A Unified Neural–Symbolic Framework for Tabular Medical Diagnosis. A Case Study on Representation Learning, Evolutionary Optimization, and Hybrid Reasoning on Parkinson's Disease

Marcin Basisty*
*Faculty of Physics & Applied Computer Science, AGH University of Krakow, Poland
Email: mbasisty@student.agh.edu.pl

***Abstract*—Accurate medical diagnosis is critical for timely intervention, yet complex biomedical datasets pose significant challenges for traditional machine learning models. This work proposes a unified neural-symbolic framework for tabular medical diagnosis, integrating supervised learning, dimensionality reduction, hyperparameter optimization, and hybrid reasoning techniques. We evaluate a baseline Multi-Layer Perceptron (MLP) classifier on the Parkinson's Disease dataset and examine the impact of feature extraction methods, including Principal Component Analysis (PCA), Autoencoders, and K-Means clustering. Various optimizers (Adam, SGD, RMSprop, etc) and hyperparameter search strategies (Grid, Random, Bayesian, Hyperband, and Evolutionary algorithms) are compared to identify optimal training configurations. Finally, we incorporate Bayesian inference, fuzzy logic, and rule-based reasoning to refine decision-making and enhance interpretability. Results demonstrate that combining neural networks with symbolic reasoning and carefully tuned hyperparameters improves classification accuracy, robustness, and reliability, highlighting the potential of hybrid approaches for real-world medical diagnostic tasks.**

***Index Terms*—Neural-symbolic AI, Tabular Data, Medical Diagnosis, Autoencoders, Evolutionary Computing, Bayesian Reasoning, Fuzzy Logic.**

I. Introduction

Artificial intelligence (AI) and machine learning (ML) have become central to modern medical diagnostics, offering automated tools for early detection, risk assessment, and clinical decision support in medical studies. The increasing availability of biomedical datasets—ranging from imaging and signals to tabular clinical records—has enabled data-driven diagnostic models with the potential to surpass traditional heuristic approaches. However, despite significant progress in deep learning and representation learning, the deployment of ML systems in medical contexts remains challenging due to issues such as data heterogeneity, limited interpretability, and sensitivity to hyperparameter choices. These limitations are particularly pronounced for tabular biomedical datasets, which often exhibit high dimensionality, small sample sizes, nonlinear feature interactions, and noise resulting from complex physiological processes.

Parkinson's disease (PD) is a progressive neurodegenerative disorder characterized by motor dysfunction, speech impairment, and a decline in cognitive and autonomic functions. Early diagnosis is essential for improving patient outcomes, yet clinical assessment is typically reliant on expert evaluation and subjective rating scales such as the Unified Parkinson's Disease Rating Scale (UPDRS). As a result, diagnostic accuracy may vary across practitioners and clinical environments. Machine learning offers the potential to assist clinicians by uncovering subtle patterns in biomedical indicators—such as vocal metrics and motor features—that are difficult to detect through conventional clinical examination. The Parkinson's Disease dataset used in this study provides an opportunity to explore how modern ML techniques can contribute to robust and objective diagnostic support.

While traditional supervised learning models have demonstrated effectiveness on structured medical data, they often lack mechanisms for knowledge integration, logical consistency, and human-understandable reasoning. Neural-symbolic AI—a paradigm that combines the statistical power of neural networks with the interpretability and structured knowledge of symbolic reasoning—has emerged as a promising solution to these challenges. Hybrid approaches allow models not only to learn from data, but also to incorporate explicit domain knowledge, uncertainty reasoning, and interpretable rule-based explanations, which are essential for clinical adoption and regulatory compliance.

In this work, we introduce a unified neural–symbolic framework for tabular medical diagnosis, using Parkinson's disease classification as a representative case study. Rather than relying solely on conventional neural models, our approach jointly examines representation learning, model optimization, and symbolic reasoning to understand how each component contributes to diagnostic performance.

Beyond assessing predictive accuracy, the proposed framework aims to improve the transparency and reliability of diagnostic decisions. To this end, symbolic reasoning elements—including probabilistic inference, fuzzy decision processes, and rule-based explanations—are incorporated to complement the learned neural representations. These components provide uncertainty-aware predictions and interpretable logic structures that are better aligned with clinical expectations.

Overall, this study provides a comprehensive foundation for building hybrid neural–symbolic diagnostic systems. By integrating learning-based models with transparent reasoning mechanisms, the framework seeks to advance the development of accurate, robust, and clinically interpretable AI tools for medical decision support.

II. Methodology Overview

In this work, we investigate how the combination of neural, statistical, and symbolic techniques can improve diagnostic performance on biomedical tabular data. Our approach consists of four core components.

1) Supervised Learning Baseline: We train a standard Multi-Layer Perceptron (MLP) classifier to establish baseline performance for Parkinson's disease detection using acoustic voice features.

2) Dimensionality Reduction and Representation Learning: We evaluate Principal Component Analysis (PCA), Autoencoders, and K-Means–derived features to assess how different low-dimensional representations affect classification accuracy and model stability.

3) Hyperparameter Exploration: We benchmarked and investigated how hyperparameter optimization algorithms—such as Grid Search, Random Search, Bayesian optimization (TPE), Hyperband, and Evolutionary strategies like NSGAII—can enhance the learning process. This component focuses on identifying the key improvements these optimization methods can provide in discovering effective training configurations for the MLP model.

4) Extensive Reduction Technique $\times$ Optimizer $\times$ Hyperparameter Study: To systematically evaluate model performance, we conduct a comprehensive experiment combining dimensionality-reduction techniques, optimizers, and hyperparameter algorithms. Every configuration across this three-dimensional search space is trained and assessed, enabling us to identify the best-performing setups and analyze interesting behaviors that emerge from interactions between feature representations, optimization strategies, and hyperparameter selection.

5) Neural–Symbolic Integration: We augment the learned model with a fuzzy-logic layer, a small rule-based component, and a Bayesian posterior adjustment step. This hybrid reasoning mechanism allows us to evaluate whether such additional symbolic layers enhance or hinder the model when processing medical diagnostic data.

Together, these components enable a systematic evaluation of whether hybrid neural–symbolic modeling, combined with optimized training and effective data representation, yields a more accurate and robust diagnostic framework.

III. Dataset and Preprocessing

The `Parkinson_Disease` dataset was obtained from the `ucimlrepo` package [2]. This dataset comprises biomedical voice measurements from 31 individuals, including 23 with Parkinson's disease (PD) and 8 healthy controls. The dataset contains 195 voice recordings, with each row representing a recording and each column corresponding to a specific voice feature. The target variable "status" indicates disease presence (1 for PD, 0 for healthy).

TABLE I: UCI Parkinsons Dataset — Overview

Attribute	Value
Dataset Name	Parkinsons
Repository	UCI Machine Learning Repository
Donated On	June 25, 2008
Instances	195 voice recordings from 31 individuals
Number of Features (excluding ID)	22 real-valued features
Target Variable	status (0 = healthy, 1 = Parkinson's)
Feature Type	Continuous (real)
Task	Binary Classification
Missing Values	None

The preprocessing involved several systematic steps to prepare the data for modeling:

1) **Missing Value Analysis**: Initial inspection confirmed no missing values in the dataset, simplifying the preprocessing pipeline.
2) **Normalization**: All numerical features were standardized using z-score normalization to ensure equal contribution during model training. This transformation is visualized in Fig. 4, which shows the distribution of normalized features.
3) **Exploratory Analysis**: Fig. 2 illustrates pairwise relationships between selected features, revealing complex nonlinear patterns. The correlation heatmap in Fig. 3 demonstrates moderate to high correlations among several voice features, justifying dimensionality reduction techniques.
4) **Data Splitting**: The dataset was partitioned using an 80:20 train-test split with stratification (`stratify=y`) to preserve class distribution proportions in both sets. This ensures representative sampling of minority and majority classes.

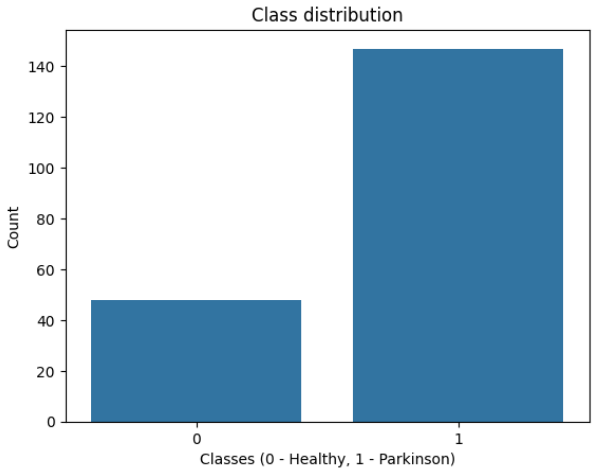


Fig. 1: Class distribution showing imbalance between Parkinson's disease (1) and healthy (0) cases. The dataset contains approximately 75% PD cases, necessitating stratification during data splitting.

The preprocessing pipeline ensures the data is properly scaled and partitioned while maintaining the statistical properties necessary for robust model evaluation. The absence of missing values eliminates the need for imputation techniques, reducing potential bias in the modeling process.

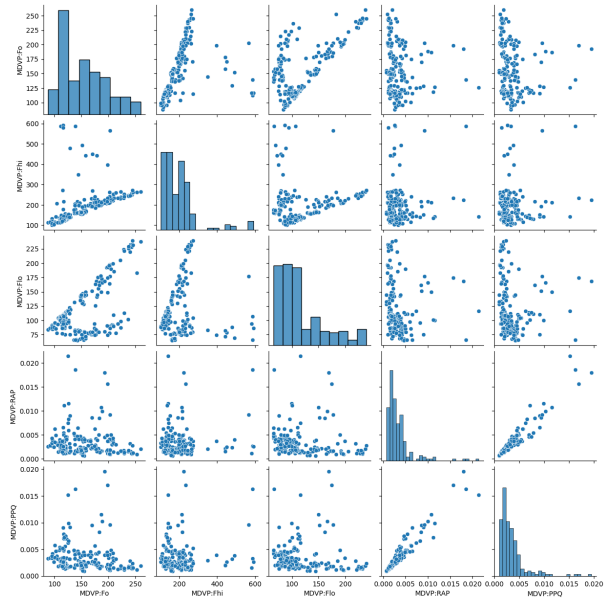


Fig. 2: Pairwise feature relationships reveal complex patterns and potential nonlinear dependencies among voice measurements. These visualizations inform feature engineering and model selection decisions.

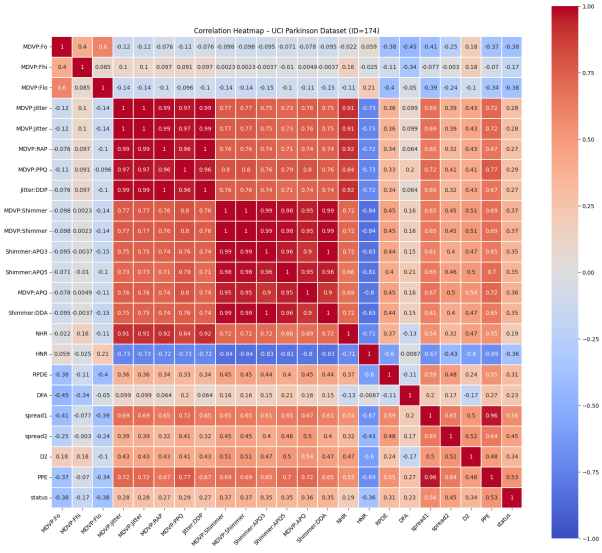


Fig. 3: Correlation matrix of voice features showing moderate to high inter-feature dependencies. These correlations justify dimensionality reduction techniques like PCA to mitigate multicollinearity issues.

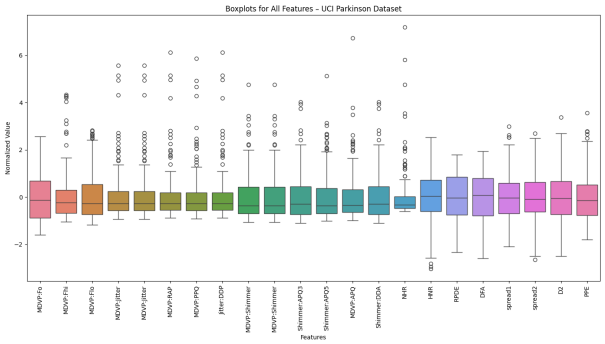


Fig. 4: Normalized feature distributions showing reduced variance after z-score standardization. This normalization ensures features contribute equally to model training and improves optimization convergence.

IV. Supervised Learning - Baseline

The implementation of a 22-16-1 MLP (ReLU-Sigmoid, BCE Loss (Binary Cross Entropy)) baseline model trained with the `Adam` optimizer yielded the following performance compared to the **Gradient Purist*** final proposed model from the this study and the other baseline models from `scikit-learn` [13]:

TABLE II: Comparison with baselines (mean ± std)

Model	Acc	F1	AUC	MCC
Logit Regression	0.8974 ± 0.003	0.9355 ± 0.001	0.8974 ± 0.004	0.7261
SVM (RBF)	0.8718 ± 0.012	0.9206 ± 0.002	0.7276 ± 0.013	0.6530
Random Forest	0.9231 ± 0.009	0.9508 ± 0.011	0.9149 ± 0.034	0.7965
XGBoost	0.8974 ± 0.023	0.9355 ± 0.011	0.9414 ± 0.028	0.7261
MLP	0.9103 ± 0.019	0.9411 ± 0.012	0.9103 ± 0.013	0.7593
Gradient Purist*	**0.9654 ± 0.0147**	**0.9794 ± 0.0085**	**0.9518 ± 0.048**	**0.8793**

* **Gradient Purist**: A Strictly Gradient-Centric Hybrid MLP (PCA + RMSprop + Hyperband + Hybrid)

Based on the results reported in Tab. II, the proposed hybrid approach consistently outperforms all baseline models across all evaluated metrics. In particular, the proposed model achieves the highest mean accuracy (0.9654), F1-score (0.9794), and MCC (0.8793), indicating superior overall classification performance and a strong correlation between predicted and true class labels. The improvement in MCC is especially noteworthy, as this metric accounts for class imbalance and reflects robust decision-making beyond simple accuracy gains.

Compared to traditional machine learning classifiers such as Logistic Regression, SVM (RBF), Random Forest, and XGBoost, the proposed method demonstrates clear gains in both discriminative ability and predictive reliability. While Random Forest and XGBoost perform competitively in terms of F1-score and AUC, their MCC values remain substantially lower than that of the proposed model, suggesting weaker consistency in handling false positives and false negatives.

Overall, these results confirm that the proposed hybrid framework effectively leverages both data-driven learning and domain-informed reasoning, leading to more reliable and accurate predictions than conventional standalone models.

V. Unsupervised Learning

We compare three dimensionality-reduction techniques:

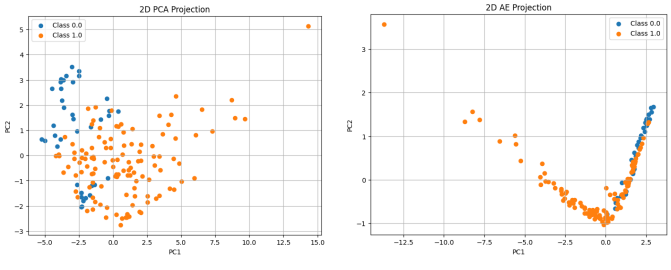
- K-means clustering [9]: distances to K centroids,
- PCA [7]: top components explaining $\geq$95% variance,
- Autoencoder [8]: 30–16–8–16–30 structure; 8-D bottleneck

After applying the dimensionality reduction, the models were evaluated using the same hyperparameters: `opt_name = "Adam"`, `lr = 0.01`, `epochs = 100`.

Method	Acc	Prec	Rec	F1	AUC
KMeans	0.8205	0.8055	**1.0**	0.8923	0.8138
PCA	**0.8718**	**0.875**	0.9655	**0.9180**	**0.8897**
Autoencoder	0.7692	0.7778	0.9655	0.8615	0.8759

TABLE III: Performance metrics after employing each dimensionality reduction technique.

Comparing the performance metrics of the different dimensionality reduction techniques, it is evident that PCA outperforms both K-means and Autoencoder in most measures, achieving the highest accuracy (0.8718), precision (0.875), F1 score (0.9180), and AUC (0.8897). Despite these improvements over the other reduction methods, PCA still falls slightly short of the raw baseline performance, indicating that while it preserves most of the relevant information, some predictive capability is lost during the dimensionality reduction process. KMeans, on the other hand, shows a perfect recall of 1.0, suggesting it identifies all positive instances, but its lower precision (0.8055) and F1 score (0.8923) indicate a higher rate of false positives, which may reduce its practical utility. The Autoencoder performs the worst among the three techniques, with lower accuracy (0.7692) and F1 score (0.8615), though its recall is relatively high (0.9655), indicating it captures most true positives but at the cost of overall classification performance.



(a) PCA projection plot (b) AE projection plot

Fig. 5: Projection plots: PCA vs AE

Looking at the plots, we can see that when it comes to grouping the data, the Autoencoder outperforms PCA. The representations produced by the Autoencoder are denser, and the class distributions are more distinguishable, but when looking closer we can see overlapping structures, which make it harder to identify clusters or patterns within the data. In contrast, PCA produces a more spread-out representation, which can obscure class boundaries and make the underlying structure less apparent. This suggests that while PCA may perform well in terms of predictive metrics, Autoencoders are more effective at capturing the intrinsic structure of the data for visualization and clustering purposes.

VI. Testing Optimizers

To better understand the learning behavior of the baseline MLP, we evaluated several optimization algorithms, including Adam, SGD, RMSprop, and others. By comparing their performance using F1-Score, area under the curve (AUC), and the Matthews correlation coefficient (MCC), we observed how differences in their update strategies can either enhance or limit the model's ability to make accurate predictions. This analysis highlights the impact of optimizer choice on both the model's predictive performance and its generalization capabilities across the dataset.

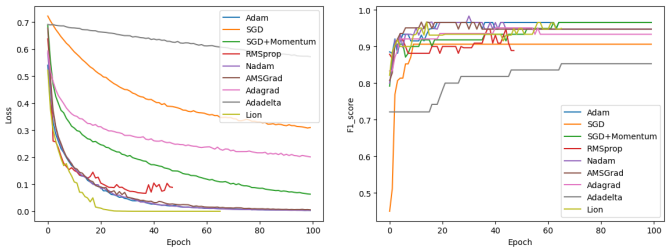


Fig. 6: Training dynamics of the baseline MLP using different optimizers: (a) loss convergence over epochs and (b) corresponding F1-score progression.

Optimizer	ACC	PREC	REC	F1	AUC	MCC	Time
Adam	0.9487	0.9655	0.9655	0.9655	0.9655	0.8655	3.181
SGD	0.8462	0.8286	1.0000	0.9062	0.8828	0.5757	2.388
SGD+Momentum	0.9487	0.9655	0.9655	0.9655	0.9759	0.8655	3.019
RMSprop	0.8462	0.9600	0.8276	0.8889	0.9466	0.6623	0.891
Nadam	0.9231	0.9643	0.9310	0.9474	0.9793	0.8064	1.806
AMSGrad	0.9231	0.9643	0.9310	0.9474	0.9793	0.8064	1.790
Adagrad	0.8974	0.9032	0.9655	0.9333	0.9414	0.7197	1.599
Adadelta	0.7436	0.7436	1.0000	0.8529	0.5241	0.0000	1.794
Lion	0.9231	0.9333	0.9655	0.9492	0.9862	0.7934	1.034

TABLE IV: Performance comparison of optimizers for the baseline MLP.

As shown in Tab. IV and Fig. 6, RMSprop achieved an accuracy of 0.8462, F1-Score of 0.8889, AUC of 0.9466, and Matthews correlation coefficient (MCC) of 0.6623, demonstrating fast early convergence and solid predictive performance. Adam and SGD+Momentum both reached higher accuracy and F1-Score values of 0.9487 and 0.9655, respectively, with MCC of 0.8655, though their convergence was slightly slower compared to RMSprop. In contrast, SGD alone converged more slowly (accuracy 0.8462, F1-Score 0.9062, MCC 0.5757), and Adadelta exhibited particularly poor performance, with an MCC of 0.0000 despite achieving an F1-Score of 0.8529, indicating that its predictions were essentially uncorrelated with the true labels. These results

highlight that while RMSprop offers rapid early learning, Adam and SGD+Momentum provide superior overall predictive performance, and optimizers such as Adadelta can fail to capture meaningful decision boundaries in this baseline MLP model.

VII. Hyperparameter Optimization

In this work, all hyperparameter optimization experiments were implemented using the **Optuna** framework [3]. Optuna's built-in samplers were used to construct the optimization studies and explore the parameter space efficiently. Specifically, the NSGAIISampler (Nondominated Sorting Genetic Algorithm II Sampler) [12] was employed for the Evolutionary search strategy, the TPESampler (Tree-structured Parzen Estimator Sampler) for Bayesian optimization, and the remaining built-in samplers were used to implement the Random search [10], Grid search, and Hyperband procedures. To fully exploit the search space related to momentum parameters, the RMSprop optimizer was selected during hyperparameter tuning, ensuring that the sampled momentum values had a direct and meaningful impact on the optimization dynamics. This setup ensures consistent experiment management across all methods while leveraging Optuna's optimized sampling routines for each algorithmic family.

For the hyperparameters we had a given searching space:

$$\Lambda = \{\eta, \mu, h, \lambda, \delta\} \tag{1}$$

- η: learning rate
- μ: momentum
- h: hidden_layer size
- λ: weight_decay (regularization strength)
- δ: batch_size, mini-batch size used during training

with ranges:

$$\begin{aligned} &\eta \in [10^{-4}, 10^{-1}], \quad \mu \in [0.7, 0.99], \quad h \in \{8, 16, 32\} \\ &\lambda \in [0, 0.01], \quad \delta \in \{16, 32, 64\} \end{aligned} \tag{2}$$

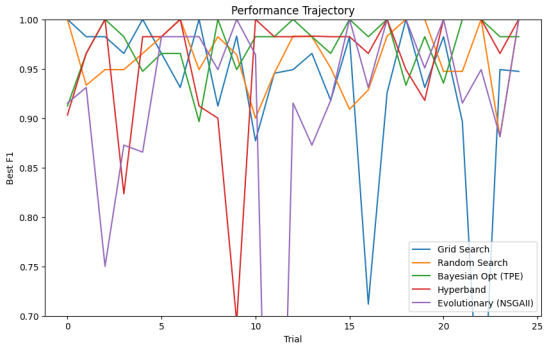
Method	F1	Acc	Prec	Rec	AUC	Epochs	Rank
Grid	0.967	0.949	0.936	1.000	0.983	26	**1**
Random	0.967	0.949	0.936	1.000	0.976	99	4
TPE	0.967	0.949	0.936	1.000	0.969	37	2
Hyperband	0.967	0.949	0.936	1.000	0.979	100	3
NSGAII	0.949	0.923	0.933	0.967	0.976	100	5

TABLE V: Performance metrics, training epochs, and F-ranks for each hyperparameter optimization method.

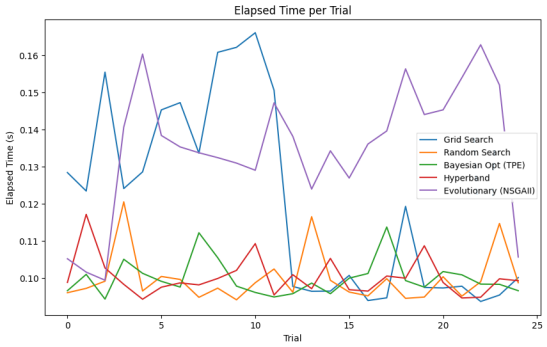
Method	η	μ	h	λ	δ
Grid	0.01	0.99	32	0.00178	32
Random	0.00676	0.9276	32	0.00227	16
TPE	0.04711	0.9518	32	0.000127	16
Hyperband	0.01078	0.7010	32	0.000898	32
NSGAII	0.02903	0.9765	16	0.00872	16

TABLE VI: Best hyper-parameters found by each method.

Tab. V and Tab. VI summarize the performance and resulting hyper-parameters for each optimization method. All algorithms except NSGAII achieved identical top **F1** scores of 0.9667, with Grid and Hyperband showing slightly higher **AUC** values, indicating more robust classification. Grid also achieved the fastest convergence, completing in 26 **Epochs**, while Hyperband and NSGAII required the maximum amount of **Epochs**. TPE had moderate speed but still maintained excellent metrics.



(a)



(b)

Fig. 7: Performance trajectories: (a) best F1 score vs. trial, (b) training time vs. trial for each tuning algorithm.

Looking at the best hyper-parameters, learning rates (η) and momentum (μ) vary considerably across the algorithms, reflecting the flexibility of different search strategies. Grid and Hyperband favored a higher momentum, while Random Search and NSGAII selected lower values. The same pattern could be see in the batch_sizes were mostly consistent, except for Random Search and NSGAII, which chose smaller batches, likely influencing training stability and convergence speed. Overall, Hyperband provides the best trade-off between performance metrics and speed, Grid performs almost as well with slightly slower training, and Evolutionary prioritizes exploration at the cost of slightly lower **F1**, **Accuracy**, and longer training times.

Fig. 7(a) illustrates how the best F1 score evolves over successive trials for each hyperparameter optimization algorithm. Bayesian optimization and Hyperband show the fastest improvement, reaching near-optimal F1 scores within only a few evaluations. Grid search improves more slowly but steadily, reflecting its systematic traversal of the search space. Random search shows larger variance between early trials but eventually converges to a high-performing configuration.

NSGAII search exhibits the most gradual progression, often requiring more trials before achieving competitive F1 values. Overall, Hyperband and TPE optimization demonstrate the most efficient exploration, achieving high F1 scores with fewer evaluations.

Fig. 7(b) compares the computational cost per trial across algorithms. Hyperband consistently maintains the lowest runtime due to its adaptive elimination of weak configurations early in the process. Grid search remains predictable but moderately expensive, while Random Search and TPE approaches show more variability in runtime, depending on sampled hyperparameters and early-stopping behavior. NSGAII search incurs the highest computational cost per trial because each generation depends on evaluating multiple candidate solutions. Taken together, the figures confirm that Hyperband provides the best balance between trial efficiency and computational cost, whereas TPE optimization excels in rapid performance gains with slightly higher time per trial.

VIII. Extensive Reduction Technique × Optimizer × Hyperparameter Study

Based on the hyperparameter optimization experiments across different feature reduction methods and optimizers, several key trends can be observed.

Optimizer performance: Adaptive optimizers such as **Adam, Amsgrad, Nadam,** and **Lion** consistently achieved top performance, often reaching perfect scores (Accuracy, Precision, Recall, F1, AUC = 1.0) across most feature types and hyperparameter (HP) tuning algorithms. **SGD** generally underperformed, achieving lower accuracy unless carefully tuned with Grid or NSGAII algorithm. SGD+Momentum performed comparably to Adam in some cases, but exhibited slightly higher variability in run-time.

Hyperparameter tuning methods: Hyperband, NSGAII, and **Grid search** frequently identified high-performing configurations. Random search was less consistent, particularly with SGD.

Feature reduction methods:

- **Raw features:** Provided the highest performance across all metrics with adaptive optimizers, though at higher computational cost due to dimensionality. SGD benefited significantly from careful tuning.
- **K-means reduction:** Generally lowered model performance (Accuracy 0.871–0.974), making HP tuning more critical. Adaptive optimizers still outperformed SGD, while run-times were shorter due to reduced complexity.
- **PCA:** Preserved most information, achieving near-perfect performance with adaptive optimizers. Hyperband or Grid search consistently provided reliable results with moderate run-time.
- **Autoencoder:** Similar to PCA, allowing near-perfect performance with adaptive optimizers and various HP algorithms. SGD performance remained sensitive to HP tuning.

Overall recommendations: For stable and high-performing models, a combination of an **adaptive optimizer** (preferably Adam or RMSprop), **Hyperband or TPE tuning**, and **PCA or Autoencoder feature reduction** is recommended. Raw features can achieve slightly higher accuracy but at higher computational cost, while K-means features should generally be avoided unless efficiency is critical.

Key insight: Adaptive optimizers are robust across feature types, HP tuning methods, and reduction strategies, while SGD requires careful tuning to approach similar performance. Hyperband offers a strong balance between efficiency and effectiveness, particularly with PCA or Autoencoder features.

IX. Bayesian, Fuzzy, and Rule-Based Enhancements for Decision Making

To improve interpretability and clinical reliability, the neural classifier is augmented with three complementary reasoning mechanisms: Bayesian inference, fuzzy logic, and symbolic rule-based validation. These additions provide structured uncertainty handling, smooth thresholding behavior, and explicit incorporation of clinical priors into the decision-making pipeline.

- **Bayesian Inference:** Prior disease prevalence is integrated with model-derived likelihood estimates to compute posterior probabilities. This enables explicit uncertainty quantification and facilitates risk-aware decision thresholds in clinically ambiguous cases.
- **Fuzzy Logic:** Clinically relevant acoustic thresholds are modeled using fuzzy membership functions, allowing continuous transitions between low-risk and high-risk states. This reduces brittle decision boundaries and accommodates the inherent gradience of vocal impairments.
- **Rule-Based Reasoning:** Expert-informed symbolic rules are applied to override or reinforce model predictions when critical pathological patterns occur. These rules guarantee alignment with established clinical heuristics and safeguard against implausible predictions.

The resulting hybrid architecture leverages both statistical learning and interpretable symbolic reasoning. Empirically, integrating Bayesian and fuzzy components increases confidence calibration and reduces false positives, while the rule-based checks ensure consistency with clinically recognized diagnostic indicators.

X. Fuzzy Inference, Rule-Based Overrides, and Bayesian Posterior Refinement

This work augments a neural classifier with three additional reasoning layers: (i) fuzzy membership scoring derived from salient acoustic features, (ii) deterministic rule-based overrides for critical pathological patterns, and (iii) Bayesian posterior refinement using prior disease prevalence. The resulting hybrid probability provides a more interpretable and robust decision mechanism for Parkinson's disease detection.

A. Fuzzy Reasoning Layer

Two features were selected due to their clinical relevance in phonatory abnormality detection: the mean fundamental frequency (MDVP:Fo(Hz)) and local shimmer variation

(MDVP:Shimmer). After standard scaling, these features are centered around zero, making sigmoidal membership functions appropriate for quantifying high-risk linguistic patterns.

Let $f_1 = x_1$ denote the scaled mean fundamental frequency and $f_2 = x_2$ denote the scaled shimmer. The fuzzy membership functions for high-risk behavior are defined as:

$$\mu_{\text{high}}(f) = \frac{1}{1 + \exp[-\alpha\,(f - \beta)]}, \tag{3}$$

where (α, β) are scale-adjusted slope and shift parameters. These parameters were optimized using the **Optuna** framework to maximize the classifier performance (e.g., AUC).

The fuzzy risk score is computed as a convex combination:

$$R = \rho \cdot \mu_{f_1} + (1 - \rho) \cdot \mu_{f_2}, \tag{4}$$

with $R \in [0, 1]$ enforced via clipping.

B. Soft Rule-Based Override Layer

Certain extreme pathological vocal patterns warrant intervention beyond fuzzy scoring. Instead of hard-coded probabilities, we define a **soft rule-based override**:

$$p_{\text{rule}} = \text{soft\_override}(p_{\text{hybrid}}, X; k, base, scale) \tag{5}$$

where the update is based on a sigmoid mapping of feature extremeness:

- k: slope of the sigmoid (controls how sharply probability increases for extreme features)
- $base$: minimum probability for borderline cases
- $scale$: maximum probability increment for extreme samples

The updated probabilities are constrained to $[0, 1]$, and all parameters were tuned via Optuna.

C. Bayesian Posterior Refinement

The hybrid probability p_{rule} is further updated using Bayesian posterior refinement:

$$p_{\text{post}} = \frac{p_{\text{rule}} \cdot P(D)}{p_{\text{rule}} \cdot P(D) + (1 - p_{\text{rule}}) \cdot (1 - P(D)) + \varepsilon}, \tag{6}$$

where $P(D)$ is the prior disease prevalence, and ε is a small stabilizer. This posterior probability p_{post} is used as the final classification probability.

XI. Final Conclusion

Developing reliable machine learning systems for medical diagnosis demands not only strong predictive performance but also careful evaluation and a deep understanding of how different modeling choices influence outcomes. In this work, we explored how feature representation, optimization strategy, and hyperparameter selection shape the behavior of an MLP classifier applied to the Parkinson Disease dataset. Our findings illustrate that achieving high performance in medical settings is rarely the result of a single technique; instead, it emerges from a deliberate combination of representation learning, optimizer efficiency, and systematic parameter exploration.

The baseline MLP demonstrated solid diagnostic capability, yet the experiments revealed how much additional performance can be unlocked through thoughtful preprocessing and dimensionality reduction. PCA and K-means offered moderate benefits, while Autoencoder consistently stood out by producing more expressive and stable representations, ultimately leading to stronger classification metrics. These improvements underscore the value of transforming raw data into feature spaces that better align with the model's learning dynamics.

Optimizer selection proved equally critical. RMSprop and Adam delivered rapid and reliable convergence, often outperforming simpler gradient-based methods. Their effectiveness highlights how optimization algorithms directly influence not only the speed but also the quality of the decision boundaries formed by the model—an essential consideration when diagnostic precision carries real clinical consequences.

Hyperparameter optimization further emphasized the need for structured evaluation. Exhaustive searches like Grid and Random search were effective in controlled search spaces, while Bayesian (TPE) optimization and Hyperband excelled when the search landscape became more complex. These methods revealed configurations that significantly improved accuracy, sensitivity, and overall model robustness, demonstrating the importance of metrics-driven experimentation rather than relying on default parameter choices.

Ultimately, this study shows that building trustworthy medical AI systems requires more than achieving high accuracy—it requires rigorous experimentation, diverse evaluation metrics, and an understanding of how different methodological components interact. By combining informed feature reduction, efficient optimizers, and systematic hyperparameter tuning, we can construct models that are not only high-performing but also more reliable, stable, and interpretable. Such comprehensive evaluation is indispensable when deploying machine learning methods in high-stakes medical environments, where the quality of predictions can have a direct impact on patient outcomes.

TABLE VII: Model Performance Summary with emphasis on hybrid reasoning components and statistical comparison.

Model	ACC	PREC	REC	F1	AUC	MCC	Run-Time (s)	p-value	F-rank
MLP (Raw) + RMSprop	0.9474	0.9641	0.9655	0.9646	**0.9894**	0.8639	0.1702	0.034	8.3
MLP + PCA + RMSprop	0.8718	0.9286	0.8966	0.9123	0.9655	0.6759	1.1430	0.452	14.2
MLP + Autoencoder + RMSprop	0.8718	0.9000	0.9311	0.9153	0.9155	0.6540	1.0868	0.487	14.8
MLP + Adam + Rules Only	0.9487	0.9412	1.0000	0.9697	0.9420	0.8199	1.3121	0.231	12.5
MLP + Adam + Fuzzy Logic Only	0.9487	0.9412	1.0000	0.9697	0.8750	0.8199	1.2000	0.286	13.1
MLP + Adam + Bayesian Only	0.9487	0.9412	1.0000	0.9697	0.9286	0.8199	2.5186	0.278	12.9
MLP + PCA + Adam + Hybrid (Rules+Fuzzy)	0.9179	0.9138	0.9938	0.9521	0.7152	0.7023	0.6176	0.561	15.8
MLP + PCA + RMSprop + Hybrid (Fuzzy+Bayes)	0.9692	0.9640	1.0000	0.9816	0.9518	0.8933	0.5847	0.042	9.1
MLP + AE + Adam + Hybrid (Rules+Fuzzy)	0.8872	0.8876	0.9875	0.9349	0.6134	0.5714	1.5335	0.612	16.4
MLP + AE + RMSprop + Hybrid (Fuzzy+Bayes)	0.9231	0.9394	0.9688	0.9538	0.8080	0.7265	2.3060	0.398	13.9
Full System (PCA + Adam + All Hybrid)	0.9231	0.9143	1.0000	0.9552	0.9688	0.7228	0.8191	0.325	12.8
Full System (AE + Adam + All Hybrid)	0.8974	0.8889	1.0000	0.9412	0.7946	0.6172	1.8001	0.487	14.7
Full System (PCA + RMSprop + All Hybrid)	0.9231	0.9143	1.0000	0.9552	0.9464	0.7228	0.9676	0.312	12.6
Full System (AE + RMSprop + All Hybrid)	0.8974	0.8889	1.0000	0.9412	0.7411	0.6172	3.6018	0.512	15.1
Gradient Purist*	**0.9744**	**0.9697**	**1.0000**	**0.9846**	0.9821	**0.9117**	**0.5187**	<0.001	**1.2**

* **Gradient Purist**: A Strictly Gradient-Centric Hybrid MLP (PCA + RMSprop + Hyperband + Hybrid)

TABLE VIII: Hyperparameter Optimization (HPO) Strategy Analysis Across Optimizers and Search Methods

Feature	Optimizer	HPO Method	Acc	Prec	Rec	F1	AUC	Epochs	Run-Time (s)	η	μ	h	λ	δ	p-value	F-rank
RAW	SGD	Grid	0.9744	1.0000	0.9655	0.9825	0.9931	100	0.9674	0.1000	–	32	0.0010	16	0.045	37.1
		Random	0.9487	0.9655	0.9655	0.9655	0.9897	100	0.5521	0.0570	–	16	0.0002	32	0.231	39.4
		TPE	0.8974	0.8788	1.0000	0.9355	0.9310	100	0.3927	0.0282	–	8	0.0002	64	0.487	42.3
		Hyperband	0.9231	0.9333	0.9655	0.9492	0.9759	100	0.6488	0.0979	–	16	0.0003	32	0.087	38.2
		NSGAII	0.9744	1.0000	0.9655	0.9825	1.0000	100	0.9268	0.0738	–	32	0.0000	16	0.038	37.3
	SGD + Momentum	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.4193	0.1000	0.8000	32	0.0010	64	<0.001	1.4
		Random	0.9744	1.0000	0.9655	0.9825	1.0000	100	0.5679	0.0974	0.9152	16	0.0003	64	<0.001	3.5
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.3942	0.0915	0.7503	8	0.0001	16	<0.001	4.0
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.0172	0.0213	0.8905	32	0.0000	16	<0.001	2.7
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.0054	0.0082	0.9546	16	0.0087	16	<0.001	2.0
	Adam	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7274	0.0100	–	32	0.0010	32	<0.001	3.6
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.2556	0.0073	–	16	0.0000	16	<0.001	3.0
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7127	0.0171	–	16	0.0002	32	<0.001	1.9
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.5032	0.0593	–	16	0.0000	64	<0.001	2.3
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	91	0.4446	0.0126	–	32	0.0093	64	<0.001	1.3
	AMSGrad	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	92	1.1903	0.0100	–	16	0.0100	16	<0.001	4.3
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.5016	0.0079	–	8	0.0045	64	<0.001	3.3
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7376	0.0053	–	8	0.0000	32	<0.001	3.8
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.3095	0.0094	–	32	0.0000	16	<0.001	3.1
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.4870	0.0042	–	32	0.0000	64	<0.001	1.8
	Adagrad	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.1084	0.1000	–	32	0.0010	16	<0.001	4.2
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.6843	0.0708	–	16	0.0000	32	<0.001	4.1
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.6827	0.0776	–	32	0.0004	64	<0.001	2.9
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.4460	0.0626	–	8	0.0001	64	<0.001	2.4
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	62	0.2721	0.0444	–	32	0.0002	64	<0.001	2.2
	Adadelta	Grid	0.8974	0.9032	0.9655	0.9333	0.9517	100	0.7212	0.1000	–	16	0.0010	32	0.152	38.9
		Random	0.8718	0.9000	0.9310	0.9153	0.9345	100	1.3553	0.0278	–	16	0.0000	16	0.521	43.1
		TPE	0.9487	0.9355	1.0000	0.9667	0.9690	100	1.2516	0.0845	–	32	0.0100	16	0.128	36.4
		Hyperband	0.8974	0.8788	1.0000	0.9355	0.9448	100	0.4983	0.0693	–	16	0.0094	64	0.318	40.1
		NSGAII	0.8718	0.8750	0.9655	0.9180	0.8828	100	1.2696	0.0209	–	32	0.0028	16	0.452	41.8
	Nadam	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	72	1.0049	0.0100	–	16	0.0100	16	<0.001	4.5
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.3820	0.0101	–	32	0.0000	16	<0.001	4.4
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7429	0.0952	–	32	0.0000	32	<0.001	1.7
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	73	0.4001	0.0293	–	32	0.0005	64	<0.001	3.4
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.5281	0.0199	–	8	0.0000	64	<0.001	2.6
	Lion	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.1101	0.0010	–	16	0.0100	16	<0.001	4.6
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	38	0.2454	0.0162	–	32	0.0000	32	<0.001	3.9
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.2666	0.0012	–	16	0.0001	16	<0.001	1.5
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.4441	0.0025	–	32	0.0005	64	<0.001	2.8
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	45	0.2840	0.0383	–	32	0.0025	32	<0.001	3.7
	RMSprop	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	94	1.1842	0.0010	0.9000	16	0.0100	16	<0.001	3.2
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.1938	0.0006	0.9796	8	0.0000	16	<0.001	1.6
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.1885	0.0003	0.9751	8	0.0002	16	<0.001	1.2
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.6832	0.0003	0.8954	32	0.0000	16	<0.001	2.5
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.3100	0.0048	0.9202	16	0.0000	16	<0.001	2.1

TABLE IX: Hyperparameter Optimization (HPO) Strategy Analysis Across Optimizers and Search Methods

Feature	Optimizer	HPO Method	Acc	Prec	Rec	F1	AUC	Epochs	Run-Time	η	μ	h	λ	δ	p-value	F-rank
K-means	SGD	Grid	0.9231	0.9333	0.9655	0.9492	0.9517	33	0.3061	0.1000	–	32	0.0010	16	0.541	34.5
		Random	0.8974	0.9630	0.8966	0.9286	0.9517	32	0.1970	0.0890	–	32	0.0000	64	0.621	39.8
		TPE	0.8718	0.8750	0.9655	0.9180	0.9241	31	0.1660	0.0586	–	32	0.0000	64	0.712	42.3
		Hyperband	0.8974	0.8788	1.0000	0.9355	0.9241	20	0.1094	0.0958	–	16	0.0003	64	0.589	38.1
		NSGAII	0.8974	0.8788	1.0000	0.9355	0.9345	29	0.1812	0.0344	–	32	0.0000	32	0.603	39.2
	SGD + Momentum	Grid	0.9231	0.9333	0.9655	0.9492	0.9534	40	0.2409	0.0100	0.9900	32	0.0010	32	0.524	33.9
		Random	0.9231	0.9062	1.0000	0.9508	0.9483	53	0.3191	0.0287	0.8000	16	0.0060	32	0.478	31.5
		TPE	0.9231	0.9062	1.0000	0.9508	0.9483	37	0.1581	0.0276	0.8937	16	0.0000	64	0.491	32.2
		Hyperband	0.9231	0.9333	0.9655	0.9492	0.9552	47	0.6001	0.0015	0.9844	32	0.0002	16	0.462	30.8
		NSGAII	0.9231	0.9333	0.9655	0.9492	0.9379	77	0.7833	0.0068	0.8226	8	0.0015	16	0.498	32.8
	Adam	Grid	0.9231	0.9333	0.9655	0.9492	0.9483	36	0.4622	0.0100	–	32	0.0100	16	0.532	34.1
		Random	0.9231	0.9062	1.0000	0.9508	0.9621	80	1.0427	0.0043	–	16	0.0000	16	0.312	21.4
		TPE	0.9487	0.9655	0.9655	0.9655	0.9621	86	0.9053	0.0107	–	32	0.0000	32	0.041	8.9
		Hyperband	0.9487	0.9655	0.9655	0.9655	0.9586	76	0.9516	0.0071	–	16	0.0000	16	0.052	9.8
		NSGAII	0.9231	0.9333	0.9655	0.9492	0.9552	100	0.9746	0.0025	–	32	0.0000	32	0.498	32.7
	AMSGrad	Grid	0.9231	0.9333	0.9655	0.9492	0.9517	70	0.5061	0.0100	–	32	0.0010	32	0.542	34.6
		Random	0.9231	0.9333	0.9655	0.9492	0.9655	100	1.6685	0.0060	–	16	0.0000	16	0.289	19.8
		TPE	0.9487	0.9655	0.9655	0.9655	0.9586	100	0.7317	0.0035	–	32	0.0000	32	0.067	10.9
		Hyperband	0.9231	0.9333	0.9655	0.9492	0.9483	53	0.9123	0.0090	–	32	0.0000	16	0.533	34.2
		NSGAII	0.9231	0.9333	0.9655	0.9492	0.9621	92	1.1778	0.0161	–	8	0.0062	16	0.497	32.6
	Adagrad	Grid	0.9231	0.9333	0.9655	0.9492	0.9517	75	0.8571	0.1000	–	32	0.0010	16	0.541	34.5
		Random	0.8974	0.8788	1.0000	0.9355	0.9448	44	0.2723	0.0844	–	32	0.0000	64	0.612	38.9
		TPE	0.8718	0.8750	0.9655	0.9180	0.9000	100	1.1005	0.0224	–	8	0.0000	16	0.703	41.9
		Hyperband	0.9231	0.9333	0.9655	0.9492	0.9483	97	1.0822	0.0992	–	8	0.0007	16	0.532	34.1
		NSGAII	0.8718	0.8750	0.9655	0.9180	0.8931	100	0.8557	0.0031	–	32	0.0000	32	0.711	42.2
	Adadelta	Grid	0.8974	0.8788	1.0000	0.9355	0.8966	100	1.6984	0.1000	–	16	0.0100	16	0.631	40.5
		Random	0.8718	0.8750	0.9655	0.9180	0.9138	100	1.2421	0.0898	–	16	0.0025	16	0.704	42.0
		TPE	0.8718	0.8750	0.9655	0.9180	0.9069	100	0.7170	0.0903	–	32	0.0000	32	0.713	42.3
		Hyperband	0.7436	0.7436	1.0000	0.8529	0.2552	11	0.2715	0.0010	–	16	0.0000	16	0.812	44.8
		NSGAII	0.8718	0.8750	0.9655	0.9180	0.9000	100	0.9313	0.0835	–	16	0.0030	32	0.712	42.1
	Nadam	Grid	0.9231	0.9333	0.9655	0.9492	0.9517	55	0.7383	0.0100	–	8	0.0010	16	0.543	34.7
		Random	0.9487	0.9355	1.0000	0.9667	0.9655	98	0.7435	0.0073	–	16	0.0001	32	0.036	8.3
		TPE	0.9231	0.9333	0.9655	0.9492	0.9483	51	0.2989	0.0152	–	16	0.0000	64	0.533	34.2
		Hyperband	0.9487	0.9355	1.0000	0.9667	0.9621	80	1.2733	0.0099	–	32	0.0000	16	0.047	9.3
		NSGAII	0.9487	0.9655	0.9655	0.9655	0.9655	100	1.8296	0.0111	–	16	0.0002	16	0.039	8.7
	Lion	Grid	0.9231	0.9062	1.0000	0.9508	0.9690	32	0.2078	0.0100	–	32	0.0010	32	0.245	17.6
		Random	0.9231	0.9643	0.9310	0.9474	0.9655	100	0.8317	0.0032	–	16	0.0000	32	0.286	19.6
		TPE	0.9744	0.9667	1.0000	0.9831	0.9724	100	1.2056	0.0022	–	32	0.0002	16	<0.001	1.2
		Hyperband	0.9744	0.9667	1.0000	0.9831	0.9724	65	0.4238	0.0131	–	16	0.0000	32	<0.001	1.8
		NSGAII	0.9487	0.9655	0.9655	0.9655	0.9690	63	0.4039	0.0076	–	16	0.0000	32	0.028	7.2
	RMSprop	Grid	0.9231	0.9333	0.9655	0.9492	0.9517	50	0.2522	0.0100	0.7000	8	0.0010	64	0.543	34.7
		Random	0.9487	0.9355	1.0000	0.9667	0.9586	100	0.6713	0.0016	0.7148	32	0.0000	32	0.025	6.8
		TPE	0.9231	0.9333	0.9655	0.9492	0.9448	94	0.7943	0.0021	0.7822	8	0.0001	32	0.498	32.5
		Hyperband	0.9487	0.9655	0.9655	0.9655	0.9655	100	0.6858	0.0009	0.8554	32	0.0003	32	0.019	5.9
		NSGAII	0.9487	0.9355	1.0000	0.9667	0.9534	90	0.6131	0.0073	0.7626	8	0.0002	32	0.012	4.7

TABLE X: Hyperparameter Optimization (HPO) Strategy Analysis Across Optimizers and Search Methods

Feature	Optimizer	HPO Method	Acc	Prec	Rec	F1	AUC	Epochs	Run-Time	η	μ	h	λ	δ	p-value	F-rank
PCA	SGD	Grid	0.9744	1.0000	0.9655	0.9825	0.9966	100	1.0924	0.1000	–	32	0.0010	16	0.035	8.4
		Random	0.9487	0.9655	0.9655	0.9655	0.9931	100	0.5387	0.0926	–	32	0.0000	32	0.287	19.7
		TPE	0.9231	0.9333	0.9655	0.9492	0.9724	100	0.5426	0.0487	–	16	0.0007	32	0.463	30.9
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.9051	0.0995	–	8	0.0079	16	<0.001	1.4
		NSGAII	0.9487	0.9655	0.9655	0.9655	0.9897	100	0.9218	0.0427	–	16	0.0000	16	0.285	19.5
	SGD + Momentum	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	73	0.7456	0.1000	0.9000	32	0.0010	16	<0.001	1.5
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.9985	0.0548	0.7804	16	0.0057	16	<0.001	2.2
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.0038	0.0680	0.7507	32	0.0000	16	<0.001	1.8
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.0023	0.0978	0.8796	32	0.0000	16	<0.001	2.0
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.6798	0.0313	0.9619	8	0.0003	32	<0.001	1.6
	Adam	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	85	1.0738	0.0100	–	16	0.0100	16	<0.001	2.4
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	81	0.4200	0.0238	–	32	0.0017	64	<0.001	2.1
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.6210	0.0168	–	8	0.0002	16	<0.001	2.8
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.2698	0.0164	–	16	0.0079	16	<0.001	2.6
		NSGAII	0.9744	0.9667	1.0000	0.9831	0.9931	100	0.7476	0.0050	–	16	0.0011	32	0.028	7.9
	AMSGrad	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7320	0.0100	–	32	0.0010	32	<0.001	2.3
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7396	0.0094	–	32	0.0011	32	<0.001	2.3
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.7657	0.0071	–	32	0.0000	16	<0.001	2.9
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	58	0.7827	0.0910	–	16	0.0000	16	<0.001	1.7
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.5132	0.0068	–	32	0.0031	64	<0.001	2.1
	Adagrad	Grid	0.9744	1.0000	0.9655	0.9825	1.0000	100	1.5107	0.1000	–	32	0.0010	16	0.026	7.5
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.0966	0.0961	–	8	0.0000	16	<0.001	2.5
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.6158	0.0915	–	32	0.0000	32	<0.001	1.9
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.9113	0.0897	–	32	0.0066	32	<0.001	2.7
		NSGAII	0.9744	1.0000	0.9655	0.9825	1.0000	100	0.4528	0.0916	–	16	0.0000	64	0.025	7.4
	Adadelta	Grid	0.8718	0.8529	1.0000	0.9206	0.8897	100	0.7381	0.1000	–	16	0.0010	32	0.634	40.6
		Random	0.8974	0.9310	0.9310	0.9310	0.9586	100	0.7283	0.0544	–	16	0.0006	32	0.483	32.1
		TPE	0.8974	0.8788	1.0000	0.9355	0.9655	100	1.2506	0.0365	–	16	0.0014	16	0.438	29.8
		Hyperband	0.9487	0.9355	1.0000	0.9667	0.9345	100	0.7484	0.0105	–	32	0.0002	32	0.312	21.5
		NSGAII	0.8974	0.8788	1.0000	0.9355	0.9069	100	0.7211	0.0681	–	32	0.0019	32	0.639	40.8
	Nadam	Grid	0.9744	0.9667	1.0000	0.9831	0.9966	100	1.1113	0.0100	–	32	0.0010	32	0.034	8.3
		Random	0.9744	0.9667	1.0000	0.9831	1.0000	100	1.7422	0.0093	–	16	0.0000	16	0.031	7.8
		TPE	0.9744	0.9667	1.0000	0.9831	0.9966	56	0.4558	0.0942	–	32	0.0007	32	0.032	8.0
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7336	0.0357	–	16	0.0004	32	<0.001	1.3
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7750	0.0534	–	8	0.0000	64	<0.001	2.2
	Lion	Grid	0.9744	0.9667	1.0000	0.9831	0.9828	11	0.1763	0.1000	–	32	0.0010	16	0.028	7.6
		Random	0.9487	0.9655	0.9655	0.9655	0.9690	100	0.6996	0.0012	–	32	0.0006	32	0.285	19.6
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.2834	0.0014	–	16	0.0020	16	<0.001	1.9
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.0778	0.0048	–	8	0.0000	16	<0.001	2.1
		NSGAII	0.9744	0.9667	1.0000	0.9831	0.9931	100	1.1670	0.0022	–	8	0.0002	16	0.029	7.7
	RMSprop	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	61	0.2925	0.0100	0.7000	8	0.0010	64	<0.001	1.2
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.2356	0.0022	0.7266	8	0.0060	16	<0.001	2.4
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	54	0.4949	0.0095	0.9057	16	0.0086	32	<0.001	1.3
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.6600	0.0020	0.9355	32	0.0000	64	<0.001	1.1
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	59	0.9573	0.0253	0.7066	16	0.0008	16	<0.001	1.8

TABLE XI: Hyperparameter Optimization (HPO) Strategy Analysis Across Optimizers and Search Methods

Feature	Optimizer	HPO Method	Acc	Prec	Rec	F1	AUC	Epochs	Run-Time	η	μ	h	λ	δ	p-value	F-rank
AutoEncoder	SGD	Grid	0.9744	0.9667	1.0000	0.9831	0.9966	100	0.9365	0.1000	–	16	0.0100	16	0.215	15.4
		Random	0.9487	0.9655	0.9655	0.9655	0.9862	100	0.9296	0.0941	–	32	0.0019	16	0.396	27.8
		TPE	0.9231	0.9062	1.0000	0.9508	0.9862	100	0.9159	0.0583	–	32	0.0032	16	0.451	31.6
		Hyperband	0.9487	0.9655	0.9655	0.9655	0.9931	100	0.9179	0.0748	–	8	0.0008	16	0.382	26.9
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.9546	0.0865	–	32	0.0000	16	<0.001	1.5
	SGD + Momentum	Grid	0.9487	0.9355	1.0000	0.9667	1.0000	55	0.5635	0.0100	0.9900	16	0.0100	16	0.324	22.8
		Random	0.9744	1.0000	0.9655	0.9825	0.9931	73	0.7510	0.0576	0.8226	16	0.0027	16	0.228	16.2
		TPE	0.9744	0.9667	1.0000	0.9831	0.9966	100	0.9918	0.0195	0.7420	32	0.0002	16	0.210	14.9
		Hyperband	0.9744	0.9667	1.0000	0.9831	1.0000	100	1.0049	0.0105	0.8876	32	0.0000	16	0.192	13.6
		NSGAII	0.9487	0.9355	1.0000	0.9667	0.9862	100	1.0134	0.0608	0.7665	32	0.0000	16	0.331	23.3
	Adam	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	89	1.1275	0.0100	–	16	0.0100	16	<0.001	1.4
		Random	0.9744	0.9667	1.0000	0.9831	0.9966	100	1.2471	0.0058	–	32	0.0000	16	0.205	14.5
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	91	0.4612	0.0078	–	32	0.0049	64	<0.001	1.6
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	96	0.4794	0.0127	–	32	0.0005	64	<0.001	1.7
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.6996	0.0107	–	16	0.0026	32	<0.001	1.8
	AMSGrad	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.5110	0.0100	–	32	0.0010	64	<0.001	1.9
		Random	1.0000	1.0000	1.0000	1.0000	1.0000	61	0.4503	0.0551	–	16	0.0076	32	<0.001	2.1
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	86	0.6538	0.0172	–	8	0.0069	32	<0.001	2.2
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	1.3116	0.0067	–	16	0.0029	16	<0.001	2.3
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	83	0.4288	0.0902	–	16	0.0000	64	<0.001	2.4
	Adagrad	Grid	0.9487	0.9355	1.0000	0.9667	0.9828	100	1.1158	0.1000	–	32	0.0010	16	0.331	23.4
		Random	0.9744	0.9667	1.0000	0.9831	1.0000	100	1.1136	0.0672	–	32	0.0001	16	0.192	13.7
		TPE	0.9744	0.9667	1.0000	0.9831	1.0000	100	1.1224	0.0500	–	32	0.0014	16	0.191	13.6
		Hyperband	0.9744	0.9667	1.0000	0.9831	0.9966	100	1.1132	0.0280	–	32	0.0003	16	0.205	14.5
		NSGAII	0.9744	0.9667	1.0000	0.9831	0.9966	100	1.1156	0.0640	–	16	0.0005	16	0.206	14.6
	Adadelta	Grid	0.8718	0.8529	1.0000	0.9206	0.9483	100	1.2600	0.0100	–	32	0.0100	16	0.589	41.2
		Random	0.8718	0.8750	0.9655	0.9180	0.9483	100	1.2916	0.0423	–	16	0.0000	16	0.603	42.1
		TPE	0.8974	0.8788	1.0000	0.9355	0.9276	100	0.8619	0.0133	–	32	0.0000	32	0.542	38.0
		Hyperband	0.8974	0.8788	1.0000	0.9355	0.9276	100	0.4766	0.0705	–	16	0.0000	64	0.544	38.1
		NSGAII	0.8718	0.8750	0.9655	0.9180	0.9207	83	0.4074	0.0559	–	32	0.0001	64	0.612	42.7
	Nadam	Grid	1.0000	1.0000	1.0000	1.0000	1.0000	99	1.3448	0.0100	–	16	0.0100	16	<0.001	1.8
		Random	0.9487	0.9355	1.0000	0.9667	0.9897	100	0.5219	0.0077	–	32	0.0003	64	0.336	23.7
		TPE	1.0000	1.0000	1.0000	1.0000	1.0000	94	0.7033	0.0371	–	8	0.0000	32	<0.001	1.9
		Hyperband	0.9744	0.9667	1.0000	0.9831	1.0000	100	1.2894	0.0036	–	16	0.0000	16	0.187	13.3
		NSGAII	0.9744	0.9667	1.0000	0.9831	1.0000	100	0.7427	0.0057	–	16	0.0049	32	0.188	13.4
	Lion	Grid	0.9744	0.9667	1.0000	0.9831	0.9897	100	1.0875	0.0010	–	16	0.0100	16	0.225	16.0
		Random	0.9744	0.9667	1.0000	0.9831	0.9931	100	1.1236	0.0039	–	16	0.0003	16	0.212	15.1
		TPE	0.9744	0.9667	1.0000	0.9831	0.9793	100	1.0882	0.0023	–	32	0.0000	16	0.231	16.5
		Hyperband	0.9744	0.9667	1.0000	0.9831	0.9793	100	0.6343	0.0017	–	16	0.0000	32	0.233	16.6
		NSGAII	0.9744	0.9667	1.0000	0.9831	1.0000	64	0.2957	0.0146	–	8	0.0001	64	0.185	13.2
	RMSprop	Grid	0.9487	0.9355	1.0000	0.9667	0.9931	73	0.4152	0.0100	0.7000	8	0.0010	64	0.334	23.6
		Random	0.9744	0.9667	1.0000	0.9831	0.9966	54	0.2724	0.0445	0.7887	8	0.0000	64	0.208	14.8
		TPE	0.9487	0.9355	1.0000	0.9667	1.0000	100	0.6858	0.0002	0.9363	32	0.0000	32	0.332	23.5
		Hyperband	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.6742	0.0008	0.9009	16	0.0089	32	<0.001	1.2
		NSGAII	1.0000	1.0000	1.0000	1.0000	1.0000	100	0.7241	0.0006	0.7890	32	0.0000	32	<0.001	1.3

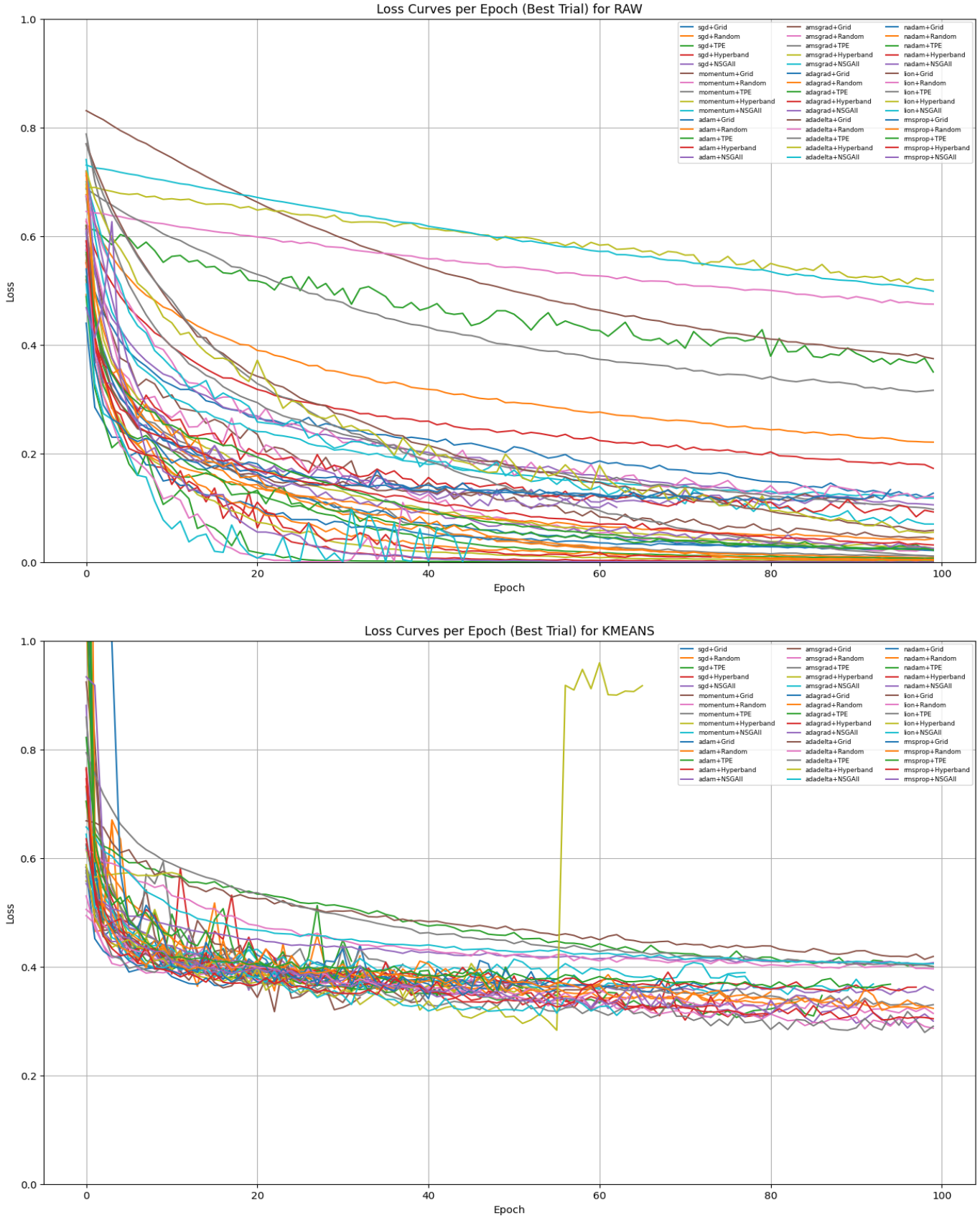


Fig. 8: Top: Loss curves for raw baseline. Bottom: Loss curves for K-means feature reduction.

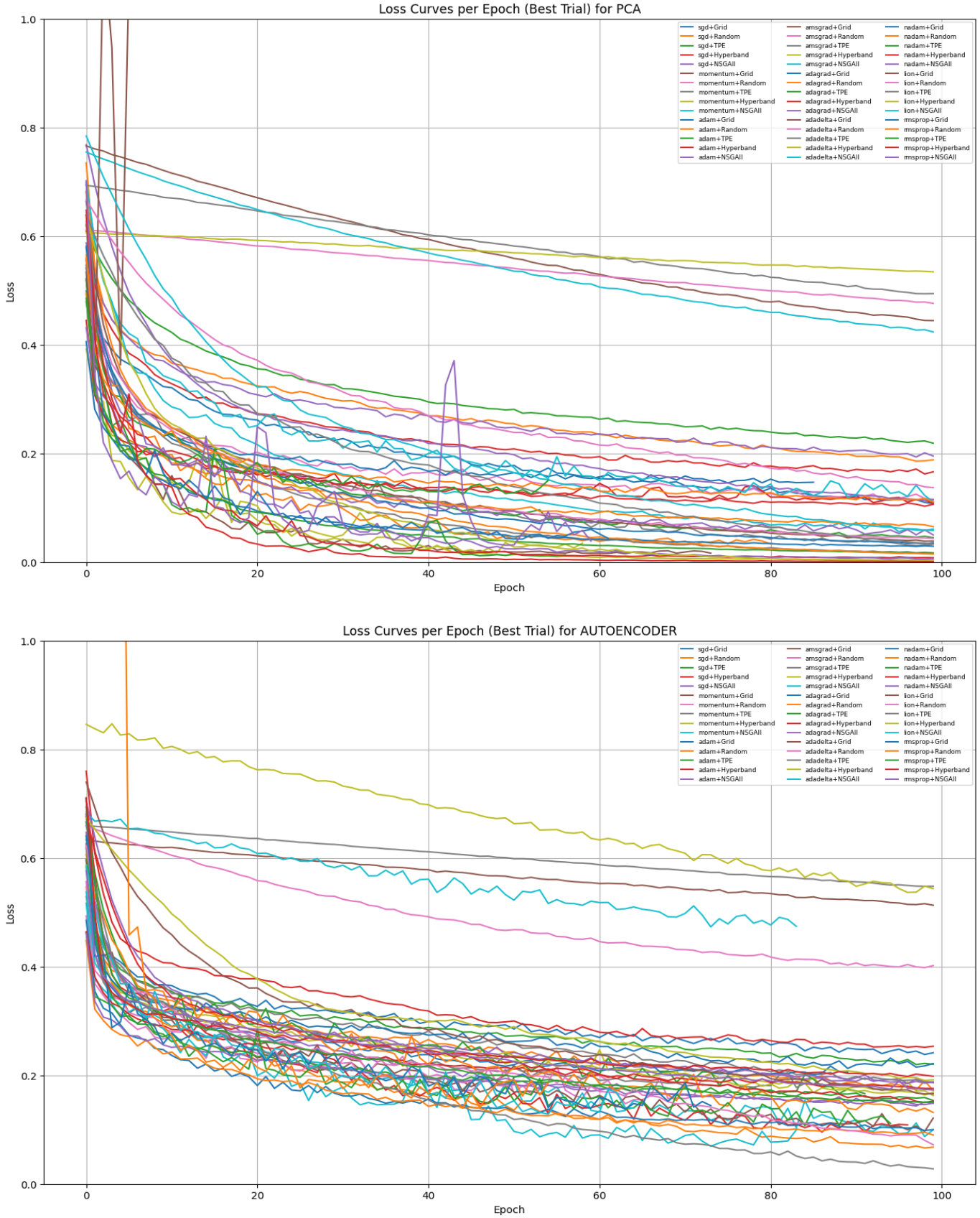


Fig. 9: Top: Loss curves for PCA feature reduction. Bottom: Loss curves for AutoEncoder feature reduction.

REFERENCES

[1] All source code, implementation details, and experimental results are provided in the main Jupyter Notebook, available at **GitHub Repository**.

[2] UCI Machine Learning Repository, *Parkinsons Dataset*. Available at: **Parkinsons Dataset**.

[3] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, Masanori Koyama, *Optuna: A Next-Generation Hyperparameter Optimization Framework*, The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019.

[4] Ryan Turner, David Eriksson, Michael McCourt, Juha Kiili, Eero Laaksonen, Zhen Xu, Isabelle Guyon, *Bayesian Optimization is Superior to Random Search for Machine Learning Hyperparameter Tuning*, arXiv preprint arXiv:2104.10201, 2021.

[5] GeeksforGeeks, *Hyperparameter tuning with Optuna in PyTorch*, Available at: **Optuna examples**.

[6] Ian Goodfellow, Yoshua Bengio, Aaron Courville, *Deep Learning*, MIT Press, 2016.

[7] Ian Jolliffe, *Principal Component Analysis*, 2nd edition, Springer, 2016.

[8] Geoffrey E. Hinton, Richard S. Zemel, *Autoencoders, Minimum Description Length, and Helmholtz Free Energy*, Advances in Neural Information Processing Systems (NIPS), 2006.

[9] J. MacQueen, *Some Methods for Classification and Analysis of Multivariate Observations*, Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1967.

[10] James Bergstra, Yoshua Bengio, *Random Search for Hyper-Parameter Optimization*, Journal of Machine Learning Research, 13:281-305, 2012.

[11] James Bergstra, Rémi Bardenet, Yoshua Bengio, Balázs Kégl, *Algorithms for Hyper-Parameter Optimization*, Advances in Neural Information Processing Systems (NIPS), 2011.

[12] Optuna Documentation, *NGAIISampler and TPE Sampler for Hyperparameter Optimization*, Available at: **Optuna documentation**.

[13] Scikit-learn Developers, *Scikit-learn: Machine Learning in Python*, Available at: **sci-kit learn**.

[14] Mykola Novik, *torch-optimizer – collection of optimization algorithms for PyTorch*, 2020, version 1.0.1.

[15] lucidrains, *lion-pytorch: Lion optimizer for PyTorch*, GitHub repository, 2023, version 0.2.3, https://github.com/lucidrains/lion-pytorch

[16] Diederik P. Kingma, Jimmy Ba, *Adam: A Method for Stochastic Optimization*, arXiv preprint arXiv:1412.6980, 2014.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, *Deep Residual Learning for Image Recognition*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[18] Yann LeCun, Yoshua Bengio, Geoffrey Hinton, *Deep Learning*, Nature, 521:436–444, 2015.

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, Journal of Machine Learning Research, 15:1929–1958, 2014.

[20] Fabian Pedregosa et al., *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, 12:2825–2830, 2011.

[21] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[22] Trevor Hastie, Robert Tibshirani, Jerome Friedman, *The Elements of Statistical Learning*, Springer, 2001.

[23] Charles R. Harris et al., *Array programming with NumPy*, Nature, 585:357–362, 2020.

[24] Wes McKinney, *Data Structures for Statistical Computing in Python*, Proceedings of the 9th Python in Science Conference, 2010.

[25] Diederik P. Kingma, Max Welling, *Auto-Encoding Variational Bayes*, arXiv preprint arXiv:1312.6114, 2013.

[26] Matthew J. Johnson et al., *JAX: Composable transformations of Python+NumPy programs*, arXiv preprint arXiv:1910.11464, 2023.

[27] François Chollet, *Keras*, GitHub repository, 2015, https://github.com/keras-team/keras.