Few-Shot High-Dimensional Feature Selection with Lagrange Programming Neural Networks

Nourane Fradi[1] **and Jérémie Cabessa**[2,3] **and Anis Zeglaoui**[4]

[1]MaPSFA-ESST Hammam Sousse, University of Sousse, 4011 Hammam-Sousse, Tunisia
[2]DAVID Laboratory, University of Versailles (UVSQ) – Paris-Saclay, 78035 Versailles, France
[3]Institute of Computer Science of the Czech Academy of Sciences, 18207 Prague 8, Czech Republic
[4]MaPSFA-ESST Hammam Sousse, ENISo, University of Sousse, 4023 Sousse, Tunisia

nourane.fradi@essths.u-sousse.tn jeremie.cabessa@uvsq.fr anis.zeglaoui@eniso.u-sousse.tn

***Abstract*—Few-shot learning and high-dimensional feature selection represent significant challenges in machine learning. In this work, we introduce LPNN-FS, a novel feature selection (FS) technique based on a Lagrange Programming Neural Network ($\mathcal{P}_k$-LPNN) originally developed in the context of compressive sampling. LPNN-FS is a continuous-time recurrent neural network whose dynamics inherently converges to an optimal sparse solution of the LASSO, with its equilibrium point acting as a feature selector. Evaluated in combination with specific downstream classifiers, our model is tested on both synthetic and real-world benchmark datasets characterized by high dimensionality and a limited number of observations. The results demonstrate that LPNN-FS competes with or outperforms state-of-the-art methods such as LASSO, k-best, and sparse PCA in this few-shot, high-dimensional setting. Overall, this study opens new avenues for leveraging compressive sampling-based techniques in challenging feature selection problems.**

***Index Terms*—Recurrent Neural Networks, Lagrange Programming Neural Networks (LPNNs), Few-shot Learning, Feature Selection, LASSO, High-Dimensional Data.**

I. Introduction

High-dimensional datasets, often composed of thousands of features, are now commonplace in areas such as genomics [1], image processing [2], and text mining [3]. While these datasets offer immense potential for insights, they also introduce significant obstacles, including the risk of overfitting, reduced model interpretability, and increased computational requirements. Moreover, many of these features are irrelevant or redundant, further complicating the learning process and degrading model performance.

To address these issues, feature selection has emerged as a crucial pre-processing step for identifying the most relevant features, enhancing efficiency, and improving overall model performance [4], [5]. Feature selection methods can be classified into three main categories: filter-based [6], wrapper-based [6], and embedded approaches [7].

Filter-based methods assess the relevance of features independently of any classifier, using statistical metrics such as mutual information or correlation. In contrast, wrapper methods evaluate subsets of features by testing their performance with a chosen machine learning model. They iteratively add or remove features to identify the combination that yields the best performance [6]. Embedded methods, on the other hand, integrate feature selection directly into the model training process.

Each approach has its own strengths and limitations. Wrapper methods tend to be more computationally expensive, particularly for high-dimensional datasets. Conversely, filter methods are computationally efficient and easy to implement, making them well-suited for large datasets, and compatible with a wide range of machine learning models [8].

A prominent feature selection method is the LASSO (Least Absolute Shrinkage and Selection Operator) [7], formulated as a least-squares minimization problem with ℓ_1-norm regularization, aimed at identifying sparse solutions. Widely employed in the field of compressive sampling for high-dimensional data analysis [9], the LASSO faces a critical limitation: the non-differentiability of its ℓ_1-norm constraint. To address this issue, a Lagrange Programming Neural Network (LPNN) approach to solving the LASSO problem has been proposed ($\mathcal{P}_k$-LPNN) [10]. In this methodology, a carefully constructed smoothing function approximating the ℓ_1-norm of LASSO is introduced. On this basis, a sequence of smoothed $\mathcal{P}_k$ minimization problems is formulated, with their k-parametrized solutions shown to asymptotically converge to the optimal solution of the LASSO. A Lagrange Programming Neural Network (LPNN) is then derived from these $\mathcal{P}_k$ problems. The neurodynamics of the LPNN is shown to evolve toward a stable equilibrium point that closely approximates the solution trajectory of the $\mathcal{P}_k$ problems, and consequently, the solution of the original LASSO.

In this paper, we introduce LPNN-FS, a filter-based feature selection method specifically designed for high-dimensional datasets with a limited number of ob-

servations. This method is built upon the $\mathcal{P}_k$-LPNN model [10] and its theoretical foundations will be presented in a subsequent journal paper. LPNN-FS is a continuous-time recurrent neural network whose dynamics naturally converge to an optimal sparse solution of the LASSO problem, with its equilibrium point serving as a feature selector. We employ LPNN-FS as a preprocessing feature selection step, preceding various common classifiers such as k-Nearest Neighbors (KNN), Logistic Regression (LR), and Gaussian Naive Bayes (NBC). In this context, we show that LPNN-FS competes with or outperforms conventional feature selection methods, including LASSO, k-best, and sparse PCA, on both synthetic and real-world high-dimensional datasets. This study highlights the potential of using compressive sampling-based techniques to address complex feature selection challenges.

This paper is organized as follows: Section II provides a comprehensive review of related works. Section III outlines the mathematical formulation and implementation of the LPNN-FS method. Section IV details the datasets and experimental setup used for evaluation. Section V presents the results and insights gained from our experiments. Finally, Section VI concludes with potential directions for future research.

II. Related Works

LASSO (Least Absolute Shrinkage and Selection Operator) [7] is a powerful regression technique known for its ability to perform feature selection, by shrinking irrelevant or redundant feature coefficients to exactly zero through ℓ_1-norm regularization. However, the LASSO problem suffers from the non-differentiability of its ℓ_1-norm constraint, which affects its numerical resolution, and potentially, its computational efficiency. Zeglaoui et al. [10] introduced a neurodynamic model, called $\mathcal{P}_k$-LPNN, which approximates the non-differentiable ℓ_1-norm at zero using a specific k-smoothing function, transforming the LASSO problem into a sequence of differentiable and convex $\mathcal{P}_k$ formulations that are solved using a recurrent neural network. Their approach competes with existing methods from compressive sampling (CS), such as LASSO-LPNN [11] and active set approaches [12], but lacks applicability to problems more directly associated with machine learning (ML).

Our LPNN-FS method addresses these limitations by approximating the ℓ_1-norm using a novel, carefully designed smoothing function that satisfies specific theoretical conditions. The convergence of LPNN-FS is theoretically established, improving upon that of $\mathcal{P}_k$-LPNN, demonstrating numerical improvements over the earlier $\mathcal{P}_k$-LPNN version while addressing its computational inefficiencies for large k values. This advancement enhances few-shot, high-dimensional feature selection.

Recent advancements in Markov chain-based feature selection, such as the Bird's Eye View (BEV) technique [13], have gained attention for their innovative, nature-inspired approach. BEV combines Evolutionary Algorithms, Genetic Algorithms for agent selection, Dynamic Markov Chains for movement within the search space, and Reinforcement Learning for rewarding progress. The use of BEV to select relevant features and significantly reduce the number of coefficients appears to outperform state-of-the-art feature selection methods. However, the diversity of techniques and algorithms involved requires extensive fine-tuning of numerous hyperparameters, making the process cumbersome. While LPNN-FS does not outperform BEV on selected benchmarks, our method still delivers competitive results with significantly reduced model complexity and computation time. Specifically, LPNN-FS involves only one hyperparameter tuning, which is easy to set and generally leads to fast convergence. Additionally, while Markov chain-based feature selection techniques are inherently stochastic and may produce suboptimal solutions, LPNN-FS is guaranteed to attain LASSO optimal solutions.

Serik et al. [14] investigated the classification of Parkinson's disease using a speech dataset with 754 features and 756 instances, applying various feature selection techniques, such as Principal Component Analysis (PCA) [15], Minimum Redundancy Maximum Relevance (mRMR) [16], and SelectKBest [17], along with different classifiers like SVM, DT, LR, RF, KNN, NB, DNN, and 1D CNN. Their results highlighted the combination of KNN and PCA as the most effective.

Building on their work, we compare the performance of LPNN-FS to that of Sparse PCA and SelectKBest by applying downstream several classifiers to the reduced feature set. As we will demonstrate later, LPNN-FS consistently outperforms these techniques, especially when combined with LR.

III. Model

We propose LPNN-FS, a novel technique for few-shot feature selection in high-dimensional settings, which reformulates the foundational LASSO problem and addresses its resolution using Lagrange Programming Neural Networks (LPNNs). The theoretical foundations of the proposed model will be developed in a subsequent journal paper.

One of the most representative models for feature selection is the LASSO problem [7], which can be formulated as a least-squares minimization problem subject to an ℓ_1-norm constraint:

$$\mathcal{P}: \min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}\left\|\mathbf{y}-\mathbf{X}\boldsymbol{\beta}\right\|_2^2 \quad \text{s.t.} \quad \left\|\boldsymbol{\beta}\right\|_1 \leq \eta, \tag{1}$$

where $\boldsymbol{\beta} \in \mathbb{R}^p$ is the coefficient vector, $\mathbf{X} \in \mathbb{R}^{n\times p}$ is the feature matrix, $\mathbf{y} \in \mathbb{R}^n$ is the target vector, and η is a

non-negative sparsity parameter. The high-dimensional setting is characterized by $n < p$, while the few-shot context reflects the fact that $n << p$.

The LPNN framework is a neurodynamic approach designed for solving specific types of constrained optimization problems by modeling the optimization process as a dynamical system. Formally, consider the problem of minimizing an objective function $f(\boldsymbol{\beta})$ under equality constraints $h(\boldsymbol{\beta}) = 0$

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} f(\boldsymbol{\beta}) \quad \text{s.t} \quad h(\boldsymbol{\beta}) = 0, \tag{2}$$

where $\boldsymbol{\beta} \in \mathbb{R}^p$ is the state of the system, $f : \mathbb{R}^p \longrightarrow \mathbb{R}$ and $h : \mathbb{R}^p \longrightarrow \mathbb{R}^s$ with $s < p$. If the functions f and h are both convex and twice differentiable, the *Lagrange Programming Neural Network (LPNN)* is a recurrent neural network built upon a dynamical system derived from the Karush-Kuhn-Tucker (KKT) conditions in convex optimization [18] as:

$$\begin{cases} \frac{d\boldsymbol{\beta}}{dt} = -\nabla_{\boldsymbol{\beta}}\mathcal{L}(\boldsymbol{\beta},\boldsymbol{\mu}), \\ \frac{d\boldsymbol{\mu}}{dt} = \nabla_{\boldsymbol{\mu}}\mathcal{L}(\boldsymbol{\beta},\boldsymbol{\mu}), \end{cases} \tag{3}$$

where $\mathcal{L}(\boldsymbol{\beta},\boldsymbol{\mu}) = f(\boldsymbol{\beta}) + \boldsymbol{\mu}^T h(\boldsymbol{\beta})$ is the Lagrangian, and $\boldsymbol{\mu}$ is the non-negative vector of Lagrange multipliers. The dynamical system (3) can be seen as a continuous-time recurrent neural network composed of two types of neurons: variable neurons and Lagrange neurons. The variable neurons work to minimize the Lagrangian function by decreasing the objective value, while the Lagrange neurons enforce the sparsity property of the related signal. However, directly applying LPNN to the LASSO problem is infeasible due to the ℓ_1-norm constraint, which is non-differentiable at zero and is expressed as an inequality.

To address this issue, we propose a twofold tailored modification for the ℓ_1-norm constraint. First, we construct a differentiable approximation of the constraint. For any $x \in \mathbb{R}$ and $k \geq 1$, we introduce the following smooth approximation for the absolute value [19]:

$$\widehat{\varphi}(x,k) = \sqrt{x^2 + \frac{1}{k^2}}. \tag{4}$$

This smooth approximation function must satisfy both the criteria outlined in [19] and the conditions specified in Definition 1 (adapted from [20]) to guarantee the theoretical properties of our model stated below.

Definition 1. Let $\varphi : \mathbb{R} \to \mathbb{R}$ be a locally Lipschitz function. The function $\widehat{\varphi} : \mathbb{R} \times [1, +\infty) \mapsto \mathbb{R}$ is called a *smoothing parametric scalar function* for φ when the following conditions hold

1) For any fixed $k \geq 1$, the function $x \mapsto \widehat{\varphi}(x,k)$ is continuously differentiable in $\mathbb{R}$, and for any fixed $x \in \mathbb{R}$, the function $k \mapsto \widehat{\varphi}(x,k)$ is differentiable in $[1, +\infty)$.
2) For any fixed $x \in \mathbb{R}$,
$$\lim_{k\to+\infty} \widehat{\varphi}(x,k) = \varphi(x)\,.$$
3) There exists a positive constant $C_{\widehat{\varphi}} > 0$ such that
$$\left|\frac{\partial\widehat{\varphi}(x,k)}{\partial k}\right| \leq C_{\widehat{\varphi}}, \ \ \forall x \in \mathbb{R}, \ \forall k \in [1, +\infty)\,.$$
4) For any $x \in \mathbb{R}$, one has
$$\lim_{k\to+\infty} \left\{ \left[\frac{\partial\widehat{\varphi}}{\partial z}(z,k)\right]_{z=x} \right\} = \lim_{k\to+\infty} \frac{\partial\widehat{\varphi}}{\partial x}(x,k) \in \partial\varphi(x)\,,$$
where ∂ denotes the subdifferential operator [18].[1]

It can be shown that $\widehat{\varphi}(x,k)$, referred to as a *k-approximate function*, satisfies the regularization properties of Definition 1. By extending this k-approximate function to the p-dimensional setting, the ℓ_1-norm can be approximated as

$$\|\boldsymbol{\beta}\|_1 = \sum_{i=1}^{p} |\beta_i| \simeq \sum_{i=1}^{p} \widehat{\varphi}(\beta_i,k) = \widehat{\Psi}(\boldsymbol{\beta},k)\,. \tag{5}$$

Based on these considerations, the LASSO problem (1) can be replaced by the following smooth optimization problem:

$$\mathcal{P}_k : \ \min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \quad \text{s.t.} \quad \widehat{\Psi}(\boldsymbol{\beta},k) \leq \eta\,. \tag{6}$$

The $\mathcal{P}_k$ problem is convex which guarantees the existence of at least one optimal solution. Furthermore, it can be shown that any optimal solution $\hat{\boldsymbol{\beta}}^{(k)}$ of $\mathcal{P}_k$ converges asymptotically (when $k \to \infty$) to a solution of the original LASSO problem (1).

Secondly, the integration of the $\mathcal{P}_k$ formulation into the LPNN framework necessitates transforming the inequality constraint in (6) into an equality. Formally, we proved the following proposition.

Proposition 1. *The smooth optimization problem $\mathcal{P}_k$ (6) can be reformulated into a $\mathcal{P}_k$-LPNN model defined as:*

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \quad s.t \quad \widehat{\Psi}(\boldsymbol{\beta},k) = \eta, \tag{7}$$

provided that

$$\eta \leq \frac{\left\|\mathbf{X}^T\mathbf{y}\right\|_2}{\left\|\mathbf{X}^T\mathbf{X}\right\|_2} - \frac{p}{2k}. \tag{8}$$

The feasibility of this formulation is guaranteed under specific conditions on η and valid for every $k \geq 1$.

The LPNN neurodynamic system (3) associated with the $\mathcal{P}_k$-LPNN model (7–8), referred to as the *LPNN feature selection (LPNN-FS) model*, is given by:

$$\begin{cases} \frac{d\boldsymbol{\beta}}{dt} = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) - \mu\nabla_{\boldsymbol{\beta}}\widehat{\Psi}(\boldsymbol{\beta},k), \\ \frac{d\mu}{dt} = \widehat{\Psi}(\boldsymbol{\beta},k) - \eta, \end{cases} \tag{9}$$

[1]In our case, $\partial|x| = \{-1\}$ if $x < 0$, $\partial|x| = \{1\}$ if $x > 0$ and $\partial|x| = [-1, 1]$ if $x = 0$.

where μ is the Lagrange scalar multiplier associated with the fresh equality constraint. The implementation of the LPNN-FS through its dynamics on an example are illustrated in Figures 1 and 2. The network consists of one Lagrange neuron and p variable neurons. Based on the current states, each variable neuron computes the time derivatives $\frac{d\beta_i}{dt}$ $(i = 1, \ldots, p)$ and the Lagrange neuron computes $\frac{d\mu}{dt}$. The outputs of the integrator are $\tilde{\beta}$ and $\tilde{\mu}$. Formally, the following recurrent equations are used:

$$\begin{aligned}\boldsymbol{\beta}(t + \Delta t) &= \boldsymbol{\beta}(t) + \Delta t \cdot \Phi(t, \Delta t, \boldsymbol{\beta}(t), \mu(t)) \\ \mu(t + \Delta t) &= \mu(t) + \Delta t \cdot \Phi(t, \Delta t, \boldsymbol{\beta}(t), \mu(t)),\end{aligned}$$

where, Δt is a small positive time step, which can be either fixed or adaptively adjusted during the simulation, and the integrator Φ is the fourth-order Runge-Kutta method. At the end of the recurrent iterations, the outputs of the variable neurons, $\tilde{\beta}_i$, are passed to a threshold neuron T which selects the N_z largest components and sets the remaining coefficients to zero. Meanwhile, the final output of the Lagrange neuron is fed into a neuron with a linear activation function. The definitive outputs are a sparse vector $\hat{\boldsymbol{\beta}} \in \mathbb{R}^p$ containing N_z nonzero components, the *feature selector* together with a scalar $\hat{\mu}$, respectively.

Simulation results demonstrate that the neuronal LPNN-FS model competes with or outperforms diverse feature selection methods in the few-shot high-dimensional setting. These capabilities are attributed to the combination of the theoretical strengths of LPNN-FS blended with the computational efficiency of neuronal models. The following theorem, with its proof to appear in a forthcoming paper, formalizes the proficiency of the neurodynamic system.

Theorem 1. *Assume that $\mathbf{X}^\top\mathbf{X}$ is positive definite. The equilibrium point $(\tilde{\beta}_e^{(k)}, \tilde{\mu}_e^{(k)})$ of the neurodynamic system (9) is asymptotically stable. Moreover, the sparse vector $\tilde{\beta}_e^{(k)}$ is an optimal solution to the $\mathcal{P}_k$-LPNN model (7).*

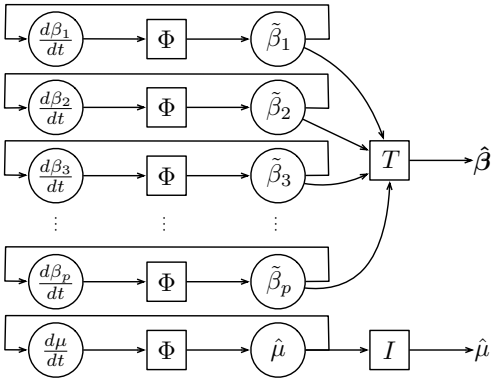

Fig. 1: Implementation of the LPNN-FS model.

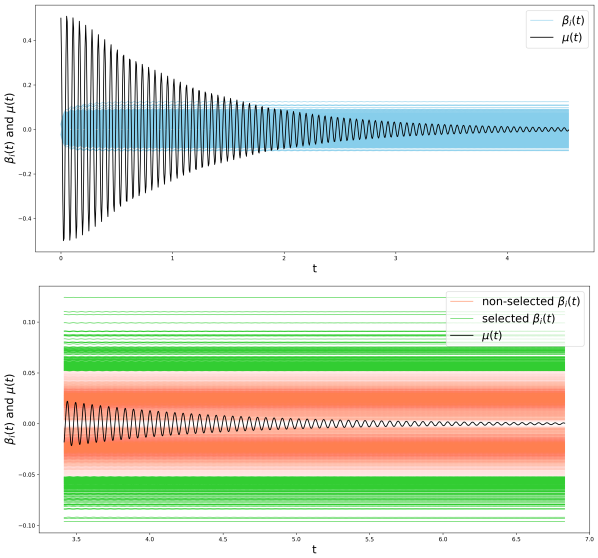

Fig. 2: Example of dynamics of the LPNN-FS model given by Equations (9). (Top) Dynamics of the variable and Lagrange neurons $\beta_i(t)$ and $\mu(t)$, respectively. (Bottom) A zoom-in on the converging part of the dynamics. The N_z largest components among the β_i (green) are selected by the threshold neuron T and set to 1 in the feature selector $\hat{\boldsymbol{\beta}}$, while the remaining components (orange) are set to 0 in $\hat{\boldsymbol{\beta}}$.

IV. Datasets

A. Synthetic Dataset

We consider a synthetic dataset to evaluate the performance of our proposed method. The synthetic dataset consists of a feature matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$, where n is the number of observations and p is the feature dimension, with $n \ll p$, as well as binary labels $\mathbf{y} \in \{-1, +1\}^n$. For a given $p \in \mathbb{N}$, we set $n = 0.1p$, and generate the feature matrix $\mathbf{X}$ and label vector $\mathbf{y}$ using the following steps:

(1) Compute the number of relevant features $\bar{p} := 3.5\left(\frac{n}{\ln(p)}\right)$, where the choice of $\bar{p} \in O\left(\frac{n}{\ln(p)}\right)$ is justified in [21], and the constant 3.5 is determined empirically;

(2) Generate a matrix of relevant features $\bar{\mathbf{X}} \in \mathbb{R}^{n\times\bar{p}}$, with entries sampled independently from the normal distribution $\mathcal{N}(0, 1)$;

(3) Generate a matrix of redundant features $\tilde{\boldsymbol{X}} \in \mathbb{R}^{n\times(p-\bar{p})}$, where each column of $\tilde{\boldsymbol{X}}$ is a random linear combination of the columns of $\bar{\boldsymbol{X}}$;

(4) Concatenate the matrices $\bar{\boldsymbol{X}}$ and $\tilde{\boldsymbol{X}}$, shuffle their columns to construct the feature matrix $\mathbf{X} = \text{Shuffle}\left[\bar{\boldsymbol{X}}, \tilde{\boldsymbol{X}}\right]$, and ensure that $\mathbf{X}$ is of full rank;

(5) Generate the binary label vector $\mathbf{y} \in \{-1, +1\}^n$ by computing scores as the dot product of the relevant features $\bar{\mathbf{X}}$ and a random coefficient vector, and

assign $+1$ to observations with scores greater than or equal to the mean, and -1 to the other observations.

The feature selection and evaluation process, using our LPNN-FS method, is detailed in Section V-A. For this study, the parameter for the k-approximate function is set to $k = 1000$. The parameters for constructing the synthetic dataset are summarized in Table I.

Parameters	Settings
p	$[1000, 2000, 3000, 4000]$
n	10% of p
$\bar{p}$	$3.5\left(\frac{n}{\ln(p)}\right)$
N_z, η	$[1.25\%, 2.5\%, 3.75\%, 5\%]$ of p
k	1000

TABLE I: Parameters used in the construction of the synthetic dataset.

B. Real Datasets

We also consider two real-world benchmark datasets for high-dimensional feature selection problems. These datasets, *DLBCL* and *Prostate*, are publicly available here. Their high dimensionality ($p \gg n$), small sample sizes, and class imbalance make them particularly challenging for feature selection tasks. Before applying the feature selection and evaluation process described in Section V-A, the datasets are normalized. The statistics of the two datasets are summarized in Table II.

Dataset	# Features (p)	# Observations (n)	# Classes
DLBCL	5469	77	2
Prostate	12600	102	2

TABLE II: Statistics of the *DLBCL* and *Prostate* datasets.

V. Results

A. Experimental Setup

We evaluate the proposed LPNN-FS method on the datasets described in Section IV. For this purpose, we employ a 10-fold cross-validation (CV) approach, where 9 folds are used for training and 1 fold is reserved for testing. For each CV iteration, we perform a *feature selection (FS)* step followed by a *feature evaluation (FE)* step.

The feature selection methods (*FS*) considered are as follows:

- **Full and Random features (full, random):** As baselines, we consider the cases where all features are retained (full) or N_z out of the p features are randomly selected uniformly (random).
- ***k*-best selection (k-best, k-best (MI)):** These two supervised methods rank features based on feature-to-feature and feature-to-output correlations, using either the statistical ANOVA F-test (k-best) or the mutual information score (k-best (MI)). The top N_z features with the highest scores are then selected.
- **Least Absolute Shrinkage and Selection Operator (LASSO):** The classical LASSO method is formulated in its unconstrained form. The involved regularization parameter, say $\alpha > 0$, controls the sparsity of the solution, and is set to 1 by default.
- **Sparse Principal Component Analysis (sparse PCA):** Since PCA is inappropriate in our few-shot high-dimensional context, sparse PCA was selected to extend the PCA scope of application.
- **Proposed LPNN-FS method (LPNN-FS):** The LPNN-FS model described in Section III is applied. When the solution $\tilde{\boldsymbol{\beta}} \in \mathbb{R}^p$ of the LPNN-FS neurodynamics (9) is obtained, only its predefined N_z largest components are taken as the features to be selected (see Figure 2).

For the LASSO, k-best, k-best (MI), and sparse PCA methods, we utilize the implementations from the Python scikit-learn library. For FE, the selected features are evaluated by training and testing the following standard ML models on the reduced datasets containing only the N_z selected features: ***k*-Nearest Neighbors (KNN)**, **Logistic Regression (LR)**, and **Gaussian Naive Bayes classifier (NBC)**. The F1-score and balanced accuracy are used as evaluation metrics, calculated on the test sets. For these models, the scikit-learn implementation with default hyperparameters is applied.

The 10-fold CV methodology involves performing the FS and FE steps. One (FE) step consists of training a specific classifier after reducing the number of features. To ensure robustness, we repeat this entire process 10 times with different random seeds, resulting in a total of 100 independent runs per dataset.

B. Synthetic Dataset

We evaluate four synthetic datasets with a total number of features $p \in \{1000, 2000, 3000, 4000\}$, as described in Section IV-A. For each dataset, we apply the LASSO, k-best, k-best (MI), and LPNN-FS methods to select N_z relevant features, where N_z corresponds to 1.25%, 2.5%, 3.75%, and 5% of p. Due to its excessive computational time, we did not include the results for sparse PCA. To assess the performance of the feature selection methods, the reduced datasets composed of the selected features and associated targets are passed to the KNN, LR, and NBC models (see Section V-A for further details).

The F1 score and balanced accuracy associated with the LPNN-FS method are reported in Table III. The highest scores are consistently achieved using LR, indicating a strong complementarity between the LPNN-FS method and the LR model. The optimal results

are obtained by selecting 5%, 5%, 3.75%, and 5% of the p features, which correspond to 50, 100, 113, and 200 selected features for $p = 1000$, 2000, 3000, and 4000, respectively. This aligns with the synthetic dataset construction involving $\bar{p} \approx 3.5\left(\frac{n}{\ln(p)}\right)$ relevant features, which translate to approximately 50, 92, 130, and 170 relevant features for $p = 1000$, 2000, 3000, and 4000, respectively.

Since the random method performed poorly and the k-best (MI) method yielded similar results to k-best, we limit our comparison of the LPNN-FS technique to LASSO and k-best. The results are illustrated in Figures 3–6. For $p = 1000$ and $p = 4000$, LPNN-FS consistently outperforms LASSO and k-best across all selected feature configurations (N_z) and downstream models (KNN, LR, and NBC). For $p = 2000$ and $p = 3000$, LPNN-FS underperforms LASSO and k-best in minority of configurations. The method however excels when $N_z = 1.25\%$ or 2.5%, demonstrating superior performance with fewer features selected features.

Overall, LPNN-FS surpasses LASSO and k-best in 85.4% of cases and achieves improvements of up to 12% over other methods.

p = 1000	**1.25%**	**2.5%**	**3.75%**	**5%**
KNN	0.716 / 0.742	0.745 / 0.768	0.738 / 0.763	0.75 / 0.775
LR	0.731 / 0.752	0.739 / 0.758	0.746 / 0.768	**0.759 / 0.778**
NBC	0.725 / 0.744	0.751 / 0.771	0.735 / 0.757	0.749 / 0.773
p = 2000	**1.25%**	**2.5%**	**3.75%**	**5%**
KNN	0.696 / 0.709	0.717 / 0.73	0.713 / 0.725	0.72 / 0.734
LR	0.737 / 0.746	0.797 / 0.804	0.796 / 0.804	**0.819 / 0.828**
NBC	0.74 / 0.749	0.803 / 0.811	0.779 / 0.789	0.798 / 0.808
p = 3000	**1.25%**	**2.5%**	**3.75%**	**5%**
KNN	0.695 / 0.702	0.749 / 0.758	0.731 / 0.737	0.707 / 0.716
LR	0.791 / 0.797	0.79 / 0.796	**0.787 / 0.795**	0.779 / 0.786
NBC	0.764 / 0.77	0.77 / 0.776	0.738 / 0.746	0.739 / 0.747
p = 4000	**1.25%**	**2.5%**	**3.75%**	**5%**
KNN	0.725 / 0.733	0.732 / 0.74	0.742 / 0.749	0.726 / 0.734
LR	0.777 / 0.782	0.816 / 0.82	0.829 / 0.833	**0.834 / 0.839**
NBC	0.78 / 0.785	0.816 / 0.82	0.818 / 0.823	0.819 / 0.823

TABLE III: F1 score and balanced accuracy achieved by the LPNN-FS method followed by the KNN, LR, and NBC models for synthetic datasets with different number of features $p = 1000, 2000, 3000, 4000$. The LPNN-FS model with various percentages of selected features $N_z = 1.25\%, 2.5\%, 3.75\%, 5\%$ of p is evaluated.

C. Real Datasets

The performance of LPNN-FS on the DLBCL and Prostate datasets is reported in Figure 7. LPNN-FS is tested in combination with 3 different classifiers (NBC, KNN and LR) and 4 percentages of selected features ($N_z = 1.25\%, 2.5\%, 5\%, 10\%$), which amounts to 12 configurations. For the DLBCL dataset, LPNN-FS outperforms all competing methods (random, k-best, k-best

	KNN	**LR**	**NBC**
random	0.557 / 0.567	0.785 / 0.79	0.663 / 0.67
full	0.588 / 0.6	0.818 / 0.823	0.745 / 0.751
k-best	0.689 / 0.697	0.791 / 0.798	0.742 / 0.748
k-best (MI)	0.627 / 0.635	0.798 / 0.803	0.737 / 0.745
LASSO	0.716 / 0.724	0.777 / 0.782	0.76 / 0.767
LPNN-FS	**0.726 / 0.734**	**0.834 / 0.839**	**0.819 / 0.823**

TABLE IV: F1 scores and balanced accuracies achieved by all feature selection methods, followed by the KNN, LR, and NBC models, for synthetic datasets with $p = 4000$ and $N_z = 5\%$ of p.

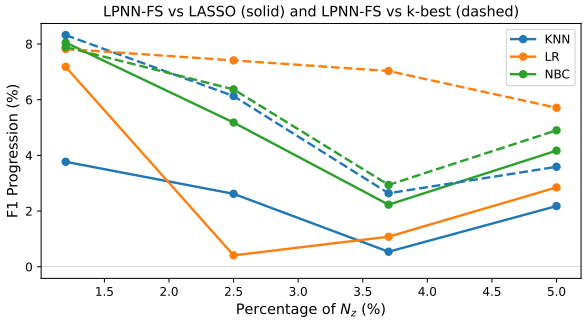

Fig. 3: Results for the synthetic dataset with $p = 1000$. Improvement in F1 score (%) achieved by LPNN-FS compared to LASSO (solid line) and k-best (dashed line). All percentages are positive, indicating that LPNN-FS consistently outperforms LASSO and k-best in this case.

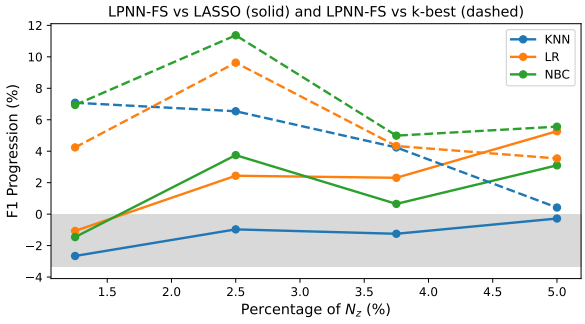

Fig. 4: Results for the synthetic dataset with $p = 2000$. The shaded area indicates configurations where LPNN-FS underperforms compared to LASSO or k-best.

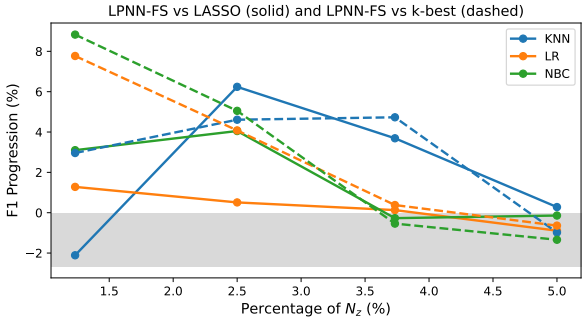

Fig. 5: Results for synthetic dataset with $p = 3000$.

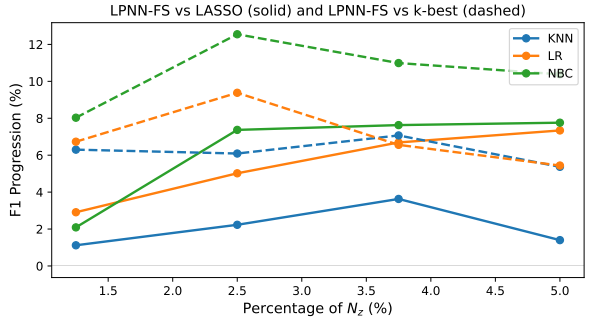

Fig. 6: Results for synthetic dataset with $p = 4000$.

(MI), and sparse PCA) in 8 out of 12 configurations. The LASSO method with default hyperparameter $\alpha = 1$ failed to select any feature. However, reducing α to 0.001, which involves performing an additional hyperparameter search, generally produced slightly better results than the LPNN-FS model. The performance trend of the LR model suggests potential for improvement by selecting more than 5% of the features, a strategy that does not apply to the other models (KNN and NB). Across all three models, the highest scores are consistently achieved using LPNN-FS.

For the Prostate dataset, LPNN-FS outperforms competing methods in 7 out of 12 configurations, which remains a majority. For all models, the best results are consistently obtained when selecting 1.25% or 2.5% of the features, highlighting the ineffectiveness of selecting too many features for this particular dataset. Furthermore, across all three models, the highest scores are again achieved using LPNN-FS.

The balanced accuracy achieved by the NBC, KNN and LR classifiers in combination with the LPNN-FS feature selection method is reported in Figure 8. For both datasets, the particularly strong scores obtained by the LPNN-FS and LR highlight the complementarity between these two methods. Furthermore, the F1 score achieved by LPNN-FS together with the NBC, KNN and LR classifiers, reported in Table V, follows a similar trend.

DLBCL	**1.25%**	**2.5%**	**5%**	**10%**
KNN	0.798	0.832	0.857	0.79
LR	0.816	0.856	0.894	**0.942**
NBC	0.488	0.7	0.648	0.57
Prostate	**1.25%**	**2.5%**	**5%**	**10%**
KNN	0.895	0.896	0.876	0.85
LR	**0.928**	0.903	0.875	0.842
NBC	0.778	0.711	0.615	0.492

TABLE V: F1 scores achieved by LPNN-FS using different percentages of selected features N_z, for the DLBCL and Prostate datasets.

VI. Conclusion

This paper introduces LPNN-FS, a novel feature selection framework that combines the theoretical strengths of smoothed LASSO formulation with the computational efficiency of Lagrange Programming Neural Networks. The integration of a smoothing approximation and a neurodynamic approach enables LPNN-FS to achieve high accuracy while maintaining computational efficiency. In the context of high-dimensional datasets containing a limited number of observations, LPNN-FS demonstrates consistent performance on both synthetic and real-world benchmarks, often outperforming traditional methods such as LASSO, k-best and sparse PCA.

For future work, we plan to integrate physics-informed neural networks (PINNs) into our method, enabling them to learn how to solve the LPNN-FS neurodynamic system directly, rather than relying on numerical analysis schemes. This approach has the potential to provide rapid solutions for the system's dynamics, paving the way for real-time, high-dimensional, few-shot feature selection. Furthermore, drawing inspiration from the LassoNet model [22], the LPNN-FS method is expected to be extended into an embedded approach by incorporating it into an echo state network (ESN) architecture, where the reservoir serves as the feature compressor.

Overall, this study bridges the domains of compressive sampling (CS) and feature selection (FS), opening the door to the development of CS-based techniques for hard FS settings.

Acknowledgment

The research was done with institutional support RVO: 67985807 and partially supported by the Czech Science Foundation grant GA25-15490S.

References

[1] E. P. Xing, M. I. Jordan, R. M. Karp, *et al.*, "Feature selection for high-dimensional genomic microarray data," in *Icml*, vol. 1, pp. 601–608, Citeseer, 2001.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[4] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[5] M. Dash and H. Liu, "Feature selection for classification," *Intelligent data analysis*, vol. 1, no. 1-4, pp. 131–156, 1997.

[6] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[7] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 58, no. 1, pp. 267–288, 1996.

[8] D. A. A. Gnana, S. A. A. Balamurugan, and E. J. Leavline, "Literature review on feature selection methods for high-dimensional data," *International Journal of Computer Applications*, vol. 136, no. 1, pp. 9–17, 2016.

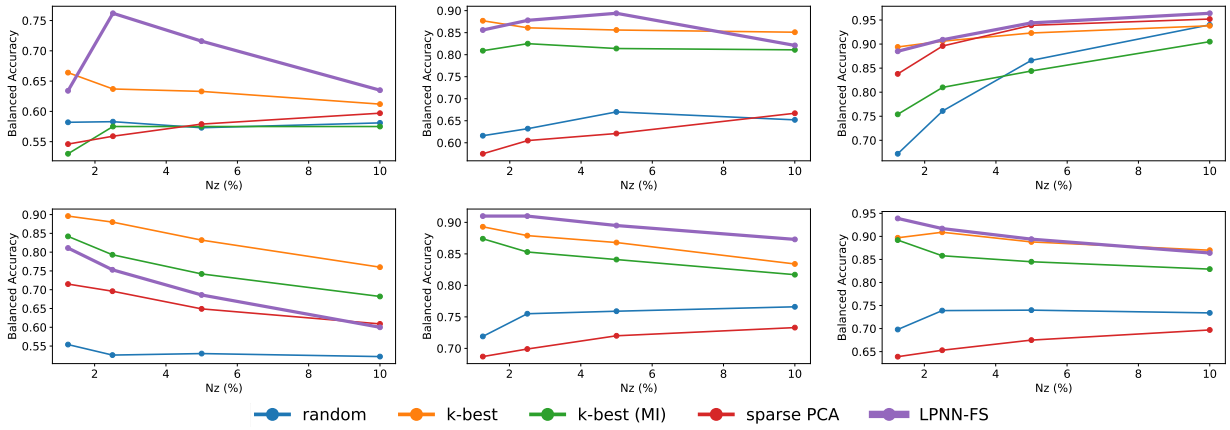

Fig. 7: Results for the DLBCL (top) and Prostate (bottom) datasets. Comparisons of the different feature selection methods (random, k-best, k-best (MI), sparse PCA and LPNN-FS) followed by three classifiers (NBC, KNN, LR). Balanced accuracy for NBC (left), KNN (middle) and LR (right). For the DLBCL and the Prostate datasets, LPNN-FS (purple curve) outperforms all other methods in 8 out of 12 and 7 out of 12 configurations, respectively.

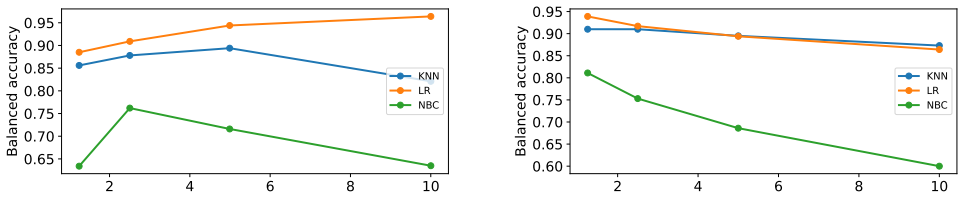

Fig. 8: Balanced accuracy of NBC, KNN and LR classifiers using LPNN-FS as feature selector for the DLBCL (left) and the Prostate (right) datasets.

[9] S. Foucart, H. Rauhut, S. Foucart, and H. Rauhut, *An invitation to compressive sensing*. Springer, 2013.

[10] A. Zeglaoui, A. Houmia, M. Mejai, and R. Aloui, "Lpnn-based approach for lasso problem via a sequence of regularized minimizations," *International Journal of Adaptive Control and Signal Processing*, vol. 35, no. 9, pp. 1842–1859, 2021.

[11] H. Wang, C. M. Lee, R. Feng, and C. S. Leung, "An analog neural network approach for the least absolute shrinkage and selection operator problem," *Neural Comput. Appl.*, vol. 29, no. 9, pp. 389–400, 2018.

[12] J. Nocedal and S. J. Wright, *Quadratic Programming*, ch. 16, pp. 448–492. New York, NY: Springer New York, 2006.

[13] S. Brahim Belhaouari, M. B. Shakeel, A. Erbad, Z. Oflaz, and K. Kassoul, "Bird's eye view feature selection for high-dimensional data," *Scientific Reports*, vol. 13, no. 1, p. 13303, 2023.

[14] N. Zhantileuov and S. Ospanov, "A comparative study of supervised machine learning and deep learning techniques with feature selection methods for classifying parkinson's disease based on speech impairments," in *2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST)*, pp. 124–129, IEEE, 2024.

[15] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[16] Y. Cai, T. Huang, L. Hu, X. Shi, L. Xie, and Y. Li, "Prediction of lysine ubiquitination with mrmr feature selection and analysis," *Amino acids*, vol. 42, pp. 1387–1395, 2012.

[17] M. Ayyanar, S. Jeganathan, S. Parthasarathy, V. Jayaraman, and A. R. Lakshminarayanan, "Predicting the cardiac diseases using selectkbest method equipped light gradient boosting machine," in *2022 6th International conference on trends in electronics and informatics (ICOEI)*, pp. 117–122, IEEE, 2022.

[18] J.-B. H. Urruty and C. Lemaréchal, *Convex analysis and minimization algorithms*. Springer-Verlag, 1993.

[19] M. Schmidt, G. Fung, and R. Rosales, "Fast optimization methods for l1 regularization: A comparative study and two new approaches," in *Machine Learning: ECML 2007: 18th European Conference on Machine Learning, Warsaw, Poland, September 17-21, 2007. Proceedings 18*, pp. 286–297, Springer, 2007.

[20] Y. Zhao, X. He, T. Huang, J. Huang, and P. Li, "A smoothing neural network for minimization l1-lp in sparse signal reconstruction with measurement noises," *Neural Networks*, vol. 122, pp. 40–53, 2020.

[21] E. Candes and T. Tao, "The dantzig selector: Statistical estimation when p is much larger than n," 2007.

[22] I. Lemhadri, F. Ruan, L. Abraham, and R. Tibshirani, "Lassonet: A neural network with feature sparsity," *J. Mach. Learn. Res.*, vol. 22, pp. 127:1–127:29, 2021.