Effectiveness of Machine Learning Models in Classifying Learners Based on Learning Curves

Improving Our Understanding of Learning Curves Through the Process of Classification

Sinan Basaran Karaarslan
Supervisor(s): Tom Viering, Cheng Yan, Sayak Mukherjee
[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Sinan Basaran Karaarslan
Final project course: CSE3000 Research Project
Thesis committee: Tom Viering, Cheng Yan, Sayak Mukherjee, Matthijs Spaan

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Abstract

In machine learning, learning curves are a metric that plots performance versus training set size. They inform decisions about data acquisition, model selection, and hyperparameter tuning. Despite their importance, recent research suggests that our understanding of learning curve behavior remains limited. In this work, we investigate learning curves from a classification perspective to better understand their structural properties. By framing learning curves as time series and applying time series classification (TSC) techniques, we uncover several key findings: (1) training accuracy curves are significantly more distinguishable across models than validation or test curves; (2) learning curves become more informative and discriminative after a sufficient number of anchor points; and (3) TSC models that emphasize global structural features outperform those focused on local or pointwise characteristics. These results not only offer new insights into the nature of learning curves but also suggest promising directions for future work, including the development of specialized models that move beyond conventional time series assumptions.

1 Introduction

Learning curves describe the performance of a machine learning model on a specific task as a function of the training set size used to solve that task [1]. Learning curves are valuable for several reasons: they help estimate how much data is needed to reach a desired level of performance, support early stopping when further training is unlikely to improve results, and enable early discarding of underperforming models during model selection [1]. In machine learning, it is common to use parametric formulas to extrapolate learning curves and make decisions based on these predictions. These formulas assume that performance generally improves as the training set grows [2]. While this is often true, real-world learning curves can behave unexpectedly, showing what are referred to as 'ill-behaving curves' [2]. As a result, there is a growing need for new models that can handle such irregular patterns more effectively.

Recent research has shown that deep learning-based approaches may offer more accurate extrapolations than traditional parametric methods. Notably, the LC-PFN [3], a transformer-based model trained on synthetic curves to approximate Bayesian inference, has demonstrated superior performance. Despite progress in modeling the shape of learning curves [2], recent findings continue to reveal unexpected patterns [4], suggesting that our understanding of learning curves remains incomplete and further exploration is warranted.

One promising direction for deepening this understanding is through classification. Prior work has observed that different learning algorithms yield distinctly shaped learning curves [5]. Building on this observation, this work investigates whether it is possible to classify learners based on the shapes of their learning curves. In doing so, we aim to explore the conditions under which learning curves are most distinguishable, and which classification methods are most effective. Through the process of classification, we may extend our knowledge on learning curves, and be able to better extrapolate them in the future. To the best of our knowledge, this is the first systematic attempt to classify learning curves by learner identity.

The key idea of this work is to treat learning curves as time series, as they exhibit core characteristics typical of time series data. To evaluate this approach, four experiments are conducted. First, binary classifiers are trained on all possible pairs of learners to explore similarities in their learning curve behavior. The second and third experiments examine the impact of different data representations on classification performance. Lastly, a range

of classifiers is tested, including state-of-the-art models from each major category of time series classification.

The remainder of this paper is structured as follows. Section 2 lays out the related work for this research. In Section 3, the core idea of treating learning curves as time series is introduced and the motivations behind this approach are outlined. Section 4 describes the experimental setup in detail, which is followed by Section 5 presenting the results of our classification experiments. In Section 6, the implications of the findings are discussed, limitations highlighted, and the results interpreted in light of the research goals. Section 7 concludes the paper and outlines possible directions for future work. Finally, Section 8 addresses responsible research practices, including considerations around reproducibility, data transparency, and ethical use of resources.

2 Related Work

This section reviews prior research that forms the foundation of this study. It covers three areas: (i) theoretical and empirical insights into learning curve behavior, (ii) data resources and models central to this work, and (iii) time series classification (TSC) methods that enable the proposed approach.

2.1 Learning Curve Behavior and Evaluation Practices

A foundational contribution to this study is the comprehensive literature review on learning curve shapes by Viering et al. [2]. Investigations into the shape and behavior of learning curves reveal that while model performance often improves with more training data, deviations from this trend—commonly referred to as 'ill-behaving curves'—are not uncommon [2]. These findings challenge the assumptions underlying many traditional parametric extrapolation formulas. Learning curves have also been shown to play a crucial role in guiding decisions around data acquisition, model selection, and early stopping [1].

Further, since a major part of this research involves comparing the performance of various classifiers, it is important to acknowledge prior work on best practices for classifier evaluation. Such prior work includes studies focused on ensuring replicability of experiments [6], the use of appropriate statistical tests to assess significance [7], and broader guidelines on how to conduct reliable comparisons between learning algorithms [8, 9]. These works helped guide the experimental setup and result interpretation in this paper.

2.2 The LCDB and the LC-PFN

Two key contributions in recent literature are central to this study: the Learning Curve Database (LCDB) [5] and the Learning Curve Prior-Fitted Network (LC-PFN) [3].

The LCDB serves as the primary data source for our experiments. This research uses version 1.1 of the LCDB, currently the largest available database of its kind [4]. The LCDB contains prediction and ground truth vectors for 24 classification algorithms across 265 datasets. These predictions are recorded at various training set sizes, referred to as anchors [5]. For each anchor point, there are 25 different data splits, providing 25 accuracy values per learner per dataset. The anchor points are consistent across all learning curves, meaning that while each curve may have different lengths, the recorded data points are always at the same positions. Further, the LCDB contains train, validation, and test accuracy curves for all of the learners.

The LC-PFN is, to our knowledge, the only deep learning model shown to consistently outperform traditional parametric methods in learning curve extrapolation. It uses a transformer architecture trained on a large set of synthetic learning curves to approximate Bayesian posterior inference [3]. Its promising results highlight the potential of neural models in this domain and motivate further research into deeper representations of learning curve dynamics. This study complements such efforts by focusing on classification as a means of understanding the structural properties of learning curves: insights which could, in turn, inform the design of better extrapolation models.

2.3 Time Series Classification Models

Given that learning curves evolve sequentially with training set size, they are naturally suited to time series modeling. A detailed discussion of this perspective is provided in Section 3. To this end, a diverse range of TSC models is evaluated in this study. To ensure broad coverage, we include representative models from each of the eight broad types of TSC models defined in the recent paper by Guijo-Rubio et al. [10].

- **Distance-based:** These models rely on distance metrics to assess similarity between time series. We use *shapeDTW* [11], which enhances dynamic time warping (DTW) by taking point-wise local structural information into consideration.
- **Feature-based:** These extract statistical features from the entire series and train a classifier on the resulting vectors. As a baseline for feature-based classifiers, FreshPRINCE [12] will be used in our experiments.
- **Interval-based:** A subset of feature-based models, these extract features from random intervals within the time series and build ensembles over them. We employ *DrCIF* [13], an extension of the Canonical Interval Forest (CIF) classifier [14], which combines diverse interval representations for improved performance.
- **Kernel/convolution-based:** These models apply convolutional operations over subseries and use the results as features. We evaluate the two variants of the original *ROCKET* [15] classifier: its more computationally efficient counterpart *MiniRocket* [16], and the performance-optimized *MultiRocket* [17], all of which use random kernels, pooling operations, and a ridge classifier.
- **Deep learning:** Deep neural networks are widely used for TSC due to their scalability and ability to learn complex patterns [18]. We use *InceptionTime* [19], a CNN-based model inspired by Inception modules from image classification. Following best practices, we use an ensemble of five InceptionTime models for improved stability and accuracy.
- **Shapelet-based:** These models classify time series based on the presence of discriminative subsequences (shapelets). We use a shapelet transform classifier implemented in `sktime` [20], which follows the approaches described in [21, 22].
- **Dictionary-based:** These models use a bag of words-like approach to base classification on the number of occurrences of approximated subseries (patterns) [10]. For our experiments we will use the Temporal Dictionary Ensemble (TDE) classifier [23], as it is the state-of-the-art model for this category.

- **Hybrid:** Finally, we evaluate *HIVE-COTE 2.0* [13], the most accurate known ensemble model, and the state-of-the-art on the UCR archive [24]. HIVE-COTE integrates classifiers from several categories, including TDE, DrCIF, Arsenal (an ensemble of ROCKET models), and a shapelet transform classifier.

3 Learning Curves as Time Series

Time series classification (TSC) involves predicting a discrete target variable from a (potentially multivariate) time series [13]. TSC methods have proven effective in domains such as healthcare [25] and seismology [26], where sequential data carries critical information. In this work, we adopt a similar perspective by treating learning curves as time series. While learning curves are not time series in the traditional sense, they exhibit several defining characteristics that justify this approach.

A time series is typically defined as "a set of data collected at usually equally spaced points in time" [27]. Although learning curves are not indexed by time, the training set size can act as a substitute for the time axis, since it increases in a fixed and monotonic order. Additionally, while anchor points may not represent equally spaced intervals in terms of training size, they are consistent across all curves. This consistency allows the treatment of them as equally spaced time steps for the purpose of modeling.

One of the key challenges in applying TSC methods to learning curves stems from the assumptions baked into many state-of-the-art models. Most of these models are developed and benchmarked using the UCR archive [24], which has become the standard testbed for TSC. However, the UCR archive has received criticism for its lack of diversity in sequence characteristics [24]. In particular, it consists entirely of uniform-length time series, where each instance has the same number of time steps. This feature represents an idealized scenario that does not always reflect real-world data [28], and learning curves are a clear example of this deviation, as their lengths can vary depending on the learner and dataset. As a result, many of the models used in our experiments required learning curves to be preprocessed into uniform-length sequences before training, prompting an exploration of various preprocessing strategies. This distinction informed both the model selection and experimental design in this study.

4 Experimental Setup

As previously mentioned, the experiments in this study are based on version 1.1 of the Learning Curve Database (LCDB) [4]. To ensure robustness, the LCDB provides data from 25 splits of learning curves for each learner in each dataset. To reduce the variance introduced by individual splits and obtain a more stable representation of each curve, the mean accuracy across these 25 splits was computed. This preprocessing step resulted in a total of 6360 learning curves prior to any further modification.

Since the task involves classification, we evaluate model performance using five-fold stratified cross-validation across all experiments. This approach ensures that class distributions are preserved within each fold, mitigating the effects of class imbalance and improving the reliability of the reported metrics. Additionally, in experiments 2 through 4, we included a dummy classifier to provide a baseline for comparison and better contextualize our results.

A key challenge in working with these learning curves is that they vary in length, due to differences in how many anchor points are present across datasets. In the context of this

study, the "length" of a curve refers to the number of accuracy values it originally contained, before any padding was applied. Many time series classification models (including those used in this research) require input sequences to be of equal length. To address this, all learning curves were padded with zeros at the end to match the length of the longest curve. Notably, the LCDB includes both error rate and classification accuracy curves for each learner. In this work, we opted to use the accuracy curves, as zero-padding is more appropriate in that context.

4.1 Grouping Classifiers For a Reasonably Difficult Classification Task

Experiment 2–4 in this study involve multiclass classification tasks spanning all 24 learners from the LCDB. Initially, each learner was treated as a distinct class label. However, this formulation proved overly challenging for most models: classification accuracies were uniformly low, and performance differences between classifiers were too small to yield meaningful comparisons.

In order to conduct more informative experiments, the task was reformulated by grouping learners into broader categories, thereby reducing the complexity of the classification problem. The goal was to create a version of the task that remained non-trivial, yet allowed for clearer differentiation among models and more interpretable results.

A total of six learner groups were constructed, with each group containing learners exhibiting similar learning curve characteristics. The primary basis for this grouping was the results from Experiment 1, where low pairwise binary classification accuracy between learners indicated a high degree of similarity in their learning curves. These empirical similarities were then supported, where possible, by theoretical or practical knowledge about the underlying behavior of the respective learners. Additionally, care was taken to minimize class imbalance among the groups. While complete uniformity was not always possible, an effort was made to keep the number of learners in each group relatively even, ensuring that no single group dominated the classification task due to data volume alone. The exact grouping is as follows:

- **Group 1:** Perceptron, PassiveAggressive, SGDClassifier, SVC_linear
- **Group 2:** Decision Tree, ExtraTree, MLP
- **Group 3:** SVC_poly, SVC_rbf, ens.ExtraTrees, ens.RandomForest, KNN
- **Group 4:** MultinomialNB, ComplementNB, GaussianNB, BernoulliNB, NearestCentroid
- **Group 5:** RidgeClassifier, LogisticRegression, LDA
- **Group 6:** DummyClassifier, SVC_sigmoid, QDA, ens.GradientBoosting

For a more detailed explanation on the formulation of each group, see Appendix A.3

4.2 Specific Experiment Setups

Until now, this section has focused on the general setup that was applied collectively to all experiments, or a group of them. Now, we will look into the specific setups of each experiment.

Experiment 1 focused on binary classification between each possible pair of learners. The binary classification was based on the sample-wise accuracy curves of the learners, and curves below a length of 50 were removed from the dataset. For the binary classification task, MiniRocket [16] was used as the classifier for its strong baseline performance and computational efficiency. The experiment was conducted separately for the train, validation, and test accuracy curves respectively.

Experiment 2 examined the trade-off between classifier performance versus increasing minimum length cutoffs for curves. Unlike Experiment 1, this and the subsequent experiments were multi-class classification tasks, where classifiers were trained to distinguish among all 24 learners (grouped into 6 groups) rather than performing pairwise binary classification. In the context of the experiment, the minimum length cutoff means that all curves that have a length below the minimum length cutoff point are discarded from the sample set. Specifically, we used validation accuracy curves in this experiment, as once again, they are more independent to the training process compared to the test and train accuracy curves [3]. Finally, to explore this trade-off, we train two classifiers: MiniRocket [16] and FreshPRINCE [12].

Experiment 3 revolved around the idea that the different types of accuracy curves provided to us by the LCDB [5], as well as the combinations of these curves, could provide further distinguishability among the learners. To explore this idea, we conducted an experiment using the MiniRocket [16] classifier, selected again for its strong baseline performance and computational efficiency. We tested various combinations of learning curves by concatenating them into a single input sequence. Padding was applied in a consistent manner: each curve was followed by zero-padding up to a fixed index, ensuring that the starting index of each subsequent curve was aligned across all samples (e.g., the second curve always started at index 133), resulting in uniform input lengths for the classifier. To maintain data quality, only curves with a minimum length of 50 were included.

Experiment 4 expanded the scope to include a broad range of classifiers, representing all eight major time series classification (TSC) model categories outlined in Section 2. The implementations of each of the algorithms used in this experiment was taken from the `sktime` [20] library. See Appendix A.1 for the hyperparameter configurations of each model used. In this experiment, classification was performed using a combination of training and validation accuracy curves. This choice was informed by the results of the third experiment (see Figure 3), where the train+validation combination significantly outperformed single-curve classification scores. Train+validation was preferred over train+test, as validation curves offer greater independence from the training process and the final evaluation [3]. Furthermore, while incorporating all three curve types produced similar classification accuracy, omitting test curves improved computational efficiency without sacrificing performance.

5 Results

To evaluate the distinguishability of learning curves across different learners, we conducted a series of classification experiments using version 1.1 of the LCDB [4]. Each experiment investigates a specific aspect of this question, from pairwise learner comparisons to model type and input curve selection.

5.1 Experiment 1: Pairwise Binary Classifiers

One of the main research questions in this study is whether learning curves from different learners can be reliably distinguished from one another. Given the 24 learners in the LCDB, a multi-class classification approach across all learners was deemed infeasible due to the complexity that such a task would introduce. Instead, a more targeted approach was adopted: binary classifiers were trained for each possible pair of learners (a total of 276 pairs), allowing a focused evaluation into how distinguishable two learners are based on their learning curves.

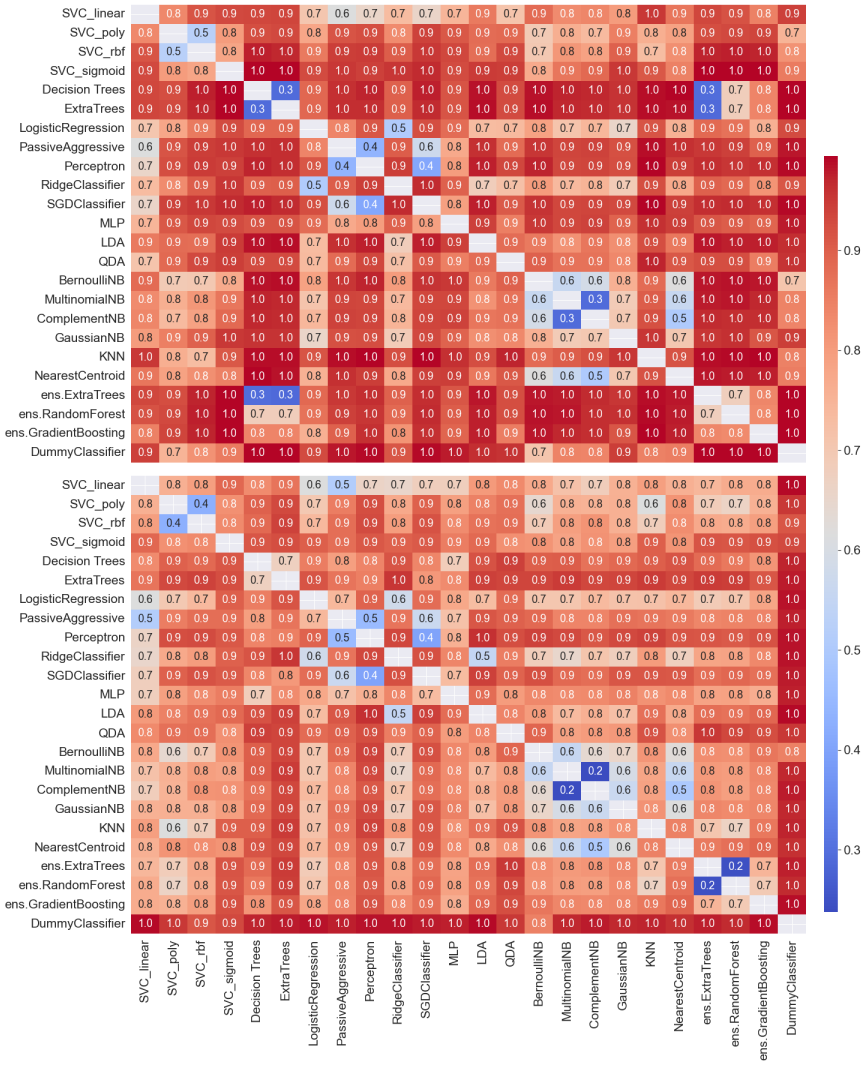

Figure 1: Pairwise binary classification accuracy matrix for: train (top), and validation (bottom) accuracy curves. Exact accuracy for the binary classifier trained on learner A and learner B is written on their intersection.

The results are presented in Figure 1. A diverse range of accuracy values is observed across learner pairs. Certain learner groups (such as variants of Naive Bayes) exhibit high similarity, making them difficult to distinguish. Conversely, some learners, including the *DummyClassifier*, *SVC_sigmoid*, and *QDA*, consistently yield high classification accuracy across all pairings, suggesting that their learning curves are notably distinct from those of other learners.

An additional observation emerges when comparing the structure of the classification matrices derived from different types of accuracy curves (train, validation, and test). Learners that exhibit similar validation accuracy curves tend to have similarly shaped test accuracy curves, and vice versa, suggesting a strong alignment between these two curve types. However, the relationship between training accuracy curves and the other two is less consistent. For instance, while *DecisionTree* and *ExtraTrees* show similar training accuracy curves to ensemble methods like *ens.ExtraTrees* and *ens.RandomForest*, their validation and test accuracy curves remain more easily distinguishable. Notably, training accuracy curves tend to yield higher classification accuracy overall, implying they are more effective at differentiating between learners. Due to the high similarity between validation and test accuracy matrices, the test accuracy matrix is not shown here to avoid redundancy and is instead included in Appendix A.2. These observations highlight the importance of curve type when analyzing learner similarity.

5.2 Experiment 2: Different Minimum Length Cutoffs

A key motivation behind this experiment is to investigate whether certain unique patterns in learning curves, particularly those associated with 'ill' behavior, tend to emerge only after a sufficient number of anchor points. If this is the case, then shorter curves may contain insufficient information for reliable classification, whereas longer curves may exhibit more distinctive features.

As previously discussed, learning curves in the LCDB vary in length due to factors such as differences in dataset sizes. It is important to understand how the minimum required length of a learning curve impacts both the classification performance and the number of usable samples. Intuitively, longer curves should offer better performance, as they provide more temporal information. However, applying a length-based filter also reduces the available training data, potentially introducing issues such as class imbalance, overfitting, and increased variance across validation splits. The results of this trade-off are shown in Figure 2.

From Figure 2, a general trend can be observed: increasing the minimum length cutoff tends to improve classification accuracy. However, this improvement plateaus and eventually reverses beyond a cutoff of approximately 90 anchor points. At this stage, the number of eligible curves drops sharply, resulting in a smaller and potentially less representative training set. This reduction not only diminishes model generalizability but also increases variance across validation folds, as seen in the rising standard deviation beyond the 110-point cutoff.

Some minor deviations from the overall trend are also observed. For instance, the relatively low variance at the 100-point cutoff and a slight uptick in accuracy between the 110 and 120 cutoffs for the FreshPRINCE classifier may reflect randomness due to limited sample sizes, rather than consistent performance gains.

Additionally, a dummy classifier was included as a baseline in this experiment. As anticipated, its accuracy and variance remained constant regardless of the minimum length cutoff applied. Specifically, the dummy classifier consistently achieved an accuracy of ap-

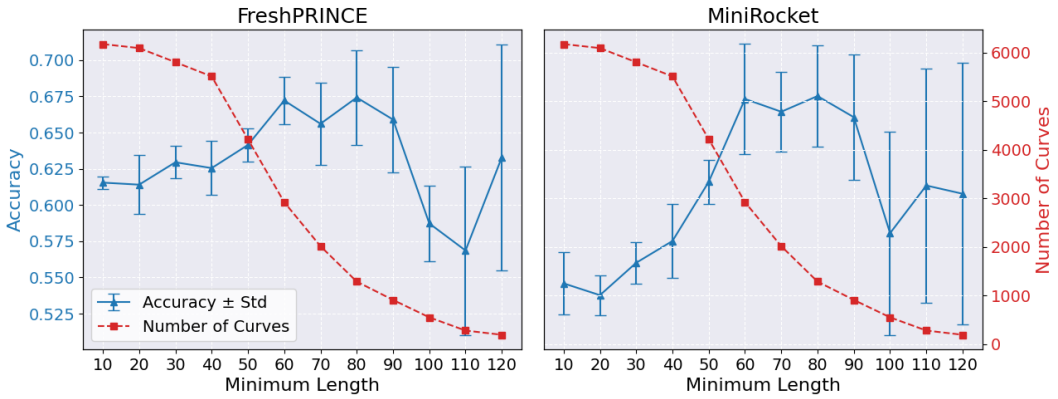

Figure 2: Accuracy ± standard deviation (blue) and number of curves left after cutoff (red) for the FreshPRINCE (left) and MiniRocket (right) classifiers as minimum length cutoff is increased.

proximately 0.236, with a standard deviation around 0.001 across all cutoff values. These stable results were omitted from Figure 2 to maintain clarity and readability of the plots.

5.3 Experiment 3: Different Combination of Curves

Until this point, the experimental focus has primarily centered on validation accuracy curves. This choice was partly motivated by the precedent set in the LC-PFN paper [3], where validation curves were used for extrapolation since they are more independent compared to training and test accuracy curves. However, upon examining the results of the pairwise classification experiments for the training and test accuracy curves (see Figures 1 and 5), we hypothesized that incorporating multiple types of curves might improve classification performance. Notably, training accuracy curves appeared to differ more across learners, whereas validation and test accuracy curves showed greater similarity. To explore this idea, we tested various combinations of learning curves by concatenating them into a single input sequence.

The results, summarized in Figure 3, reveal several key insights. First, classification based on training accuracy curves alone outperforms using only validation or test accuracy curves. This supports our initial hypothesis that training curves are more distinguishable across learners. Second, the similarity between validation and test curves is reflected in their comparable classification results; switching between them (e.g., validation-only vs. test-only, or validation+train vs. test+train) yields similar accuracies. This is further supported by the observation that combining validation and test curves does not yield improvements over using either curve individually.

Overall, the highest performance was achieved when training accuracy curves were combined with either validation, test, or both. A t-test analysis confirmed that the combinations train+validation, train+test, and all three curves significantly outperformed all other combinations, while showing no statistically significant differences among themselves. In addition, using only training accuracy curves was significantly better than using only validation, only test, or validation+test. These findings suggest that training curves contain the most informative signals for classification, and that combining them with other curve types may

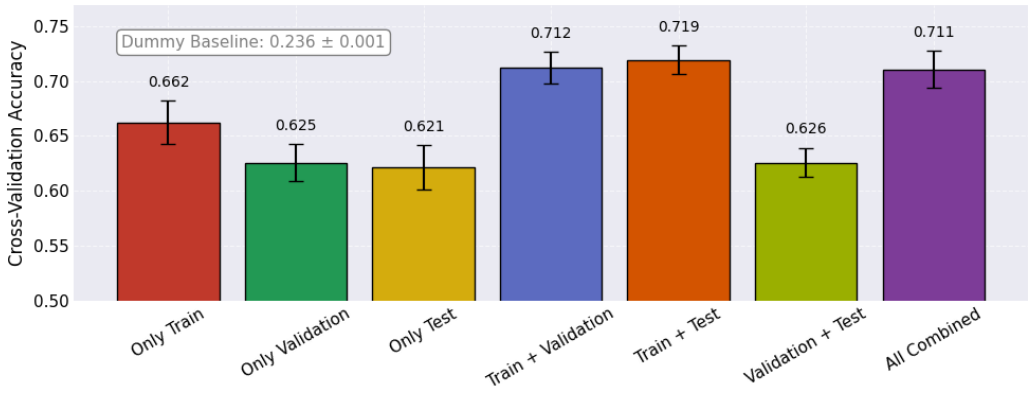

Figure 3: Cross-validation accuracy (starting from 0.50) for each combination of curves. Exact accuracy value written on top of the bars. Accuracy ± standard deviation shown as error on the bars. Baseline score of DummyClassifier indicated on the top left.

enhance performance without introducing redundancy.

5.4 Experiment 4: Model Comparison

The final experiment consists of comparing different time series classification (TSC) models on the task of classifying learners based on the learning curves. This comparison helps identify which model types are most effective at capturing the distinguishing features of learning curves and provides insight into what makes learning curves unique.

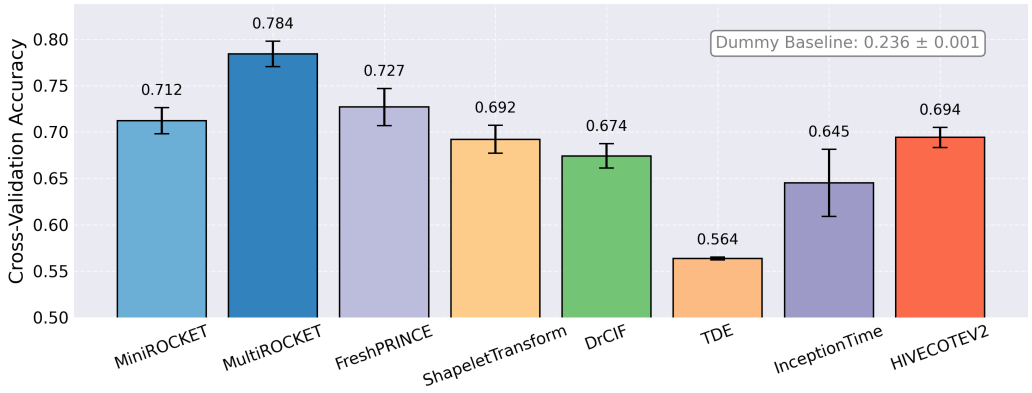

Figure 4: Cross-validation accuracy (starting from 0.50) for each TSC model. Exact accuracy value written on top of the bars. Accuracy ± standard deviation shown as error on the bars. Baseline score of DummyClassifier indicated on the top right.

As shown in Figure 4, the best-performing model is MultiRocket [17], whose superiority is statistically significant based on a paired t-test. Overall, feature-based time series classifiers consistently outperformed other model categories. Notably, both MiniRocket [16]

and FreshPRINCE [12] achieved higher accuracy than HIVE-COTE 2.0 [13], a model widely recognized as state-of-the-art on the UCR benchmark [24]. In contrast, the Temporal Dictionary Ensemble (TDE) [23] performed substantially worse than the other models, suggesting that dictionary-based approaches may be poorly suited for learning curve classification. The weak performance of TDE may have also negatively impacted the ensemble performance of HIVE-COTE 2.0, which incorporates TDE as a component model.

It is worth noting that several models, including Shapelet Transform, DrCIF, TDE, and HIVE-COTE 2.0, were subject to training time limits of 2, 4, 4, and 12 hours respectively. The time limits were put in place in order to maintain practical experiment runtimes. While these constraints may have marginally reduced performance, prior studies indicate that these models typically reach at least 99% of their full potential within such time limits [13]. As such, the relative rankings observed here are unlikely to be significantly impacted by these restrictions.

6 Discussion and Future Work

The results presented in Section 5 offer several insights that contribute to a deeper understanding of the behavior and structure of learning curves across different learners.

First, the pairwise classification experiment and the corresponding results in Figure 1 reveal consistent patterns of similarity and distinction among learners based on their training, validation, and test accuracy curves. Several learner pairs, such as *SVC_poly* and *SVC_rbf*, *ens.ExtraTrees* and *ens.RandomForest*, and all variations of Naive Bayes classifiers exhibit strong structural similarities across all three curve types, making them more difficult to differentiate. In contrast, learners such as *DummyClassifier*, *QDA*, and *SVC_sigmoid* consistently display highly distinctive curve patterns, yielding high classification accuracy in pairwise comparisons. These observations are consistent with findings from the LCDB 1.1 paper [4], which identifies these same learners as being the most "ill-behaved." However, our results suggest that ill-behavior alone does not fully explain similarity or distinctiveness in learning curves. For instance, *DecisionTree*, *ExtraTrees*, and *ens.GradientBoosting* all exhibit low rates of ill-behaved curves (1.5%, 1.9%, and 1.9% respectively for validation accuracy curves), yet only the first two demonstrate high similarity in their validation accuracy curves, indicating that other structural factors are also at play.

Another key observation is the comparatively high distinguishability of training accuracy curves across learners, relative to validation and test accuracy curves. Interestingly, validation and test accuracy curves exhibit highly consistent structural alignment. Learners with similar validation accuracy curves also tend to have similar test accuracy curves, and vice versa. These relationships are further reinforced by the results in Figure 3, which show minimal performance gains when combining validation and test accuracy curves. This suggests that many validation curve-based approaches developed in the context of LCDB [5] and LC-PFN [3] may extend effectively to test accuracy curves with little modification. In contrast, the distinctiveness of training accuracy curves presents a potential opportunity for future work: their incorporation could enhance learner classification or provide complementary insights not captured by validation or test accuracy curves alone.

The experiment in Section 5.2 examined how varying the minimum-length threshold impacts classification performance. As anticipated, longer curves generally lead to higher accuracy, as they provide the model with more informative input. However, setting the threshold too high limits the number of usable samples, which can hurt performance due to insufficient training data. Although the two classifiers used in this experiment showed minor

differences in their behavior, both largely followed this overall trend. The results indicated that a minimum length threshold in the range of 50 to 90 offered the best balance between accuracy and variance.

Finally, in Section 5.4, we compared the performance of various TSC models for classifying learning curves. Among all evaluated methods, *MultiRocket* [17] achieved the highest accuracy, aligning with prior results showing that it is competitive with the state-of-the-art on the UCR archive [24], including *HIVE-COTE 2.0*. Notably, the *FreshPRINCE* [12] classifier also demonstrated surprisingly strong performance. While FreshPRINCE is generally considered a strong but relatively simple feature-based model, its performance on learning curves exceeded that of more complex deep learning models such as *InceptionTime* [19], which typically perform better on standard time series classification tasks [10].

These results suggest that feature-based classifiers, such as FreshPRINCE, DrCIF [13], and MultiRocket (which incorporates first-order derivatives), are particularly effective for learning curve data. This implies that global characteristics of curves (such as trend, slope, and variability), may be more informative for distinguishing learners than localized temporal patterns or precise pointwise differences. In contrast, deep learning models like InceptionTime, which are optimized for capturing small-scale local behavior and temporal information, may be less well-suited to the broader structural nature of learning curves. It is also important to note that, while learning curves can be treated as time series for classification purposes, they lack certain defining characteristics of traditional time series, such as seasonality or periodicity. Furthermore, models that rely on strict temporal alignment (like the distance-based *shapeDTW* [11]) struggle with the inherent noise and variability of learning curves. This supports the conclusion that learning curves are better understood through their high-level structure rather than precise alignment or local behavior.

It is important to acknowledge a limitation in the experiments of Section 5: hyperparameter tuning was not performed exhaustively for the TSC models. Instead, we used general-purpose settings recommended by the model authors. This likely had the greatest impact on more complex models, such as InceptionTime, which are more sensitive to hyperparameters. However, because no model received specialized tuning, the comparisons remain fair. Moving forward, a promising research direction would be the development of deep learning models tailored specifically to learning curves; models that do not rely on time series assumptions, but instead focus on the unique structural characteristics of learning curve data.

Overall, these findings emphasize the nuanced structure of learning curves and the importance of selecting appropriate curve types and lengths when designing models or analyses based on them. They also highlight several promising directions for future work, including more systematic use of training curves and deeper investigation into curve behaviors beyond those captured by traditional metrics of ill-behavior.

7 Conclusions

This work demonstrates that learning curves, commonly used to assess model performance across increasing training set sizes, can reveal additional insights when analyzed through the process of classification. By examining the training, validation, and test accuracy curves available in the LCDB [5], we find that validation and test curves share strong structural similarities, while training curves display distinctly different patterns. This suggests that while validation accuracy has traditionally been the focus of learning curve research, test

accuracy curves can serve a similar role, and incorporating training curves may offer new, complementary perspectives.

A key methodological contribution of this study is the framing of learning curves as time series, allowing us to apply a wide range of time series classification (TSC) methods to distinguish between learners. Among the evaluated TSC approaches, feature-based models performed best, likely because they emphasize global curve characteristics such as trend, variability, and slope. In contrast, deep learning models and distance-based methods, which rely more heavily on temporal alignment or local patterns, were less effective, likely due to the noisy and non-periodic nature of learning curves. These findings highlight the need for future research into models tailored specifically to learning curves, rather than generic time series.

We also explored how the minimum number of curve points affects classification performance. Our results indicate that longer curves provide more discriminative power, particularly after a certain threshold of anchor points is reached. This finding aligns with previous research [5] showing that certain types of ill-behavior, such as peaking, may only emerge later in the learning process.

Finally, our results show that distinguishability is not limited to ill-behavior as a primary differentiator between learners. Even learners with low rates of ill-behaved curves [4] can be effectively distinguished, suggesting that additional, currently underexplored structural features may contribute to differences in learning curves across learners.

8 Responsible Research

This research was carried out in alignment with the TU Delft Code of Conduct for Responsible Research Practices [29]. In particular, it was guided by the university's core values: diversity, integrity, respect, engagement, courage, and trust. The following paragraphs outline the specific measures taken to follow these principles throughout the research process.

Ensuring the reproducibility of experimental results was a key objective throughout this study. To support reproducibility, each experiment was executed multiple times to account for variance and provide statistically robust results. Wherever randomness was involved, a fixed seed was used to guarantee consistency across runs. Finally, all code and implementation details have been made publicly available. The repository containing the code used in this research can be accessed here. For more details regarding the repository, see Appendix B.1.

All external works and sources that informed this research have been appropriately cited. In addition, this thesis makes limited use of generative AI tools to support the formulation and structuring of text. These tools were not used to generate ideas, design experiments, or analyze data. A detailed overview of how generative AI was used can be found in Appendix B.2.

As with any machine learning research, data bias is an important consideration. The experiments in this study are based on data obtained from the Learning Curve Database (LCDB) [5], which provides prediction data from 24 classifiers across 265 datasets. Rather than relying on a single training-test split, the LCDB includes 25 randomized splits at each anchor point. This helps reduce variance and mitigate the effect of noise, contributing to more reliable performance estimates. Moreover, because this work operates on learning curves rather than raw data involving human inputs, it is less susceptible to certain forms of bias typically associated with sensitive or demographic information.

The broader objective of this research, and of ongoing work on LC-PFNs, is to improve learning curve extrapolation. Better extrapolation techniques can support more efficient model selection, early stopping strategies, and better-informed decisions about data acquisition [1]. While increasing the efficiency and accessibility of machine learning has clear benefits (such as reducing computational cost and democratizing access) it also carries ethical considerations. Machine learning has already had a significant impact in fields such as healthcare and education. However, the same advancements can be exploited by malicious parties for harmful purposes, such as generating deepfakes, conducting automated phishing attacks, or reinforcing algorithmic bias in surveillance systems. As such, it is essential that improvements in efficiency are accompanied by continued attention to ethical standards, transparency, and the development of regulations.

References

[1] Felix Mohr and Jan N. van Rijn. Learning curves for decision making in supervised machine learning: a survey. *Machine Learning*, 113(11-12):8371–8425, December 2024.

[2] Tom J. Viering and Marco Loog. The shape of learning curves: a review. *CoRR*, abs/2103.10948, 2021.

[3] Tom Julian Viering, Steven Adriaensen, Herilalaina Rakotoarison, and Frank Hutter. From epoch to sample size: Developing new data-driven priors for learning curve prior-fitted networks. In *AutoML Conference 2024 (Workshop Track)*, 2024.

[4] Cheng Yan, Felix Mohr, and Tom Viering. Lcdb 1.1: A database illustrating learning curves are more ill-behaved than previously thought, 2025.

[5] Felix Mohr, Tom J. Viering, Marco Loog, and Jan N. van Rijn. Lcdb 1.0: An extensive learning curves database for classification tasks. In Massih-Reza Amini, Stéphane Canu, Asja Fischer, Tias Guns, Petra Kralj Novak, and Grigorios Tsoumakas, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2022, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 3–19. Springer, 2023.

[6] Remco R. Bouckaert. Estimating replicability of classifier learning experiments. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 15, New York, NY, USA, 2004. Association for Computing Machinery.

[7] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30, 2006.

[8] Remco R. Bouckaert. Choosing between two learning algorithms based on calibrated tests. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, pages 51–58. AAAI Press, 2003.

[9] Thomas G Dietterich. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1(3):317–328, 1997.

[10] David Guijo-Rubio, Matthew Middlehurst, Guilherme Arcencio, Diego Furtado Silva, and Anthony Bagnall. Unsupervised feature based algorithms for time series extrinsic regression. *Data Mining and Knowledge Discovery*, 38(4):2141–2185, May 2024.

[11] Jiaping Zhao and Laurent Itti. shapedtw: Shape dynamic time warping. *Pattern Recognition*, 74:171–184, 2018.

[12] Matthew Middlehurst and Anthony Bagnall. *The FreshPRINCE: A Simple Transformation Based Pipeline Time Series Classifier*, pages 150–161. Springer International Publishing, 2022.

[13] Matthew Middlehurst, James Large, Michael Flynn, Jason Lines, Aaron Bostrom, and Anthony Bagnall. Hive-cote 2.0: a new meta ensemble for time series classification. *Machine Learning*, 110(11-12):3211–3243, September 2021.

[14] Matthew Middlehurst, James Large, and Anthony Bagnall. The canonical interval forest (cif) classifier for time series classification. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 188–195. IEEE, December 2020.

[15] Angus Dempster, François Petitjean, and Geoffrey I. Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, July 2020.

[16] Angus Dempster, Daniel F. Schmidt, and Geoffrey I. Webb. Minirocket: A very fast (almost) deterministic transform for time series classification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery amp; Data Mining*, KDD â21, pages 248–257. ACM, August 2021.

[17] Chang Wei Tan, Angus Dempster, Christoph Bergmeir, and Geoffrey I. Webb. Multirocket: Multiple pooling operators and transformations for fast and effective time series classification, 2022.

[18] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, March 2019.

[19] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F. Schmidt, Jonathan Weber, Geoffrey I. Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, September 2020.

[20] Markus Löning, Anthony Bagnall, Sajaysurya Ganesh, Viktor Kazakov, Jason Lines, and Franz J. Király. sktime: A unified interface for machine learning with time series, 2019.

[21] Jon Hills, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony Bagnall. Classification of time series by shapelet transformation. *Data Mining and Knowledge Discovery*, 28, 05 2013.

[22] Aaron Bostrom and Anthony Bagnall. *Binary Shapelet Transform for Multiclass Time Series Classification*, pages 24–46. 07 2017.

[23] Matthew Middlehurst, James Large, Gavin Cawley, and Anthony Bagnall. *The Temporal Dictionary Ensemble (TDE) Classifier for Time Series Classification*, pages 660–676. Springer International Publishing, 2021.

[24] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive, 2019.

[25] Wanpracha Artit Chaovalitwongse, Oleg A. Prokopyev, and Panos M. Pardalos. Electroencephalogram (eeg) time series classification: Applications in epilepsy. *Annals of Operations Research*, 148:227–250, 2006.

[26] Monica Arul and Ahsan Kareem. Applications of shapelet transform to time series classification of earthquake, wind and wave data. *Engineering Structures*, 228:111564, February 2021.

[27] Merriam-Webster. Time series, 2025. Retrieved May 7, 2025.

[28] Chang Wei Tan, François Petitjean, Eamonn Keogh, and Geoffrey I. Webb. Time series classification for varying length series, 2019.

[29] Sabine Roeser and Samantha Copeland. TU Delft Code of Conduct: Why What Who How, 2020. Final published version.

A Experiments

This section holds additional figures and information regarding the experiments of this study.

A.1 Model Hyper-Parameters

Table 1: Hyper-parameters for each model used in the experiments (as implemented in `sktime`)

Model	**Hyper-Parameters (defaults)**
FreshPRINCE	`n_estimators=200`, `random_state=42`, `default_fc_parameters=comprehensive`
DrCIF	`n_estimators=500`, `min_interval=4`, `n_intervals=None`, `base_estimator=CIT`, `random_state=42`, `time_limit_in_minutes=240`
MiniRocket	`num_kernels=10000`, `max_dilations_per_kernel=32`, `random_state=42`
MultiRocket	`num_kernels=10000`, `max_dilations_per_kernel=32`, `n_features_per_kernel=4`, `random_state=42`
Arsenal	`num_kernels=2000`, `n_estimators=25`, `rocket_transform='rocket'`
InceptionTime	`n_epochs=1500`, `batch_size=64`, `n_filters=32`, `use_residual=True`, `use_bottleneck=True`, `bottleneck_size=32`, `depth=6`, `kernel_size=40`, `random_state=42`
ShapeletTransformClassifier	`n_shapelet_samples=10000`, `batch_size=100`, `random_state=42`, `transform_limit_in_minutes=120`
TDE	`n_parameter_samples=250`, `max_ensemble_size=50`, `max_win_len_prop=1`, `min_window=10`, `randomly_selected_params=50`, `dim_threshold=0.85`, `max_dims=20`, `random_state=42`, `time_limit_in_minutes=240`
HIVE-COTE	Includes components: TDE, DrCIF, ShapeletTransform, Arsenal Each component uses its own default hyper-parameters as above `random_state=42`, `time_limit_in_minutes=720`

A.2 Experiment 1: Test Accuracy Pairwise Matrix

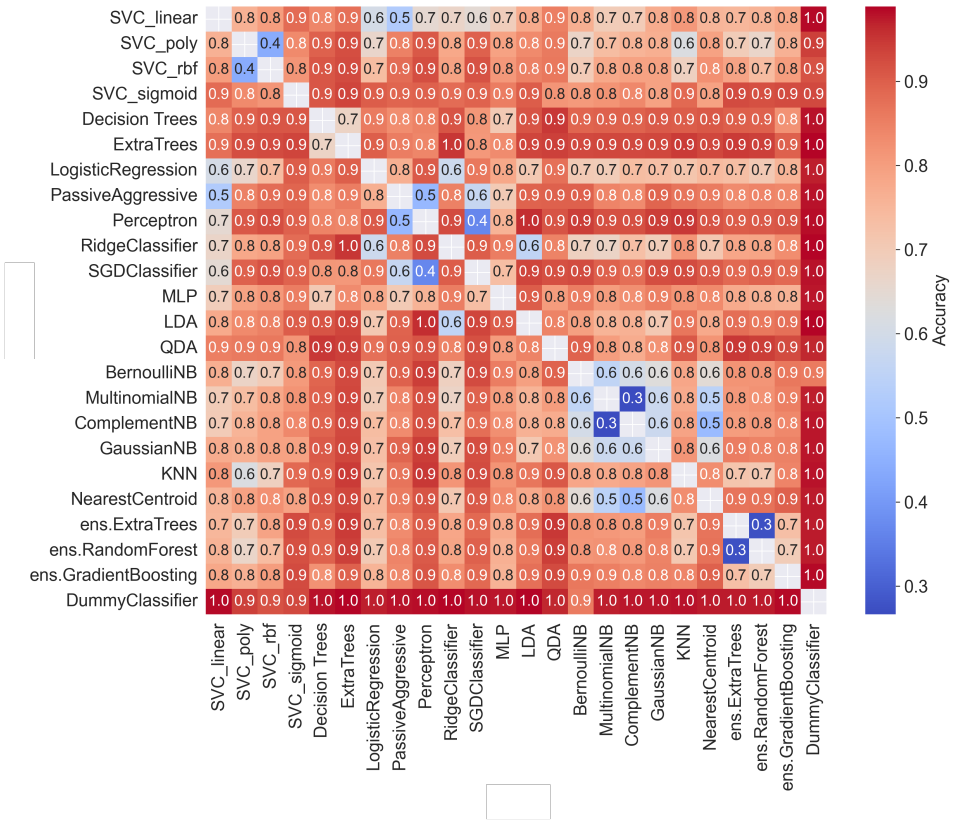

Figure 5: Pairwise binary classification accuracy matrix for test accuracy curves. Exact accuracy for the binary classifier trained on learner A and learner B is written on their intersection.

A.3 Group Formulation: in Detail

For experiments 2–4, we simplified the classification task by grouping the 24 learners into six distinct categories. These groups were constructed by considering similarities in validation accuracy curves, theoretical relationships among models, and trying to minimize class imbalance across groups.

Group 1 was defined primarily based on empirical similarities in their validation accuracy curves, as seen in Figure 1. These similarities were somewhat unexpected from a theoretical perspective but were consistent across multiple runs.

Group 2 includes Decision Tree and ExtraTree, which are closely related both theoretically and empirically. MLP was added to this group primarily to reduce label imbalance, though it also shares some surface-level similarity in performance patterns with the decision tree-based models.

Group 3 was formed based on both theoretical and empirical similarities. The support vector classifiers (SVC_poly and SVC_rbf) naturally fit together, while the ensemble methods (ens.ExtraTrees and ens.RandomForest) are structurally similar and commonly used for tabular data. Interestingly, KNN also exhibited similar validation accuracy curves and was thus included in this group.

Group 4 contains all variants of the Naive Bayes classifier, which consistently showed similar validation performance. NearestCentroid was grouped here due to its close resemblance in behavior to the Naive Bayes models, despite its differing underlying assumptions.

Group 5 includes RidgeClassifier, LogisticRegression, and LDA, all of which are linear models that operate similarly in terms of classification approach and yield comparable learning dynamics.

Group 6 consists of outlier models whose learning curves were clearly distinguishable from all other learners. While there is no strong theoretical link among the models in this group, they were grouped together due to their shared dissimilarity with all other groups.

B Responsible Research Considerations

This section contains additional information regarding the responsible research considerations of this work.

B.1 Reproducibility and the Codebase

All code necessary to reproduce the experiments in this study is publicly available at: `https://github.com/sinanbasaran/cse-3000`. Each of the four main experiments is implemented in a dedicated Jupyter notebook:

- `experiment-pairwise-classification.ipynb` - implements Experiment 1 (Pairwise Binary Classifiers)
- `experiment-different-lengths.ipynb` - implements Experiment 2 (Different Minimum Length Cutoffs)
- `experiment-combined-curves.ipynb` - implements Experiment 3 (Combined Curve Types)
- `experiment-model-comparisons.ipynb` - implements Experiment 4 (Model Comparisons)

To ensure full transparency and reproducibility:

- All figures used in the paper are available in the `plots` directory.
- All classification scores, metrics, and experiment outputs are stored in the `scores` directory.
- Trained models for all experiments are saved in the `trained_models` directory.

The only exception is HIVE-COTE 2.0, which is excluded from the `trained_models` directory due to its size (approximately 14 GB per fold, across 5 folds). All other models are available and can be used directly to replicate our results.

B.2 Usage of Generative AI

The following table outlines the use of generative AI tools during the writing of this paper:

Task	**Yes**	**No**
Rewriting and polishing paragraphs	✓	
Help with LaTeX formatting (e.g., clickable references, figures, tables)	✓	
Designing experiment structure		✓
Implementing the models or running the experiments		✓
Formulating research question or hypothesis		✓
Analyzing results or drawing conclusions		✓
Writing any section from scratch		✓

Table 2: Overview of generative AI usage during the writing of this paper.

For further clarification, the writing process was conducted as follows:

- Each section was initially written independently by the author.
- Generative AI was then used selectively to improve phrasing and polish specific paragraphs.
- The resulting output was subsequently **carefully reviewed** and **revised as needed** to ensure consistency with the author's writing style, intent, and the overall content of the paper.

An example prompt to rewrite a section is as follows:

> "Rewrite the following paragraph on related work, do not add additional information, and make sure to use simple language, similar to the one already used. <section>"