTime model that integrates global atomic concept information (rising, falling, and plateau) by concatenating time-averaged concept probabilities from a pretrained U-Time model into the final feature layer. It reflects the overall concept prevalence while ignoring temporal variations within the signal.
- **Dual Encoder – atomic (DE-A):** A hybrid architecture that processes raw signals and concept signals in parallel. An H-InceptionTime encoder processes the raw time

[1]The original H-InceptionTime implementation ensembles five classifiers. For computational efficiency, we use a single classifier.

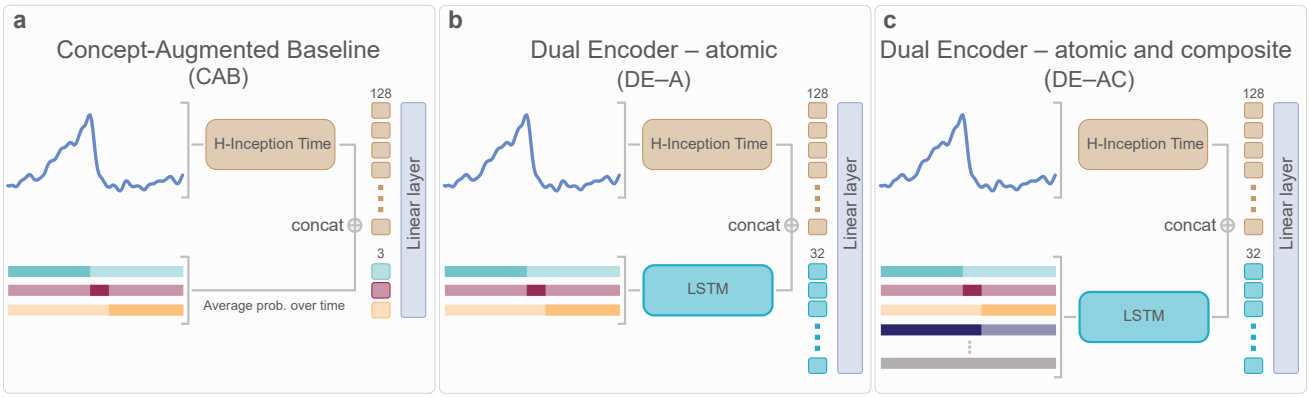

Figure 2: Overview of the proposed model architectures. (a) Concept-Augmented Baseline (CAB), which augments the raw input with global predicted concept probabilities; (b) Dual Encoder – atomic (DE-A), which jointly encodes raw signals and atomic concept representations; and (c) Dual Encoder – atomic and composite (DE-AC), an extension of (b) that also integrates compositional concepts such as *peaks*.

series, while an LSTM encodes the full sequence of concept probabilities from U-Time. The outputs of both encoders are fused before classification, allowing the model to integrate local signal features with the temporal evolution of semantic concepts.

- **Dual Encoder – atomic and composite (DE-AC):** Because the concepts learned through dense labeling capture only three basic local trends, we hypothesize that they may be insufficient to represent more complex temporal dynamics. To address this, we extend the Dual Encoder to incorporate composite concepts, i.e, higher-order compositions of simpler patterns, such as *peak* (rise $\rightarrow$ fall), *valley* (fall $\rightarrow$ rise), *negative plateau* (fall $\rightarrow$ plateau $\rightarrow$ rise), and *positive plateau* (rise $\rightarrow$ plateau $\rightarrow$ fall).

3.4 Post-hoc Concept-based Explanations

We perform post-hoc interpretability analysis to automatically assign concept labels (atomic and composite) to specific channels or neurons, thereby revealing the semantic structure of the model's internal representations. Since CRP supports both local and global interpretability, we use it to identify the most relevant parts of the input signal for different components of the network.

Relevance scores are first computed across layers using CRP to determine the most influential channels in convolutional layers and neurons in fully connected layers. To interpret their functional roles, we apply Relevance Maximization, which retrieves representative input patterns that maximize each unit's relevance and thus expose what it has learned. For convolutional layers, we further localize the receptive field of the most relevant neuron within each channel to highlight the specific segment of the input contributing most to the model's decision.

These localized input regions are then analyzed using the U-Time model to identify the atomic and composite concepts present in them. Prior to inference, signals are smoothed with a moving-average filter, and only atomic concepts covering at least 20% of the relevant region are retained, with the threshold chosen empirically. Contiguous atomic segments are merged to form composite concepts, e.g., peaks. This process maps the most relevant regions back to human-understandable temporal structures.

4 Experimental Setup

We conducted three sets of experiments to evaluate our framework: (1) a comparison of concept annotation methods on a controlled synthetic dataset; (2) an evaluation of classification performance on univariate time series, and (3) a post-hoc analysis of concept-based explanations on heartbeat classification.

4.1 Automatic Concept Annotation

Dataset. We generated synthetic signals with varying slopes and noise levels across different random seeds, resulting in a dataset that comprises 22,464 training, 1,728 validation, and 3,888 test samples. Details on the data split are provided in Appendix A.1.

Training. To ensure a fair comparison across methods, we performed a hyperparameter grid search using the validation set for both change-point detection and rule-based approaches. The best configuration for each method was selected based on the mean Intersection over Union (IoU), computed per signal to give equal weight to all samples. This metric reflects both the quality of segmentation and its alignment with the concept labels. For the U-Time model, we conducted ablation studies to determine the optimal configuration. Full details on hyperparameters, grid search, and ablation design are provided in Appendix B.1.

4.2 Univariate Time Series Classification

Dataset. We perform our univariate time series experiments on the UCR time series archive (Dau et al. 2019). Similarly to the experimental setup proposed by Middlehurst et al. (2021), we excluded datasets with unequal-length time series or missing values to avoid biasing the results, since the attributed concepts to time steps with missing values would

not be accurate. We also discard the *Fungi* dataset, as it contains only one training instance per class. After filtering, 112 out of the original 128 datasets remain.

Training. All models included a H-InceptionTime backbone as the primary time series encoder, inspired by the *Bake Off Redux* study (Middlehurst, Schäfer, and Bagnall 2024) and the `aeon` package (Middlehurst et al. 2024). We used a single classifier per experiment, and training was conducted using the predefined UCR train/test splits provided in the `aeon` package with five seeds to ensure robustness.

For concept-based models, the U-Time encoder was frozen. Signals shorter than 500 samples were interpolated to match U-Time's input size (1000). Longer sequences were either processed through non-overlapping sliding-window inference or upsampled using an empirically derived variability ratio based on the synthetic dataset.

Comprehensive details on hyperparameters, interpolation factor estimation, and the identification of composite concepts are provided in Appendix B.2.

4.3 Post-hoc Concept-based Explanations

Dataset. We used the PhysioNet MIT-BIH Arrhythmia dataset (Moody and Mark 2001), comprising Electrocardiogram (ECG) recordings from 47 subjects. Signals were aligned to the R–R peak and padded to a fixed length following Kachuee, Fazeli, and Sarrafzadeh (2018).

Training. We trained a 1D CNN classifier following the configuration of Achtibat et al. (2023), achieving a test accuracy of 92% (more details in Appendix B.3).

Concept-based Explainability. After training, we applied the CRP with the `EpsilonPlusFlat` composite to compute relevance scores conditioned on the PVC (Premature Ventricular Contraction) class, using a sample correctly classified by the model. The top five most relevant channels were selected for further analysis. For concept identification, the U-Time model was used to detect atomic and composite concepts within the relevant signal regions. Before inference, the signals were smoothed with a moving average filter (window size of 12) and interpolated to 1000 time steps.

5 Results

5.1 Automatic Concept Annotation

Figure 3 shows the performance of the different concept annotation methods, grouped by three distinct test conditions: *Noise* × *Slope*, *Seed*, and *Both*. The *Noise* × *Slope* condition involves combinations of noise and slope levels that were unseen during training, although each individual component (noise or slope) appeared in other combinations. Thus, this condition evaluates generalization to new compositions of familiar patterns rather than entirely new signal structures. In contrast, the *Seed* condition introduces completely unseen signal shapes, better reflecting distribution shifts expected in real-world datasets like the UCR. The *Both* condition combines unseen seeds and unseen noise-slope combinations, representing the most challenging scenario.

In the *Both* condition, U-Time achieved the highest median IoU (89.3%), followed by TSAX (76.4%) and SSTS (74.5%), while change-point methods ranged between 62.2% and 63.2%. Similar performance patterns were observed under the *Noise* × *Slope* and *Seed* conditions.

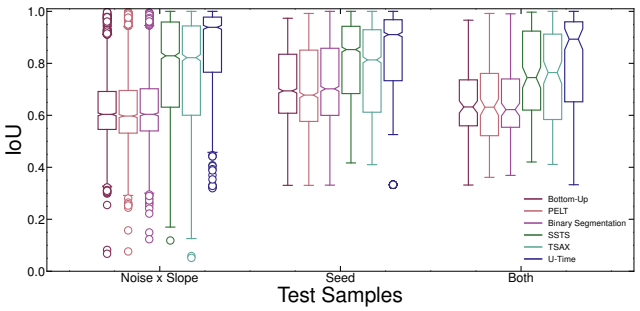

Figure 3: Our adapted U-Time consistently outperforms other baselines. Performance comparison of segmentation methods using notched box plots of IoU (%) across test signals. We evaluate generalization under three conditions: unseen seeds, unseen noise-slope combinations, and both.

All methods exhibited the largest performance spread in the *Noise* × *Slope* setting. Change-point approaches and SSTS generalized better to unseen random seeds than to new noise–slope combinations, whereas U-Time, having learned signal-shape representations, generalized more effectively to novel *Noise* × *Slope* pairs. Classical methods, which rely on fixed thresholds or heuristics rather than learned representations, were particularly sensitive to such variations.

Overall, U-Time and rule-based methods significantly outperformed all change-point methods, which did not differ significantly from one another. Nevertheless, the sharper performance decline of rule-based methods in the *Both* condition suggests limited generalization and higher sensitivity to parameter tuning. In contrast, U-Time maintained robust and consistent performance across all evaluation conditions.

5.2 Univariate Time Series Classification

Figure 4 shows the performance of the three models described in Section 3.3 across the UCR archive, compared to the baseline model. Most datasets cluster near the diagonal, indicating that the concept-based models achieve performance comparable to the baseline on average.

When compared with state-of-the-art (SOTA) interpretable methods, such as dictionary- and shapelet-based classifiers (Figure 5), our models achieve statistically comparable performance to the Random Dilated Shapelet Transform (RDST) and WEASEL V2, which represent the current best-performing shapelet- and dictionary-based methods according to Middlehurst, Schäfer, and Bagnall (2024).

However, when averaging performance across five seeds (Figure 6), the difference becomes statistically significant: our proposed models perform slightly worse than RDST and WEASEL V2, yet remain significantly better than WEASEL V1 and the RSF. This suggests that our models present higher variability, possibly due to sensitivity to weight initialization and reliance on training-loss-based model selection. Potential solutions include ensembling multiple runs, as done in Middlehurst, Schäfer, and Bagnall (2024).

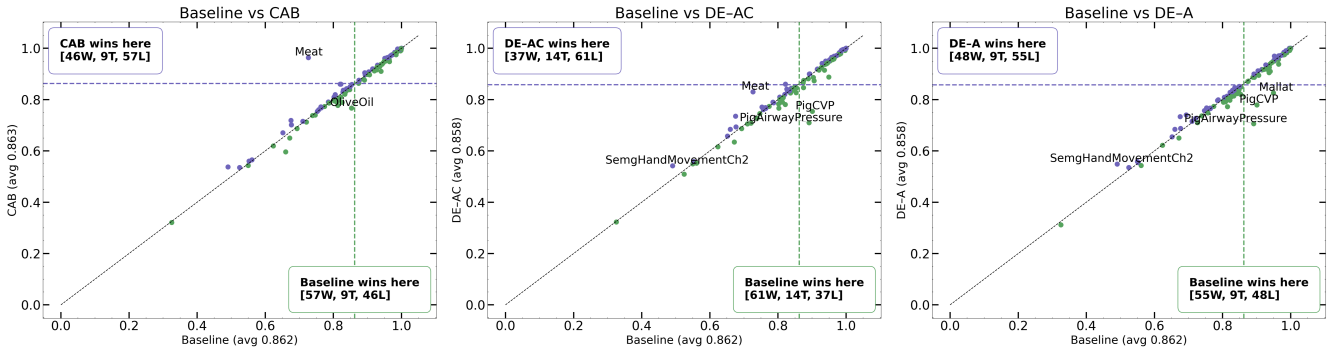

Figure 4: Classification performance across the UCR archive for each model pair: (Left) Baseline vs. CAB, (Middle) Baseline vs. DE-AC, (Right) Baseline vs. DE-A.

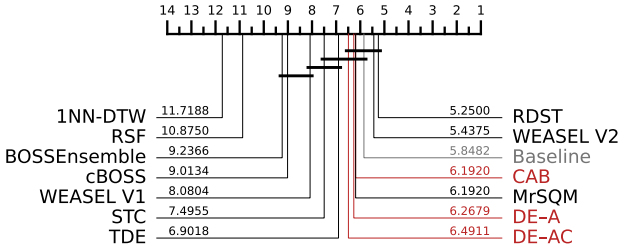

Figure 5: Critical difference plot showing the statistical significance of rankings achieved by the models in comparison to interpretable SOTA methods, such as dictionary- and shapelet-based, with the *original split* of the UCR archive. Reference results for the dictionary- and shapelet-based methods were provided by the `aeon` package (v1.1.0).

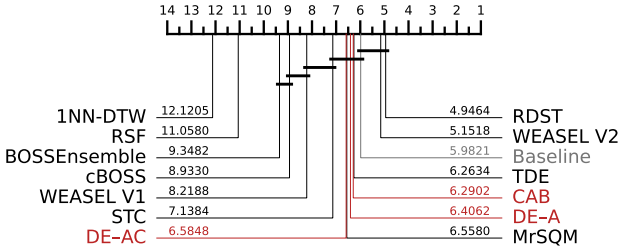

Figure 6: Critical difference plot showing the statistical significance of rankings achieved by the models in comparison to interpretable SOTA methods, such as dictionary- and shapelet-based across the *first five splits* of the UCR archive provided by Middlehurst, Schäfer, and Bagnall (2024).

5.3 Post-hoc Concept-based Explanation

As illustrated in Figure 7, combining U-Time with CRP enables automatic labeling of what each channel is learning within the model. In the last convolutional layer, channel 23 exhibits a polysemantic nature, encoding both peaks and plateaus, with three out of five reference samples highlighting the plateau, which may correspond to the compensatory pause characteristic of PVC. Channel 16, in contrast, consistently encodes a valley followed by a peak.

Tracing relevance back to the first convolutional layer clarifies how low-level features contribute to these high-level representations. For instance, channel 22 encodes peaks, rises followed by plateaus, and falls, whereas channel 56 mainly encodes peaks. This analysis shows how complex patterns emerge from combinations of simpler features.

From a practical perspective, these post-hoc concept labels help clinicians and domain experts interpret which signal characteristics influence the model's predictions. Mapping channels and neurons to interpretable temporal patterns allows assessment of whether the model relies on medically relevant or potentially spurious features. This insight opens avenues for building taxonomies connecting domain-agnostic concepts to domain-specific ones, guiding model pruning or refinement, and identifying pure circuits (i.e. paths in the network corresponding to unique patterns). Overall, this might contribute to enhancing model validation, trust, and the design of decision support systems that provide concept-level explanations alongside predictions.

6 Discussion

This work revisited the role of symbolic and semantic representations as a foundation for interpretability in time series analysis. Within the taxonomy proposed by Theissler et al. (2022), our framework primarily belongs to the *subsequence-based* family of methods, as it grounds explanations in short, interpretable temporal patterns such as rising, falling, and plateau trends.

Although our proposed models did not exceed the accuracy of SOTA classifiers, they achieved performance that remained remarkably close, showing that concept-level supervision and interpretability at the input level can be introduced without a prohibitive cost in performance. In addition, the CAB model makes it easy to see how each concept affects the prediction, since the concept probabilities go through a single linear layer. For the other two dual-encoder

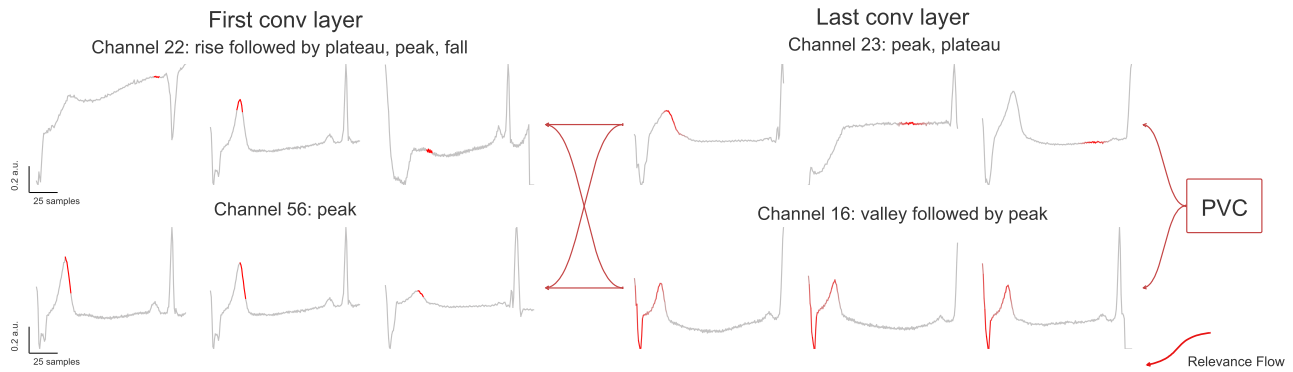

Figure 7: Post-hoc concept-based explanation bridging CRP and semantic concepts in time series classification. For a correctly classified Premature Ventricular Contraction (PVC) sample, the most relevant channels in the last convolutional layer are identified using CRP. We then trace the relevance from these channels back to the first convolutional layer to identify which low-level channels contribute most to their activations. The channel labels were automatically generated by our dense-labeling model and corresponding rules, indicating the concepts each channel encodes.

models, concept contributions can still be analyzed using attribution methods, for example. We conjecture that this semantic representation offers greater interpretability than dictionary-based methods, whose symbolic encodings often lack intrinsic semantic meaning.

The dense-labeling model was trained and validated on synthetic data; we wondered how it generalizes to more complex datasets such as the UCR archive, which encompasses a wide variety of signal types and behaviors. We conducted a qualitative inspection of predicted concepts across the UCR archive (see Supplementary Material[2]) and found that overall, the model correctly identified meaningful temporal structures such as rising, falling, and plateau segments. Nonetheless, the model occasionally misclassifies small fluctuations as rises or falls, particularly in highly interpolated or noisy signals, while rapid changes in low-interpolated signals may go undetected or be misclassified. While this sensitivity may be mitigated through optimized smoothing, interpolation remains beneficial for capturing fine-grained temporal features. Future work could explore dataset-specific kernel optimization to improve robustness.

We initially expected that introducing more complex concepts, such as *peaks* and *valleys*, would enhance performance beyond the three atomic concepts. Surprisingly, this was not the case. The addition of composite concepts did not yield consistent improvements and, in some cases, even led to minor performance drops. This counterintuitive outcome likely arises from the increased model complexity introduced by additional concept channels, which can destabilize optimization in smaller datasets. Moreover, noisy signals containing multiple small peaks may cause redundant or spurious activations, ultimately offsetting the benefits of richer compositional representations. Overall, these results suggest that while composite concepts enrich semantic representation, their effective use may require larger datasets or refined regularization strategies to stabilize learning, and further investigation into more robust strategies for combining concept information with raw signal representations.

[2]The supplementary material is available at: https://doi.org/10.5281/zenodo.17833105.

7 Limitations

Despite these promising findings, several limitations remain. The current concept space, although extended with a few composite patterns, still represents only a shallow hierarchy of temporal semantics. This limited coverage restricts the model's ability to capture finer or more domain-specific patterns. Training the dense-labeling model to detect additional primitives (e.g., slope, noise) and involving domain experts to disassemble complex concepts are potential directions for expanding this space. Furthermore, the dense-labeling model can be sensitive to minor fluctuations, occasionally producing noisy concept boundaries. Performance variability, especially in low sample sizes or imbalanced datasets, also indicates that training stability could be improved. Future work will focus on expanding the concept hierarchy toward more structured and domain-relevant representations, while enhancing the robustness of both annotation and classification models.

8 Conclusion

This work introduced a concept-based framework for interpretable time series classification that bridges symbolic and neural representations. By training a dense-labeling model on synthetic signals, we showed that high-level, domain-agnostic concepts such as rising, falling, and plateau can be effectively learned and transferred to diverse datasets in the UCR archive. These concept signals serve two complementary purposes: they act as auxiliary supervision to guide model training, and as post-hoc labels to explain the behavior of trained networks. Across experiments, concept-aware models achieved competitive performance while offering transparent, human-aligned reasoning over temporal data. Overall, our results demonstrate that integrating concept-level structure into deep learning offers a promising path toward interpretable time series models.

Acknowledgements

This work was supported by European funds through the Recovery and Resilience Plan, project "Center for Responsible AI", project number C645008882-00000055. We thank Mariana Pinto for assistance with figure illustrations. We thank all contributors to the UCR Time Series Classification Archive for making their datasets publicly available.

References

Achtibat, R.; Dreyer, M.; Eisenbraun, I.; Bosse, S.; Wiegand, T.; Samek, W.; and Lapuschkin, S. 2023. From attribution maps to human-understandable explanations through Concept Relevance Propagation. *Nature Machine Intelligence*, 5(9): 1006–1019.

Armstrong, S. L.; Gleitman, L. R.; and Gleitman, H. 1983. What some concepts might not be. *Cognition*, 13(3): 263–308.

Bai, J. 1997. Estimating Multiple Breaks One at a Time. *Econometric theory*, 13(3): 315–352.

Bordt, S.; Finck, M.; Raidl, E.; and von Luxburg, U. 2022. Post-Hoc Explanations Fail to Achieve their Purpose in Adversarial Contexts. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, 891–905. New York, NY, USA: Association for Computing Machinery. ISBN 9781450393522.

Bostrom, A.; and Bagnall, A. 2017. Binary Shapelet Transform for Multiclass Time Series Classification. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems XXXII: Special Issue on Big Data Analytics and Knowledge Discovery*, 24–46. Springer.

Dau, H. A.; Bagnall, A.; Kamgar, K.; Yeh, C.-C. M.; Zhu, Y.; Gharghabi, S.; Ratanamahatana, C. A.; and Keogh, E. 2019. The UCR Time Series Archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6): 1293–1305.

Dempster, A.; Schmidt, D. F.; and Webb, G. I. 2023. HYDRA: competing convolutional kernels for fast and accurate time series classification. *Data Mining and Knowledge Discovery*, 37(5): 1779–1805.

Dempster, A.; Schmidt, D. F.; and Webb, G. I. 2024. QUANT: a minimalist interval method for time series classification. *Data Mining and Knowledge Discovery*, 38(4): 2377–2402.

Giese, D. A.; Bourne, J. R.; and Ward, J. W. 1979. Syntactic Analysis of the Electroencephalogram. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(8): 429–435.

Guillaume, A.; Vrain, C.; and Elloumi, W. 2022. Random Dilated Shapelet Transform: A New Approach for Time Series Shapelets. In *International Conference on Pattern Recognition and Artificial Intelligence*, 653–664. Springer.

Horowitz, S. L. 1975. A Syntactic Algorithm for Peak Detection in Waveforms with Applications to Cardiography. *Commun. ACM*, 18(5): 281–285.

Ismail-Fawaz, A.; Devanne, M.; Weber, J.; and Forestier, G. 2022. Deep Learning For Time Series Classification Using New Hand-Crafted Convolution Filters. In *2022 IEEE International Conference on Big Data (Big Data)*, 972–981.

Kachuee, M.; Fazeli, S.; and Sarrafzadeh, M. 2018. ECG Heartbeat Classification: A Deep Transferable Representation. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*, 443–444.

Karlsson, I.; Papapetrou, P.; and Boström, H. 2016. Generalized random shapelet forests. *Data mining and knowledge discovery*, 30(5): 1053–1085.

Keogh, E. J.; Chu, S.; Hart, D.; and Pazzani, M. J. 2001. An Online Algorithm for Segmenting Time Series. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, ICDM '01, 289–296. USA: IEEE Computer Society. ISBN 0769511198.

Killick, R.; Fearnhead, P.; and Eckley, I. 2012. Optimal Detection of Changepoints With a Linear Computational Cost. *Journal of the American Statistical Association*, 107: 1590–1598.

Koh, P. W.; Nguyen, T.; Tang, Y. S.; Mussmann, S.; Pierson, E.; Kim, B.; and Liang, P. 2020. Concept Bottleneck Models. In III, H. D.; and Singh, A., eds., *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 5338–5348. PMLR.

Le Nguyen, T.; Gsponer, S.; Ilie, I.; O'Reilly, M.; and Ifrim, G. 2019. Interpretable time series classification using linear models and multi-resolution multi-domain symbolic representations. *Data Mining and Knowledge Discovery*, 33(4): 1183–1222.

Lin, J.; Keogh, E.; Lonardi, S.; and Chiu, B. 2003. A Symbolic Representation of Time Series, with Implications for Streaming Algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, DMKD '03, 2–11. New York, NY, USA: Association for Computing Machinery. ISBN 9781450374224.

Lin, J.; Keogh, E.; Wei, L.; and Lonardi, S. 2007. Experiencing SAX: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2): 107–144.

Lipton, Z. C. 2018. The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3): 31–57.

Lucas, B.; Shifaz, A.; Pelletier, C.; O'neill, L.; Zaidi, N.; Goethals, B.; Petitjean, F.; and Webb, G. I. 2019. Proximity Forest: an effective and scalable distance-based classifier for time series. *Data Mining and Knowledge Discovery*, 33(3): 607–635.

Michel, P.; Levy, O.; and Neubig, G. 2019. Are Sixteen Heads Really Better than One? In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc.

Middlehurst, M.; and Bagnall, A. 2022. The FreshPRINCE: A Simple Transformation Based Pipeline Time Series Classifier. In *International Conference on Pattern Recognition and Artificial Intelligence*, 150–161. Springer.

Middlehurst, M.; Ismail-Fawaz, A.; Guillaume, A.; Holder, C.; Guijo-Rubio, D.; Bulatova, G.; Tsaprounis, L.; Mentel, L.; Walter, M.; Schäfer, P.; et al. 2024. aeon: a Python toolkit

for learning from time series. *Journal of Machine Learning Research*, 25(289): 1–10.

Middlehurst, M.; Large, J.; Cawley, G.; and Bagnall, A. 2020. The Temporal Dictionary Ensemble (TDE) Classifier for Time Series Classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 660–676. Springer.

Middlehurst, M.; Large, J.; Flynn, M.; Lines, J.; Bostrom, A.; and Bagnall, A. 2021. HIVE-COTE 2.0: a new meta ensemble for time series classification. *Mach. Learn.*, 110(11–12): 3211–3243.

Middlehurst, M.; Schäfer, P.; and Bagnall, A. 2024. Bake off redux: a review and experimental evaluation of recent time series classification algorithms. *Data Mining and Knowledge Discovery*, 38(4): 1958–2031.

Middlehurst, M.; Vickers, W.; and Bagnall, A. 2019. Scalable Dictionary Classifiers for Time Series Classification. In *International conference on intelligent data engineering and automated learning*, 11–19. Springer.

Miot, A.; and Drigout, G. 2020. An Empirical Study of Neural Networks for Trend Detection in Time Series. *SN Computer Science*, 1(6): 347.

Moody, G. B.; and Mark, R. G. 2001. The Impact of the MIT-BIH Arrhythmia Database. *IEEE engineering in medicine and biology magazine*, 20(3): 45–50.

Murphy, G. L. 2002. *The Big Book of Concepts*. The MIT Press. ISBN 978-0-262-28035-8.

Nguyen, T. L.; and Ifrim, G. 2022. Fast Time Series Classification with Random Symbolic Subsequences. In *International Workshop on Advanced Analytics and Learning on Temporal Data*, 50–65. Springer.

Opsahl, T. A.; and Antun, V. 2024. Achieving Data Efficient Neural Networks with Hybrid Concept-based Models. *arXiv preprint arXiv:2408.07438*.

Perslev, M.; Jensen, M.; Darkner, S.; Jennum, P. J.; and Igel, C. 2019. U-Time: A Fully Convolutional Network for Time Series Segmentation Applied to Sleep Staging. *Advances in neural information processing systems*, 32.

Rodrigues, J.; Folgado, D.; Belo, D.; and Gamboa, H. 2019. SSTS: A syntactic tool for pattern search on time series. *Information Processing & Management*, 56: 2019.

Schäfer, P.; and Leser, U. 2023. WEASEL 2.0: a random dilated dictionary transform for fast, accurate and memory constrained time series classification. *Machine Learning*, 112(12): 4763–4788.

Schäfer, P. 2015. The BOSS is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6): 1505–1530.

Schäfer, P.; and Högqvist, M. 2012. SFA: A Symbolic Fourier Approximation and Index for Similarity Search in High Dimensional Datasets. In *Proceedings of the 15th International Conference on Extending Database Technology*, 516–527. Berlin Germany: ACM. ISBN 978-1-4503-0790-1.

Schäfer, P.; and Leser, U. 2017. Fast and Accurate Time Series Classification with WEASEL. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 637–646. Singapore Singapore: ACM. ISBN 978-1-4503-4918-5.

Tenenbaum, J. B. 1999. *A Bayesian Framework for Concept Learning*. Ph.D. thesis, Massachusetts Institute of Technology.

Theissler, A.; Spinnato, F.; Schlegel, U.; and Guidotti, R. 2022. Explainable AI for Time Series Classification: A Review, Taxonomy and Research Directions. *Ieee Access*, 10: 100700–100724.

Trahanias, P.; and Skordalakis, E. 1990. Syntactic Pattern Recognition of the ECG. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7): 648–657.

Truong, C.; Oudre, L.; and Vayatis, N. 2020. Selective review of offline change point detection methods. *Signal Processing*, 167: 107299.

Udupa, J. K.; and Murthy, I. S. N. 1980. Syntactic Approach to ECG Rhythm Analysis. *IEEE Transactions on Biomedical Engineering*, BME-27(7): 370–375.

van Sprang, A.; Acar, E.; and Zuidema, W. 2024. Enforcing Interpretability in Time Series Transformers: A Concept Bottleneck Framework. arXiv:2410.06070.

Vielhaben, J.; Lapuschkin, S.; Montavon, G.; and Samek, W. 2024. Explainable AI for time series via Virtual Inspection Layers. *Pattern Recognition*, 150: 110309.

Voita, E.; Talbot, D.; Moiseev, F.; Sennrich, R.; and Titov, I. 2019. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. In Korhonen, A.; Traum, D.; and Màrquez, L., eds., *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 5797–5808. Florence, Italy: Association for Computational Linguistics.

Ye, L.; and Keogh, E. J. 2009. Time Series Shapelets: A New Primitive for Data Mining. In *Knowledge Discovery and Data Mining*.

Zhang, K.; Li, Y.; Chai, Y.; and Huang, L. 2018. Trend-based Symbolic Aggregate Approximation for Time Series Representation. In *2018 Chinese Control And Decision Conference (CCDC)*, 2234–2240. IEEE.

A Synthetic Dataset

Semi-automatic concept labeling pipeline. The semi-automatic labeling procedure is summarized in Figure 8. It consists of two main stages: (1) reference set construction and feature extraction, and (2) automatic segmentation.

First, we generate synthetic time series using different random seeds and parameter combinations (see Equation 1). For each time series, we apply trend labeling under multiple parameter settings and manually select the configuration that produces the most coherent segment-level trends. We then extract features from each labeled segment (e.g., variance of the first derivative and slope), standardize them, and cluster with K-Means to identify representative trend clusters.

$$\forall i \in \{1, \ldots, N\},\ \forall t \in]t_i, t_{i+1}], \quad Y_t = Y_{t_i} + \beta_i (t - t_i) + \epsilon_i, \tag{1}$$

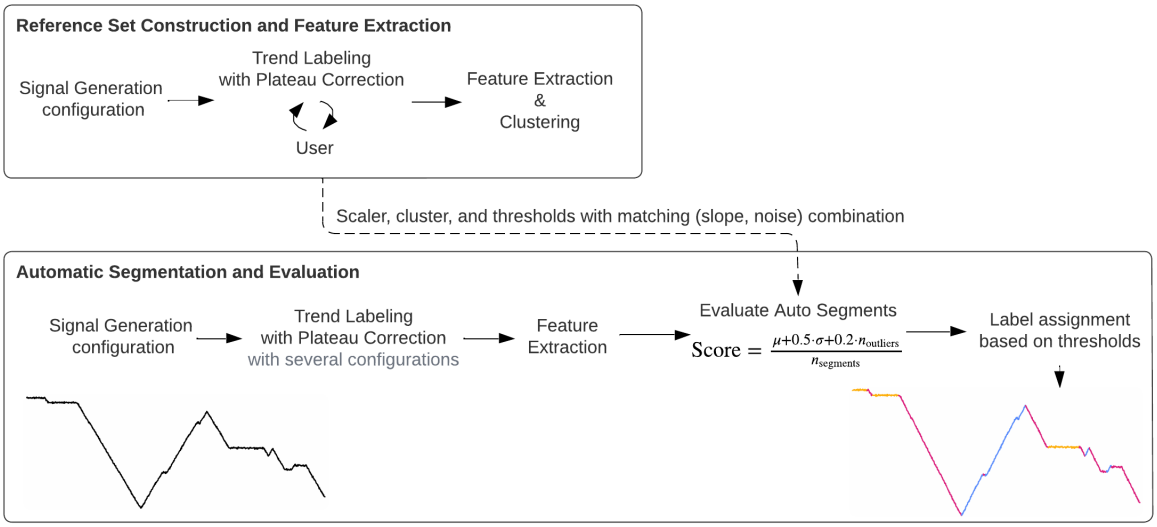

Figure 8: Overview of the semi-automatic labeling pipeline. We generate a rich representative dataset with raw time series and ground-truth concept annotations.

where:

- N is the number of segments (user configurable).
- $\beta_i \in \left\{-\frac{\beta_{\max}}{n}, \ldots, -\frac{\beta_{\max}}{1}, 0, \frac{\beta_{\max}}{1}, \ldots, \frac{\beta_{\max}}{n}\right\}$ is the slope for segment i, sampled from a discrete set. Here, $\beta_{\max} > 0$ is the maximum slope, and $n \in \mathbb{N}^*$ controls the granularity (both user configurable).
- $\epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$ is an i.i.d. Gaussian noise term for segment i, with standard deviation $\sigma_i > 0$. Where σ_i^2 is sampled from a uniform distribution with minimum and maximum values configurable by the user.
- t_i and t_{i+1} are the start and end time indices of segment i.

By varying the ratio between slope and noise, we control the ambiguity of visual patterns. High slopes and low noise yield visually unambiguous trends (e.g., clearly rising or falling segments), while small slopes combined with higher noise produce visually ambiguous or borderline-plateau segments. This tunability allows us to systematically assess how models handle ambiguous regions.

This reference set plays a key role in reducing the need for manual labeling. Rather than annotating each new time series from scratch, we evaluate the coherence of automatic segmentation results by comparing them to a small set of high-quality labeled examples. This enables automatic parameter selection while maintaining consistency and interpretability across signals.

For a new time series, we evaluate multiple labeling configurations. For each configuration, we extract the same features and compute the Euclidean distance to the reference cluster centroids. Each configuration is scored based on its average distance, variability, and the number of outlier segments (those above the 90th percentile). The configuration with the lowest score is selected, and a final label assignment is performed using slope-based thresholds. A post-processing step then ensures temporal consistency by correcting short segments.

A.1 Dataset split

To assess generalization, we reserved two specific seeds and slope-noise configurations that differ from, but are similar to, those used during training. This ensures the model is evaluated on previously unseen signal patterns. The validation set includes two additional seeds drawn from the same parameter range as the training set.

B Experimental Setup

B.1 Concept Annotation Methods

To evaluate concept annotation quality, we used the IoU, computed per signal to ensure equal weighting across samples (see Figure 9). This metric captures both segmentation accuracy and the temporal alignment between predicted and ground-truth concept labels.

Hyperparameter search space. In Table 1, we present the hyperparameter values used in the grid search to find the optimal configuration using the validation set for both the change-point detection and rule-based approaches.

U-Time. The model was trained using ground-truth labels from our synthetic dataset with cross-entropy loss and an Adam optimizer with a learning rate of 1×10^{-4} and weight decay of 1×10^{-4}. A learning rate scheduler reduced the rate by a factor of 0.5 after two epochs without improvement in validation loss. Early stopping was applied with a patience of 10 epochs. Models were trained for up to 75 epochs with a batch size of 64.

We performed an ablation study to identify the parameters that most affect the model's generalization, covering the three distinct conditions mentioned in the main paper.

In Table 2, we observed that reducing the model depth from 4 to 3 improved performance, suggesting that deeper

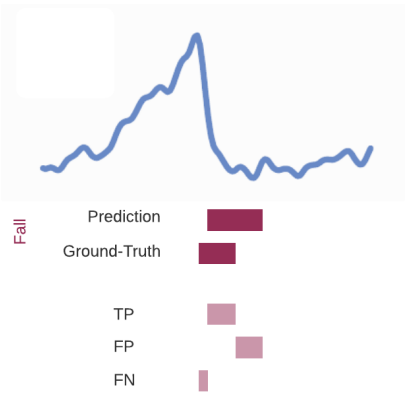

Figure 9: Illustration of the IoU concept used to evaluate segmentation quality. The plot shows a sample signal with predicted and ground-truth fall segments, along with the corresponding true positives (TP), false positives (FP), and false negatives (FN). The IoU metric quantifies the overlap between prediction and ground truth.

models are prone to overfitting in this task. Kernel size effects depended on depth: with depth 4, larger kernels performed better, likely compensating for the loss of fine-grained temporal details in deeper models by expanding receptive fields. Shallower models (depth 3), preserving more local information, performed best with smaller kernels. Adding dropout (0.2) improved generalization, particularly on *Seed* and *Both* splits where unseen shapes challenge the model.

Performance generally decreased when moving from *Noise* × *Slope* to *Both*, as the latter combines both unseen signal shapes and unseen slope-noise patterns. Higher scores on *Noise* × *Slope* suggest the model may memorize shapes and simply adapt to new *Noise* × *Slope* pairs. The pronounced drop in the *Seed* and *Both* settings highlights the importance of evaluating on truly novel shapes. Dropout and reduced depth lessened this drop, likely by preventing overfitting and memorization, leading to improved robustness.

B.2 Classification

Training setup. All models were implemented in PyTorch (v2.5.1). Weights of convolutional and linear layers were initialized using Xavier uniform initialization, with biases set to zero. For batch normalization layers, weights were set to 1 and biases to 0.

The Adam optimizer was used with a learning rate of 1×10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1 \times 10^{-7}$. To adjust the learning rate during training, we employed a `ReduceLROnPlateau` scheduler, reducing the learning rate by a factor of 0.5 if the training loss does not improve for 50 epochs, with a minimum learning rate of 1×10^{-4}. The model was saved based on the best training loss.

Variability-based interpolation. Initially, as U-Time expects signals of length 1000, shorter signals were interpolated to this length, while longer signals were processed using a sliding window inference with non-overlapping windows. The final partial window was included to ensure full

Algorithm	Parameter	Search Range
Change-point Detection	*model type*	{L1, L2}
	minimum segment size	{1, 3, 5}
	jump	{2, 3, 4}
	penalty	{0.001, 0.01, 0.1}
TSAX	*segment splits and sub splits*	{[20, 5], [20, 4], [20, 2], [25, 5], [25, 4],[50, 5], [50, 4]}
All	*threshold*	{0.0001, 0.001, 0.01}
All (Preprocessing)	*kernel lenght moving window*	{5, 9, 15, 19, 25}* {hanning}
All (Postprocessing)	*enable*	{False, True}
	minimum segment sizes	{5, 10, 15}

Table 1: Hyperparameter search space for the concept annotation methods. Change-point detection was performed with the `ruptures` library (Truong, Oudre, and Vayatis 2020). SSTS used the derivative connotation. (*) Kernel length 30 was considered only for the rule-based approach.

temporal coverage. However, since the UCR signals can be sparsely sampled (e.g., upward, downward, or plateau segments spanning few time steps) compared to our densely sampled synthetic signals, some longer signals resulted in segments with inaccurate concept identification. Therefore, we determined the interpolation factor using the synthetic test set as an independent reference for estimating typical local signal variability.

Local signal variability was estimated by smoothing each normalized signal with a moving average of length equal to 5 and computing the first-order derivative. The standard deviation of the derivative was measured within sliding windows of the same length, and the median of these local standard deviations was taken as a robust variability measure. A 95th percentile value of median derivative variability was then established across the reference dataset and the UCR datasets (training sets) for comparison. Signals shorter or equal to 500 samples or with low variability were directly upsampled to the model input size (1000) if needed, while signals with both high variability (ratio > 2.5) and sufficient length (> 500 samples) were linearly upsampled proportionally to their variability. The 2.5 threshold was empirically chosen based on the visualization of the concept identification.

Composite concept detection. For composite concepts, atomic concept predictions were first segmented into contiguous sequences of the same predicted label. *Peaks*, *valleys*, and *directional plateaus* were detected by pattern matching on these segments (e.g., *peak* (rise → fall), *valley* (fall → rise), *negative plateau* (fall → plateau → rise), and *positive plateau* (rise → plateau → fall)). The probabil-

Dropout (0.2)	Depth	CF	KS	IoU (%) Seed	N×S	Both
×	4	2	5	71.2	96.6	65.6
×	4	2	3	73.0	**97.7**	64.8
×	4	1.5	5	69.9	<u>97.2</u>	64.7
×	3	2	5	79.8	96.5	73.7
×	3	2	3	<u>83.8</u>	95.8	<u>81.9</u>
×	3	1.5	3	80.0	93.0	74.3
✓	3	2	3	**91.1**	93.9	**89.3**

Table 2: Ablation study of U-Time for the concept annotation task. Median IoU (%) is reported across different input conditions: unseen seeds, unseen noise-slope combinations (N×S), and samples with both. We vary depth, kernel size (KS), complexity factor (CF), and the use of dropout. Best results in bold, second-best underlined.

ity of each composite concept was computed as the product of the mean probabilities of its constituent atomic concepts within the segment, approximating the joint likelihood of sequential atomic events and ensuring that low-confidence atomic detections proportionally reduced composite confidence. Overlapping composite concepts were allowed when distinct sequences satisfied multiple compositional patterns.

B.3 Post-hoc Concept-based Explanation

CNN architecture & training. Preprocessing included normalization by subtracting the mean value (0.2115) and dividing by the standard deviation (0.2462). The classifier consists of three feature extraction blocks. Each block contains a 1D convolutional layer (64 channels, kernel size 7), followed by ReLU activation and 1D max-pooling (kernel size 3, stride 2). After the feature extractor, a 1D average-pooling layer (kernel size 7) and flatten operation feed two fully connected layers (64 and 32 neurons, each followed by ReLU). The final linear layer has five neurons corresponding to the target classes. The model was trained the model for 25 epochs using the RMSprop optimizer (learning rate 1×10^{-4}, weight decay 1×10^{-6}) and cross-entropy loss with class balancing to address dataset imbalance.

C Classification

Comparison with SOTA methods. For comparison with SOTA approaches in Figure 5 and 6, we adopted the same methods included in the *Bake Off Redux* study (Middlehurst, Schäfer, and Bagnall 2024), namely the Shapelet Transform Classifier (STC) (Bostrom and Bagnall 2017), Generalised Random Shapelet Forest (RSF) (Karlsson, Papapetrou, and Boström 2016), MrSQM (Nguyen and Ifrim 2022), Random Dilated Shapelet Transform (RDST) (Guillaume, Vrain, and Elloumi 2022), BOSS (Ensemble-BOSS) (Schäfer 2015), WEASEL v1.0 (Schäfer and Leser 2017), WEASEL v2.0 (Schäfer and Leser 2023), Contractable BOSS (cBOSS) (Middlehurst, Vickers, and Bagnall 2019), and Temporal Dictionary Ensemble (TDE) (Middlehurst et al. 2020). Reference results for all these methods were retrieved with the `aeon` package (v1.1.0).

In this section, we also cover the best classifier for each of the eight categories defined by Middlehurst, Schäfer, and Bagnall (2024), highlighted in blue in Figure 10. These include models such as HIVE-COTE v2.0 (HC2) (Middlehurst et al. 2021), H-InceptionTime (H-IT) (Ismail-Fawaz et al. 2022), MultiROCKET-Hydra (MR-Hydra) (Dempster, Schmidt, and Webb 2023), Fresh Pipeline with Rotation Forest Classifier (FP) (Middlehurst and Bagnall 2022), Proximity Forest (PF) (Lucas et al. 2019), and QUANT (Dempster, Schmidt, and Webb 2024).

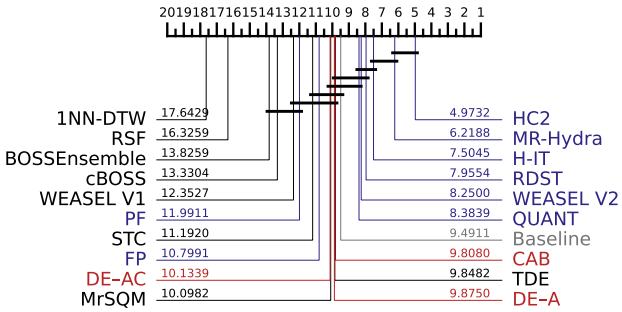

Figure 10: Critical difference plot showing the statistical significance of rankings achieved by the models in comparison to dictionary- and shapelet-based SOTA and top-performing methods in the respective category, across the *first five splits* of the UCR archive provided by Middlehurst, Schäfer, and Bagnall (2024).

As expected, the top-performing methods remain the best overall, except for FP and PF. In this case, our models reach competitive performance when compared to other interpretable approaches; however, non-interpretable models still achieve the highest accuracy. Our goal is not to surpass highly specialized SOTA architectures, but to demonstrate that our framework can achieve accuracy competitive with top-performing time series classifiers while offering a substantially richer and more structured form of interpretability. There is also room for further improvement: ensemble strategies could significantly boost results, as illustrated by the performance gap between our baseline and H-IT, likely influenced by different computational resources, but especially by the ensemble architecture.

Additional results. Analysis of Tables 3 and 4 shows that the proposed models generally perform on par with the baseline, while providing notable improvements on some datasets (e.g., *Beef*), suggesting that concept integration is beneficial in specific cases. However, certain datasets exhibit high variability (e.g., *Meat*, *Mallat*), which may be due to weight initialization and the small size of these datasets. A potential solution could be to integrate an ensemble approach.

Concept-based representations appear particularly helpful in datasets such as *ACSF1*, *Rock*, and *SmallKitchenAppliances*, where concept integration visibly helps distinguish between classes, even by inspection of the raw signals.

	Baseline	CAB	DE-AC	DE-A
ACSF1	0.82 ± 0.06	**0.86 ± 0.03**	**0.86 ± 0.05**	0.83 ± 0.06
Adiac	**0.84 ± 0.01**	0.83 ± 0.02	0.83 ± 0.01	0.83 ± 0.02
ArrowHead	**0.89 ± 0.03**	0.88 ± 0.02	0.88 ± 0.04	**0.89 ± 0.03**
BME	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Beef	0.69 ± 0.06	0.69 ± 0.07	0.69 ± 0.08	**0.74 ± 0.10**
BeetleFly	0.82 ± 0.13	**0.86 ± 0.10**	0.82 ± 0.10	0.81 ± 0.11
BirdChicken	0.94 ± 0.08	0.91 ± 0.08	**0.95 ± 0.05**	0.91 ± 0.09
CBF	0.99 ± 0.01	0.99 ± 0.01	**1.00 ± 0.00**	0.99 ± 0.00
Car	0.89 ± 0.04	**0.90 ± 0.03**	**0.90 ± 0.03**	0.89 ± 0.02
Chinatown	**0.96 ± 0.02**	**0.96 ± 0.02**	**0.96 ± 0.02**	**0.96 ± 0.02**
ChlorineConcentration	**0.88 ± 0.01**	0.87 ± 0.01	0.87 ± 0.01	0.87 ± 0.01
CinCECGTorso	**0.84 ± 0.04**	**0.84 ± 0.05**	**0.84 ± 0.04**	0.82 ± 0.05
Coffee	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Computers	**0.84 ± 0.03**	**0.84 ± 0.04**	**0.84 ± 0.03**	**0.84 ± 0.03**
CricketX	0.84 ± 0.02	0.83 ± 0.02	0.83 ± 0.00	**0.85 ± 0.02**
CricketY	**0.84 ± 0.01**	**0.84 ± 0.01**	**0.84 ± 0.01**	**0.84 ± 0.01**
CricketZ	0.85 ± 0.01	**0.86 ± 0.02**	0.85 ± 0.01	0.85 ± 0.01
Crop	**0.77 ± 0.00**	**0.77 ± 0.00**	**0.77 ± 0.00**	**0.77 ± 0.00**
DiatomSizeReduction	**0.94 ± 0.06**	0.93 ± 0.06	**0.94 ± 0.06**	0.93 ± 0.05
DistalPhalanxOutlineAgeGroup	0.75 ± 0.04	0.74 ± 0.04	0.75 ± 0.04	**0.76 ± 0.04**
DistalPhalanxOutlineCorrect	**0.81 ± 0.02**	0.79 ± 0.01	0.79 ± 0.02	0.79 ± 0.02
DistalPhalanxTW	0.65 ± 0.01	**0.67 ± 0.02**	0.66 ± 0.02	0.65 ± 0.01
ECG200	0.90 ± 0.02	**0.91 ± 0.01**	0.89 ± 0.02	**0.91 ± 0.01**
ECG5000	**0.94 ± 0.00**	**0.94 ± 0.00**	**0.94 ± 0.00**	**0.94 ± 0.01**
ECGFiveDays	**1.00 ± 0.00**	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.01
EOGHorizontalSignal	**0.85 ± 0.10**	0.84 ± 0.12	0.84 ± 0.12	0.84 ± 0.12
EOGVerticalSignal	**0.76 ± 0.14**	**0.76 ± 0.15**	**0.76 ± 0.14**	0.75 ± 0.16
Earthquakes	0.72 ± 0.02	0.71 ± 0.01	0.71 ± 0.03	**0.73 ± 0.02**
ElectricDevices	**0.85 ± 0.08**	**0.85 ± 0.08**	**0.85 ± 0.08**	**0.85 ± 0.08**
EthanolLevel	**0.81 ± 0.02**	0.80 ± 0.01	**0.81 ± 0.02**	**0.81 ± 0.03**
FaceAll	0.95 ± 0.08	**0.96 ± 0.07**	0.95 ± 0.08	0.95 ± 0.08
FaceFour	0.93 ± 0.06	0.93 ± 0.05	**0.94 ± 0.04**	0.93 ± 0.04
FacesUCR	**0.97 ± 0.01**	**0.97 ± 0.01**	**0.97 ± 0.01**	**0.97 ± 0.01**
FiftyWords	**0.82 ± 0.01**	0.81 ± 0.02	0.78 ± 0.02	0.80 ± 0.03
Fish	**0.98 ± 0.00**	**0.98 ± 0.01**	**0.98 ± 0.01**	0.97 ± 0.01
FordA	**0.95 ± 0.01**	**0.95 ± 0.00**	**0.95 ± 0.00**	**0.95 ± 0.01**
FordB	**0.92 ± 0.05**	**0.92 ± 0.04**	**0.92 ± 0.04**	**0.92 ± 0.04**
FreezerRegularTrain	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
FreezerSmallTrain	0.93 ± 0.07	0.92 ± 0.07	0.92 ± 0.06	**0.94 ± 0.05**
GunPoint	**1.00 ± 0.00**	0.99 ± 0.01	**1.00 ± 0.00**	**1.00 ± 0.00**
GunPointAgeSpan	**0.98 ± 0.01**	0.97 ± 0.02	**0.98 ± 0.02**	**0.98 ± 0.01**
GunPointMaleVersusFemale	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
GunPointOldVersusYoung	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Ham	0.80 ± 0.06	**0.81 ± 0.06**	0.78 ± 0.06	0.79 ± 0.08
HandOutlines	**0.95 ± 0.01**	**0.95 ± 0.01**	**0.95 ± 0.01**	**0.95 ± 0.01**
Haptics	0.55 ± 0.02	0.54 ± 0.01	0.55 ± 0.02	**0.56 ± 0.01**
Herring	**0.67 ± 0.04**	0.65 ± 0.06	0.63 ± 0.04	0.65 ± 0.07
HouseTwenty	**0.96 ± 0.02**	0.95 ± 0.03	**0.96 ± 0.01**	0.94 ± 0.04
InlineSkate	0.55 ± 0.03	**0.56 ± 0.03**	**0.56 ± 0.03**	0.55 ± 0.03
InsectEPGRegularTrain	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
InsectEPGSmallTrain	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
InsectWingbeatSound	**0.62 ± 0.01**	**0.62 ± 0.01**	**0.62 ± 0.01**	**0.62 ± 0.01**
ItalyPowerDemand	0.96 ± 0.01	**0.97 ± 0.01**	0.96 ± 0.01	**0.97 ± 0.00**
LargeKitchenAppliances	**0.94 ± 0.02**	**0.94 ± 0.03**	**0.94 ± 0.02**	**0.94 ± 0.02**
Lightning2	**0.81 ± 0.03**	0.78 ± 0.02	0.79 ± 0.02	**0.81 ± 0.04**
Lightning7	0.81 ± 0.04	**0.82 ± 0.03**	0.80 ± 0.04	0.81 ± 0.04

Table 3: Accuracy results across 112 datasets from the UCR archive. Values are reported as mean ± standard deviation across the 5 seeds.

	Baseline	CAB	DE-AC	DE-A
Mallat	**0.95 ± 0.02**	**0.95 ± 0.01**	0.89 ± 0.08	0.83 ± 0.26
Meat	0.73 ± 0.28	**0.96 ± 0.03**	0.83 ± 0.28	0.71 ± 0.34
MedicalImages	0.79 ± 0.02	0.79 ± 0.01	0.79 ± 0.01	**0.80 ± 0.02**
MiddlePhalanxOutlineAgeGroup	**0.56 ± 0.05**	**0.56 ± 0.05**	0.55 ± 0.03	0.54 ± 0.05
MiddlePhalanxOutlineCorrect	0.83 ± 0.02	0.83 ± 0.02	**0.84 ± 0.01**	0.81 ± 0.00
MiddlePhalanxTW	0.52 ± 0.05	**0.54 ± 0.04**	0.51 ± 0.04	**0.54 ± 0.03**
MixedShapesRegularTrain	**0.97 ± 0.00**	**0.97 ± 0.00**	**0.97 ± 0.00**	**0.97 ± 0.00**
MixedShapesSmallTrain	**0.92 ± 0.01**	**0.92 ± 0.01**	**0.92 ± 0.01**	**0.92 ± 0.01**
MoteStrain	**0.87 ± 0.03**	0.86 ± 0.02	**0.87 ± 0.03**	**0.87 ± 0.03**
NonInvasiveFetalECGThorax1	**0.96 ± 0.00**	0.95 ± 0.01	**0.96 ± 0.00**	**0.96 ± 0.00**
NonInvasiveFetalECGThorax2	0.96 ± 0.00	0.96 ± 0.00	0.96 ± 0.00	**0.97 ± 0.00**
OSULeaf	0.95 ± 0.01	0.96 ± 0.01	0.96 ± 0.01	**0.97 ± 0.02**
OliveOil	**0.85 ± 0.05**	0.77 ± 0.18	0.83 ± 0.06	0.82 ± 0.12
PhalangesOutlinesCorrect	0.83 ± 0.02	**0.84 ± 0.01**	**0.84 ± 0.00**	**0.84 ± 0.01**
Phoneme	**0.33 ± 0.02**	0.32 ± 0.02	0.32 ± 0.02	0.31 ± 0.01
PigAirwayPressure	**0.89 ± 0.03**	0.88 ± 0.04	0.71 ± 0.05	0.71 ± 0.03
PigArtPressure	**0.94 ± 0.0**4	0.93 ± 0.04	0.91 ± 0.05	0.90 ± 0.05
PigCVP	**0.90 ± 0.03**	**0.90 ± 0.03**	0.75 ± 0.04	0.78 ± 0.06
Plane	**1.00 ± 0.01**	**1.00 ± 0.00**	**1.00 ± 0.01**	**1.00 ± 0.00**
PowerCons	0.98 ± 0.01	0.98 ± 0.01	**0.99 ± 0.01**	0.98 ± 0.02
ProximalPhalanxOutlineAgeGroup	**0.82 ± 0.02**	0.80 ± 0.03	0.81 ± 0.02	0.80 ± 0.02
ProximalPhalanxOutlineCorrect	0.89 ± 0.01	**0.90 ± 0.02**	**0.90 ± 0.02**	**0.90 ± 0.02**
ProximalPhalanxTW	0.79 ± 0.02	0.79 ± 0.02	0.79 ± 0.03	**0.80 ± 0.03**
RefrigerationDevices	0.71 ± 0.10	**0.72 ± 0.09**	0.71 ± 0.10	**0.72 ± 0.11**
Rock	0.66 ± 0.12	0.60 ± 0.14	**0.68 ± 0.10**	**0.68 ± 0.08**
ScreenType	0.68 ± 0.06	**0.70 ± 0.07**	0.69 ± 0.07	0.69 ± 0.07
SemgHandGenderCh2	**0.88 ± 0.02**	**0.88 ± 0.01**	0.85 ± 0.04	0.87 ± 0.01
SemgHandMovementCh2	0.49 ± 0.07	0.54 ± 0.03	0.54 ± 0.03	**0.55 ± 0.04**
SemgHandSubjectCh2	0.68 ± 0.03	0.72 ± 0.04	**0.74 ± 0.07**	0.73 ± 0.03
ShapeletSim	**0.81 ± 0.08**	0.80 ± 0.05	**0.81 ± 0.07**	0.77 ± 0.05
ShapesAll	**0.93 ± 0.01**	**0.93 ± 0.01**	**0.93 ± 0.01**	0.92 ± 0.01
SmallKitchenAppliances	0.75 ± 0.04	0.75 ± 0.01	**0.77 ± 0.03**	**0.77 ± 0.03**
SmoothSubspace	**0.98 ± 0.01**	**0.98 ± 0.01**	**0.98 ± 0.01**	**0.98 ± 0.00**
SonyAIBORobotSurface1	**0.91 ± 0.06**	0.90 ± 0.06	0.87 ± 0.08	0.89 ± 0.06
SonyAIBORobotSurface2	**0.95 ± 0.01**	**0.95 ± 0.01**	**0.95 ± 0.01**	**0.95 ± 0.01**
StarLightCurves	**0.98 ± 0.00**	**0.98 ± 0.00**	**0.98 ± 0.00**	**0.98 ± 0.00**
Strawberry	**0.98 ± 0.01**	**0.98 ± 0.00**	**0.98 ± 0.00**	0.97 ± 0.00
SwedishLeaf	0.96 ± 0.01	**0.97 ± 0.01**	**0.97 ± 0.01**	**0.97 ± 0.01**
Symbols	**0.97 ± 0.01**	**0.97 ± 0.02**	**0.97 ± 0.01**	**0.97 ± 0.01**
SyntheticControl	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
ToeSegmentation1	0.96 ± 0.01	0.95 ± 0.01	0.96 ± 0.01	0.96 ± 0.01
ToeSegmentation2	0.96 ± 0.02	0.96 ± 0.01	0.96 ± 0.02	0.96 ± 0.01
Trace	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
TwoLeadECG	0.99 ± 0.01	**1.00 ± 0.00**	0.99 ± 0.01	0.98 ± 0.02
TwoPatterns	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
UMD	**0.99 ± 0.01**	0.98 ± 0.02	**0.99 ± 0.00**	**0.99 ± 0.00**
UWaveGestureLibraryAll	**0.95 ± 0.01**	0.94 ± 0.00	0.94 ± 0.00	**0.95 ± 0.01**
UWaveGestureLibraryX	**0.83 ± 0.00**	**0.83 ± 0.00**	**0.83 ± 0.01**	**0.83 ± 0.01**
UWaveGestureLibraryY	0.76 ± 0.00	**0.77 ± 0.01**	**0.77 ± 0.00**	0.76 ± 0.00
UWaveGestureLibraryZ	0.76 ± 0.01	**0.77 ± 0.01**	**0.77 ± 0.01**	**0.77 ± 0.01**
Wafer	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Wine	0.83 ± 0.14	0.79 ± 0.12	**0.84 ± 0.12**	0.81 ± 0.10
WordSynonyms	**0.74 ± 0.01**	**0.74 ± 0.02**	0.73 ± 0.02	**0.74 ± 0.02**
Worms	**0.80 ± 0.02**	0.79 ± 0.02	0.77 ± 0.04	0.79 ± 0.03
WormsTwoClass	0.83 ± 0.02	0.82 ± 0.06	0.83 ± 0.02	**0.84 ± 0.05**
Yoga	**0.92 ± 0.01**	0.91 ± 0.01	0.91 ± 0.01	**0.92 ± 0.01**

Table 4: (Continuation) Accuracy results across 112 datasets from the UCR archive. Values are reported as mean ± standard deviation across the 5 seeds.

To evaluate the impact of a validation set and assess possible improvements for models using concepts (i.e., models with more inputs), the training set was further split into training and validation subsets stratified sampling to preserve label distribution and ensure that each class was represented in both splits. By default, 15% was reserved for validation. For small training sets, especially when some classes are underrepresented, the validation size was adjusted to ensure that minority classes are not excluded. If a label appears only once in the training set, it was included in the training subset, and stratified sampling is applied to the rest. Instead of saving the model based on the best training loss, we saved the model based on the best validation loss.

As observed in Tables 5 and 6, models using fine-grained concepts benefited the most from the validation set, showing improved performance across 22 datasets, whereas the other two models improved in 14 datasets. However, most datasets exhibited a performance drop (81 for Baseline, 86 for CAB, 73 for DE-A, and 77 for DE-AC). Datasets such as *Earthquakes*, *Ham*, and *MiddlePhalanxOutlineAgeGroup*, which have more than 100 training signals, benefited from the validation set. In contrast, datasets such as *FaceFour*, *Beef*, and *OliveOil*, which have 30 or fewer training signals, experienced a substantial drop in performance, likely due to the small size of the validation subset.

Model analysis. We evaluated the impact of including concepts in the model (analysis performed with seed 0) in two ways:

- **Representation Shift:** Cosine similarity between the H-InceptionTime backbone features of the Baseline and DE-A. Lower similarity indicates a greater shift in internal representations caused by concept supervision.
- **Concept Contribution:** Median class-wise relative contribution of concept features to predictions, computed using classifier weights and average concept activations. Higher values indicate greater reliance on concept-level information.

For this analysis, we selected datasets that were farther from the diagonal in the confrontation maps presented in the main paper. Datasets with higher concept contribution, such as *Mallat*, *PigCVP*, and *SemgHandMovementCh2*, also exhibit larger shifts in the H-InceptionTime backbone representations (cosine similarity ranged from 57 to 71%; concept contribution ranged from 24 to 27%), indicating a stronger reliance on concept features for predictions. Performance-wise, *Mallat* remains almost unchanged, *PigCVP* drops by 10.1%, and *SemgHandMovementCh2* improves by 9.5%. Furthermore, datasets with low concept contribution, like *Meat* and *OliveOil* (concept contribution ranged from 5 to 6%; representation shift from 66 to 78%), show smaller changes in accuracy, with *Meat* remaining similar and *OliveOil* improving by 6.6%, reflecting the predominance of backbone features in classification.

Overall, these results suggest that concept supervision preserves accuracy while meaningfully altering internal representations, highlighting the influence of interpretable, human-aligned concepts as inputs.

Memory and runtime analysis. Baseline and CAB models exhibited similar GPU memory consumption (1.7GB), while DE-A and DE-AC used more memory (3.3–3.4GB) for *ACSF1*, reflecting the higher requirements of heavily interpolated signals. Despite the increase, the larger models do not require prohibitively more memory. Runtime per seed was approximately 68-70min for DE-A and DE-AC, 7min for CAB, and 3min for Baseline. For the *Adiac* dataset, runtimes were approximately 25min for DE-A and DE-AC, 17min for CAB, and 11min for Baseline, with memory consumption of 0.35GB for DE-A and DE-AC and 0.24GB for Baseline and CAB. These results indicate that very high interpolation increases both runtime and memory requirements, whereas standard interpolation to the model input size adds only a modest overhead.

These two datasets were conducted on NVIDIA GPUs as follows: Baseline, DE-A, and DE-AC on L40S, and CAB on V100S.

	Baseline	Δ Baseline	CAB	Δ CAB	DE-AC	Δ DE-AC	DE-A	Δ DE-A
ACSF1	0.82	-0.11	0.85	-0.03	**0.86**	-0.07	0.84	-0.08
Adiac	0.83	-0.00	**0.86**	0.03	0.84	0.02	0.85	0.01
ArrowHead	0.83	-0.06	0.83	-0.04	0.81	-0.02	**0.87**	-0.03
BME	**0.99**	0.00	**0.99**	-0.01	**0.99**	-0.01	**0.99**	0.00
Beef	**0.20**	-0.53	**0.20**	-0.50	**0.20**	-0.53	**0.20**	-0.57
BeetleFly	0.65	-0.10	0.65	-0.10	**0.80**	0.05	0.70	-0.05
BirdChicken	**0.90**	-0.05	0.85	0.00	**0.90**	-0.05	0.85	0.00
CBF	**0.99**	-0.01	**0.99**	-0.01	**0.99**	-0.01	0.98	-0.01
Car	**0.92**	0.02	0.87	-0.03	0.90	0.02	0.90	0.00
Chinatown	0.49	-0.48	0.57	-0.41	**0.78**	-0.20	0.48	-0.50
ChlorineConcentration	**0.85**	-0.04	0.84	-0.04	0.82	-0.05	0.84	-0.05
CinCECGTorso	**0.84**	0.01	0.81	-0.00	0.80	-0.04	0.80	-0.01
Coffee	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
Computers	0.77	-0.01	0.76	-0.01	**0.82**	0.02	0.81	0.02
CricketX	0.82	-0.02	0.79	-0.04	0.81	-0.03	**0.84**	-0.01
CricketY	**0.81**	-0.04	0.76	-0.07	0.79	-0.05	**0.81**	-0.03
CricketZ	0.82	-0.03	**0.83**	-0.01	0.79	-0.04	0.81	-0.04
Crop	0.70	-0.07	0.69	-0.08	0.71	-0.07	**0.72**	-0.05
DiatomSizeReduction	0.84	-0.11	0.83	-0.10	**0.85**	-0.10	0.83	-0.10
DistalPhalanxOutlineAgeGroup	0.65	-0.07	0.68	-0.07	0.71	-0.03	**0.73**	0.01
DistalPhalanxOutlineCorrect	**0.79**	0.00	0.75	-0.04	0.78	0.00	0.71	-0.05
DistalPhalanxTW	0.46	-0.20	**0.63**	-0.02	0.56	-0.07	0.58	-0.07
ECG200	0.88	-0.03	0.89	-0.02	0.86	-0.03	**0.94**	0.04
ECG5000	0.90	-0.04	0.90	-0.04	**0.92**	-0.02	0.91	-0.03
ECGFiveDays	**1.00**	0.00	**1.00**	-0.00	**1.00**	-0.00	**1.00**	0.00
EOGHorizontalSignal	0.64	-0.02	0.64	0.01	**0.67**	0.04	0.62	-0.01
EOGVerticalSignal	0.48	-0.02	0.46	-0.03	0.49	-0.02	**0.52**	0.06
Earthquakes	0.75	0.02	0.68	-0.01	**0.76**	0.04	**0.76**	0.05
ElectricDevices	**0.72**	0.01	0.68	-0.03	0.69	-0.02	**0.72**	0.00
EthanolLevel	0.68	-0.11	0.74	-0.04	0.69	-0.10	**0.76**	-0.03
FaceAll	0.80	-0.00	**0.82**	-0.02	0.81	-0.01	0.81	-0.00
FaceFour	0.16	-0.78	0.16	-0.80	**0.30**	-0.66	0.25	-0.70
FacesUCR	0.95	-0.01	0.95	-0.01	0.95	-0.01	**0.96**	-0.01
FiftyWords	0.77	-0.06	**0.80**	-0.02	0.73	-0.07	0.76	-0.07
Fish	**0.98**	-0.01	**0.98**	0.01	**0.98**	0.01	**0.98**	0.01
FordA	0.95	-0.01	0.95	0.00	**0.96**	0.00	0.95	-0.01
FordB	0.81	-0.02	**0.83**	-0.01	0.80	-0.05	0.82	-0.02
FreezerRegularTrain	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
FreezerSmallTrain	0.80	-0.01	0.69	-0.10	0.82	-0.01	**0.85**	-0.01
GunPoint	0.99	-0.01	0.99	-0.01	**1.00**	0.00	0.99	-0.01
GunPointAgeSpan	0.99	-0.00	0.98	-0.01	**1.00**	0.01	0.99	0.00
GunPointMaleVersusFemale	**1.00**	-0.00	**1.00**	0.00	**1.00**	-0.00	**1.00**	0.00
GunPointOldVersusYoung	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
Ham	0.71	0.00	**0.76**	0.03	0.70	0.02	0.73	0.07
HandOutlines	0.94	-0.01	0.94	-0.03	**0.95**	-0.01	**0.95**	-0.00
Haptics	0.47	-0.08	**0.52**	-0.03	0.47	-0.07	0.49	-0.08
Herring	**0.59**	-0.12	**0.59**	-0.16	0.41	-0.27	**0.59**	-0.17
HouseTwenty	0.92	-0.02	0.92	0.00	0.93	-0.01	**0.94**	0.00
InlineSkate	0.19	-0.32	**0.47**	-0.05	0.41	-0.11	0.38	-0.14
InsectEPGRegularTrain	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
InsectEPGSmallTrain	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
InsectWingbeatSound	**0.62**	-0.01	0.60	-0.04	**0.62**	-0.01	**0.62**	-0.02
ItalyPowerDemand	0.95	-0.02	**0.97**	-0.00	0.96	-0.01	0.96	-0.01
LargeKitchenAppliances	**0.90**	-0.01	0.86	-0.03	0.86	-0.04	**0.90**	0.00
Lightning2	0.77	-0.02	0.77	-0.02	**0.79**	0.00	0.74	-0.08
Lightning7	0.74	-0.10	0.64	-0.19	**0.77**	-0.07	0.71	-0.08

Table 5: Accuracy results across 112 datasets from the UCR archive when using a validation set. Values are reported for the original seed, and ΔBaseline, ΔCAB, ΔDE-AC, and ΔDE-A indicate the performance difference between using a validation set and not (on the original split).

	Baseline	Δ Baseline	CAB	Δ CAB	DE-AC	Δ DE-AC	DE-A	Δ DE-A
Mallat	0.94	0.01	**0.95**	0.00	0.89	-0.00	0.93	0.00
Meat	0.92	-0.03	0.92	-0.02	0.93	-0.02	**0.95**	0.00
MedicalImages	**0.77**	0.00	0.71	-0.07	0.75	-0.03	**0.77**	-0.02
MiddlePhalanxOutlineAgeGroup	**0.60**	0.07	0.46	-0.03	0.55	0.05	0.54	0.03
MiddlePhalanxOutlineCorrect	**0.84**	-0.01	0.80	-0.02	0.81	-0.04	0.82	0.01
MiddlePhalanxTW	0.42	-0.10	0.24	-0.27	**0.47**	0.00	0.41	-0.08
MixedShapesRegularTrain	**0.97**	0.00	**0.97**	-0.00	**0.97**	-0.00	**0.97**	-0.01
MixedShapesSmallTrain	0.90	-0.02	0.90	-0.03	0.86	-0.06	**0.91**	-0.02
MoteStrain	**0.85**	-0.05	0.69	-0.20	**0.85**	-0.03	0.84	-0.03
NonInvasiveFetalECGThorax1	**0.96**	0.00	0.95	-0.01	0.95	-0.00	**0.96**	-0.01
NonInvasiveFetalECGThorax2	0.95	-0.01	0.95	-0.01	0.94	-0.01	**0.96**	-0.01
OSULeaf	0.90	-0.03	0.94	0.00	**0.96**	0.00	0.94	0.00
OliveOil	0.40	-0.40	0.40	-0.40	0.40	-0.37	**0.53**	-0.40
PhalangesOutlinesCorrect	0.80	-0.00	0.83	0.00	**0.84**	0.01	**0.84**	0.01
Phoneme	**0.23**	-0.12	0.22	-0.12	0.22	-0.12	0.17	-0.16
PigAirwayPressure	**0.80**	-0.15	0.78	-0.16	0.44	-0.27	0.49	-0.23
PigArtPressure	**0.79**	-0.20	0.74	-0.26	0.72	-0.28	0.74	-0.26
PigCVP	**0.81**	-0.14	0.80	-0.15	0.63	-0.16	0.65	-0.20
Plane	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
PowerCons	0.97	-0.02	**0.99**	0.01	0.98	-0.01	**0.99**	-0.01
ProximalPhalanxOutlineAgeGroup	**0.84**	0.00	0.80	-0.04	0.81	-0.01	**0.84**	0.02
ProximalPhalanxOutlineCorrect	**0.90**	-0.01	**0.90**	-0.02	0.89	-0.04	0.89	-0.03
ProximalPhalanxTW	**0.82**	0.02	0.78	-0.02	0.80	0.01	0.80	0.00
RefrigerationDevices	0.49	-0.05	**0.54**	-0.01	0.50	-0.04	0.32	-0.20
Rock	**0.68**	0.16	**0.68**	0.10	0.52	-0.14	**0.68**	-0.04
ScreenType	0.53	-0.05	0.55	-0.04	**0.58**	-0.00	**0.58**	0.01
SemgHandGenderCh2	**0.86**	0.00	**0.86**	-0.02	**0.86**	0.01	**0.86**	-0.01
SemgHandMovementCh2	0.50	0.11	0.41	-0.09	0.51	-0.03	**0.54**	-0.01
SemgHandSubjectCh2	**0.70**	0.00	0.62	-0.11	0.65	-0.10	0.64	-0.08
ShapeletSim	0.77	-0.15	0.84	-0.02	**0.87**	-0.03	0.86	0.04
ShapesAll	**0.90**	-0.01	**0.90**	-0.01	0.88	-0.03	0.88	-0.03
SmallKitchenAppliances	0.78	0.06	**0.79**	0.03	0.75	0.02	**0.79**	0.03
SmoothSubspace	0.95	-0.02	0.96	-0.02	**0.97**	-0.01	0.96	-0.03
SonyAIBORobotSurface1	0.77	-0.05	**0.82**	-0.01	0.81	0.04	0.77	-0.05
SonyAIBORobotSurface2	**0.95**	-0.02	0.94	-0.01	0.94	-0.00	0.94	-0.01
StarLightCurves	0.97	-0.00	0.96	-0.01	**0.98**	0.00	0.97	-0.00
Strawberry	0.98	-0.01	0.98	0.00	0.98	0.00	**0.99**	0.01
SwedishLeaf	**0.97**	0.00	0.96	-0.01	0.96	-0.01	0.96	-0.01
Symbols	0.97	-0.01	**0.98**	-0.00	0.97	-0.01	0.97	-0.01
SyntheticControl	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	0.98	-0.02
ToeSegmentation1	0.95	-0.01	0.95	-0.00	**0.96**	-0.00	**0.96**	0.00
ToeSegmentation2	0.92	0.00	0.92	-0.02	**0.94**	0.02	0.93	-0.02
Trace	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
TwoLeadECG	0.96	-0.04	**0.99**	-0.01	**0.99**	-0.01	0.95	-0.05
TwoPatterns	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
UMD	0.98	-0.01	**0.99**	0.00	**0.99**	0.00	**0.99**	-0.01
UWaveGestureLibraryAll	**0.94**	-0.00	**0.94**	-0.01	**0.94**	-0.00	0.92	-0.02
UWaveGestureLibraryX	0.78	-0.04	0.74	-0.09	**0.80**	-0.03	0.79	-0.03
UWaveGestureLibraryY	0.71	-0.05	0.74	-0.02	**0.76**	-0.01	0.75	-0.02
UWaveGestureLibraryZ	0.70	-0.07	**0.75**	-0.02	0.72	-0.05	0.73	-0.04
Wafer	0.99	-0.00	**1.00**	-0.00	**1.00**	0.00	**1.00**	0.00
Wine	0.63	-0.04	**0.67**	0.06	0.63	-0.04	0.61	-0.06
WordSynonyms	0.67	-0.06	**0.68**	-0.05	**0.68**	-0.04	0.66	-0.07
Worms	0.51	-0.27	**0.70**	-0.12	0.51	-0.32	0.55	-0.26
WormsTwoClass	0.61	-0.18	0.57	-0.16	**0.66**	-0.17	0.64	-0.10
Yoga	0.89	-0.02	0.87	-0.04	0.89	-0.02	**0.90**	-0.02

Table 6: (Continuation) Accuracy results across 112 datasets from the UCR archive when using a validation set. Values are reported for the original seed, and Δ Baseline, Δ CAB, Δ DE-AC, and Δ DE-A indicate the performance difference between using a validation set and not (on the original split).