CAUCHY – Jurnal Matematika Murni dan Aplikasi
Volume 11 (1) (2026), Pages 820-841
p-ISSN: 2086-0382; e-ISSN: 2477-3344

A Damped Hessian-Free Newton–Conjugate Gradient Method for Weighted Multiclass Neural Classification

Andy Irawan[1], Zainal Abidin[1], and Mohammad Jamhuri[2*]

[1] *Department of Informatics Engineering, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang, Indonesia*
[2] *Department of Mathematics, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang, Indonesia*

Abstract

This study presents a deterministic damped Hessian-free Newton–CG method for weighted multiclass neural classification. The method is built from a weighted categorical cross-entropy objective, a damped local quadratic model, and a matrix-free curvature representation through Hessian–vector products. The search direction is computed by an inexact conjugate gradient solve, while Armijo backtracking and adaptive damping are used to improve stability. The method is implemented for the classification of academic predicate categories using preprocessed student data with mixed categorical and numerical features. Its numerical behavior is compared with SGD with momentum, RMSProp, and Adam under the same loss, initialization, and network architecture. The proposed method is computationally feasible, attains the best overall weighted test-set performance among the compared methods, and exhibits a distinct optimization trajectory driven by curvature-informed updates. These results show that a damped Hessian-free formulation provides a mathematically transparent, reproducible, and practically competitive framework for second-order optimization in multiclass neural classification.

Keywords: conjugate gradient; Hessian-free optimization; multiclass classification; neural networks; second-order methods

1. Introduction

Optimization is central to neural-network training. In multiclass classification, empirical risk minimization is usually carried out by first-order methods such as SGD, momentum-based variants, RMSProp, and Adam because they are simple to implement and scale well in practice [1–5]. Their updates, however, are driven mainly by gradient information and can therefore be sensitive to ill-conditioning, local curvature variation, and unfavorable geometry of the objective [1, 6]. This motivates continued interest in second-order and curvature-informed methods, particularly in settings where more structured search directions may improve optimization behavior [7, 8].

A standard second-order viewpoint is based on a local quadratic model. Let $\theta \in \mathbb{R}^p$ denote the parameter vector and let $\mathcal{L}(\theta)$ be the empirical loss. Around the current iterate, one considers

$$m_\theta(d) = \mathcal{L}(\theta) + \nabla\mathcal{L}(\theta)^\top d + \frac{1}{2} d^\top \nabla^2 \mathcal{L}(\theta) d, \tag{1}$$

*Corresponding author. E-mail: m.jamhuri@live.com

Submitted: January 07, 2026 Reviewed: February 13, 2026 Accepted: March 28, 2026
DOI: https://doi.org/10.18860/cauchy.v11i1.40243

where d is a search direction. Minimization of Eq. (1) leads formally to a Newton step [6]. For neural networks, however, explicitly forming and inverting the Hessian is often too expensive. Hessian-free approaches address this difficulty by using Hessian–vector products instead of the full Hessian matrix [7, 9–11].

A practical variant is damped Hessian-free optimization, in which the Newton system is regularized as

$$(H(\theta) + \lambda I)d = -g(\theta), \tag{2}$$

with $g(\theta) = \nabla\mathcal{L}(\theta)$, $H(\theta) = \nabla^2\mathcal{L}(\theta)$, and $\lambda > 0$ a damping parameter. The term λI improves numerical robustness when the local curvature is poorly conditioned or indefinite. When products with $H(\theta) + \lambda I$ are available, Eq. (2) can be solved approximately by conjugate gradient (CG), producing a Hessian-free Newton–CG method suitable for large-scale settings [6, 9, 12].

Despite the maturity of Hessian-free ideas in numerical optimization, many applied machine-learning studies still treat the optimizer largely as a black box. The mathematical structure of the objective, the role of damping, the matrix-free curvature construction, and the relation between predicted and actual decrease are often left implicit. From a computational-mathematics viewpoint, however, these are precisely the ingredients needed to interpret an algorithm as a genuine second-order method rather than as a heuristic update rule.

This work adopts that perspective for multiclass neural classification on educational data. The task is to predict academic predicate categories from student records containing both categorical and numerical features. Rather than using the dataset merely as a benchmarking vehicle, we use it to study a mathematically explicit and reproducible damped Hessian-free Newton–CG method under a weighted categorical cross-entropy objective. The resulting method is compared with three standard first-order optimizers: SGD with momentum, RMSProp, and Adam.

The present work is also related to recent curvature-based studies for classification, including inexact generalized Gauss–Newton–CG for binary cross-entropy minimization and Gauss–Newton-based neural-network optimization for classification tasks [13–15]. Relative to those studies, the present work moves to weighted multiclass neural classification and adopts a damped Hessian-free Newton–CG formulation.

The main contributions of this study are threefold. First, we formulate a deterministic damped Hessian-free scheme for weighted multiclass neural classification, including the weighted empirical objective, the damped quadratic model, and the matrix-free Hessian–vector product construction. Second, we present a reproducible implementation based on inexact CG, adaptive damping, Armijo backtracking, and a descent safeguard. Third, we provide a numerical study showing how this curvature-informed method behaves relative to SGD with momentum, RMSProp, and Adam under the same loss, initialization, and network architecture.

The remainder of the paper is organized as follows. Section 2 presents the mathematical formulation, computational properties, and experimental protocol. Section 3 reports and discusses the numerical results. Section 4 concludes the paper and outlines future work.

2. Methods

This section presents the weighted multiclass classification model, the damped Hessian-free Newton–CG method, several basic computational properties of the algorithm, and the experimental protocol. The goal is to keep the mathematical formulation, implementation, and reported diagnostics tightly aligned.

2.1. Problem Setting and Data Representation

Let

$$\mathcal{D}_{\text{raw}} = \{(\xi_i, c_i)\}_{i=1}^{N}$$

denote the original dataset, where ξ_i is the raw feature record of the ith observation and

$$c_i \in \{1, 2, 3, 4\}$$

is the corresponding academic predicate label. The class ordering used in this study is

$$1 \leftrightarrow \text{Cukup (Sufficient)},$$
$$2 \leftrightarrow \text{Memuaskan (Satisfactory)},$$
$$3 \leftrightarrow \text{Sangat Memuaskan (Highly Satisfactory)},$$
$$4 \leftrightarrow \text{Cumlaude (Cum laude)}.$$

The predictors consist of categorical and numerical variables. After cleaning, a preprocessing map

$$T : \mathcal{X}_{\text{raw}} \to \mathbb{R}^d$$

is applied, where categorical variables are one-hot encoded and numerical variables are standardized. Each observation is therefore represented by

$$x_i = T(\xi_i) \in \mathbb{R}^d.$$

For the multiclass formulation, labels are also written in one-hot form:

$$y_i = (y_{i1}, y_{i2}, y_{i3}, y_{i4})^\top \in \{0,1\}^4, \qquad \sum_{k=1}^{4} y_{ik} = 1,$$

with

$$y_{ik} = \begin{cases} 1, & k = c_i, \\ 0, & k \neq c_i. \end{cases}$$

Thus the processed dataset used by the optimizer is

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}, \qquad x_i \in \mathbb{R}^d,\ y_i \in \{0,1\}^4.$$

2.2. Deterministic Neural Network Classifier

Let $\theta \in \mathbb{R}^p$ denote the vector of all trainable parameters. The classifier used in this study is a fully connected feedforward neural network without dropout, so the empirical objective is deterministic once the data and parameters are fixed.

The network consists of three hidden layers with ReLU activation and one softmax output layer. Writing the parameter blocks as

$$\theta = (W_1, b_1, W_2, b_2, W_3, b_3, W_4, b_4),$$

the forward propagation is

$$\begin{aligned} h^{(1)}(x) &= \sigma(W_1 x + b_1), \\ h^{(2)}(x) &= \sigma(W_2 h^{(1)}(x) + b_2), \\ h^{(3)}(x) &= \sigma(W_3 h^{(2)}(x) + b_3), \\ z_\theta(x) &= W_4 h^{(3)}(x) + b_4, \end{aligned}$$

where $\sigma(u) = \max\{0, u\}$ is the ReLU function applied componentwise [16, 17].

The class probabilities are

$$p_{\theta,k}(x) = \frac{\exp(z_{\theta,k}(x))}{\sum_{j=1}^{4} \exp(z_{\theta,j}(x))}, \qquad k = 1,2,3,4. \tag{3}$$

The class probabilities are obtained through the softmax mapping in Eq. (3). These probabilities define the multiclass predictive distribution used throughout the paper, and they enter directly into the weighted loss function introduced in Eq. (4). Hence

$$p_\theta(x) = \big(p_{\theta,1}(x), p_{\theta,2}(x), p_{\theta,3}(x), p_{\theta,4}(x)\big)^\top \in \mathbb{R}^4.$$

Remark 1. Because ReLU is not twice differentiable at zero, second-order expressions are understood on regions where the activation pattern is fixed. In automatic differentiation, Hessian–vector products are evaluated almost everywhere, which is sufficient for the present numerical method [10].

2.3. Weighted Empirical Objective

All compared methods are trained under the same weighted categorical cross-entropy objective. Let

$$\omega = (\omega_1, \omega_2, \omega_3, \omega_4)^\top \in \mathbb{R}^4_{++}$$

be the vector of positive class weights computed from the training data. Class weighting is used to mitigate the imbalance of the target distribution [18, 19].

Definition 1. For each sample (x_i, y_i), define the weighted categorical cross-entropy loss by

$$\ell_i(\theta) = -\sum_{k=1}^{4} \omega_k \, y_{ik} \log p_{\theta,k}(x_i). \tag{4}$$

Because y_i is one-hot encoded,

$$\ell_i(\theta) = -\omega_{c_i} \log p_{\theta,c_i}(x_i).$$

Definition 2. The weighted empirical risk is

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \ell_i(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{4} \omega_k \, y_{ik} \log p_{\theta,k}(x_i), \tag{5}$$

and the optimization problem is

$$\min_{\theta \in \mathbb{R}^p} \mathcal{L}(\theta). \tag{6}$$

The sample-wise weighted loss in Eq. (4) induces the weighted empirical risk in Eq. (5). Accordingly, the optimization problem considered in this work is the minimization problem stated in Eq. (6). This is the same loss used for all compared optimizers, so the observed numerical differences can be interpreted more directly as consequences of the optimization mechanism rather than differences in the loss function itself. Let

$$g(\theta) = \nabla \mathcal{L}(\theta) \qquad \text{and} \qquad H(\theta) = \nabla^2 \mathcal{L}(\theta)$$

denote the gradient and Hessian of the weighted empirical risk, wherever defined [1, 19, 20].

2.4. Damped Quadratic Model and Matrix-Free Curvature

At the current iterate θ, the objective is approximated locally by a damped quadratic model.

Definition 3. For a search direction $d \in \mathbb{R}^p$ and a damping parameter $\lambda > 0$, define

$$m_\theta^{(\lambda)}(d) = \mathcal{L}(\theta) + g(\theta)^\top d + \frac{1}{2} d^\top (H(\theta) + \lambda I) d, \tag{7}$$

where I is the identity matrix.

The corresponding damped Newton system is

$$(H(\theta) + \lambda I) d = -g(\theta). \tag{8}$$

Definition 4. For $\theta \in \mathbb{R}^p$ and $\lambda > 0$, define the damped curvature operator

$$B_{\theta,\lambda} = H(\theta) + \lambda I. \tag{9}$$

The damped local model in Eq. (7) leads to the regularized Newton system in Eq. (8). The shifted operator in Eq. (9) plays a central role in the proposed method, since it is this operator—rather than the undamped Hessian alone—that determines the curvature-informed search direction.

The full Hessian is never formed explicitly. Instead, the method uses Hessian–vector products.

Lemma 1. *Assume that $\mathcal{L}$ is twice differentiable at θ. Then for every $v \in \mathbb{R}^p$,*

$$H(\theta)v = \nabla_\theta\big(g(\theta)^\top v\big). \tag{10}$$

Proof. Since

$$g(\theta)^\top v = \sum_{j=1}^{p} \frac{\partial \mathcal{L}(\theta)}{\partial \theta_j} v_j,$$

differentiating with respect to θ_i gives

$$\frac{\partial}{\partial \theta_i}\big(g(\theta)^\top v\big) = \sum_{j=1}^{p} \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta_i \partial \theta_j} v_j.$$

Hence

$$\nabla_\theta\big(g(\theta)^\top v\big) = H(\theta)v.$$

□

The identity in Eq. (10) is the basis of Hessian-free optimization and enables efficient second-order computation through automatic differentiation [7, 9–11]. Using Eq. (10), the action of the damped curvature operator can be evaluated in matrix-free form by Eq. (11), so the method never needs to assemble the full Hessian explicitly. Therefore,

$$B_{\theta,\lambda}v = H(\theta)v + \lambda v. \tag{11}$$

2.5. Vectorized Parameter Representation

In implementation, the network parameters are stored as tensors of different shapes. For second-order optimization, these tensors are flattened and concatenated into a single vector:

$$\theta = (\theta^{(1)}, \ldots, \theta^{(M)}) \longleftrightarrow \mathrm{vec}(\theta) \in \mathbb{R}^p.$$

Likewise, any search direction $d \in \mathbb{R}^p$ is reshaped back into the original tensor structure before being applied. This representation allows the gradient, Hessian–vector product, and CG iteration to be written in a unified Euclidean form.

2.6. Inexact Damped Newton–CG Iteration

At outer iteration t, let

$$\theta_t \in \mathbb{R}^p, \qquad g_t = \nabla\mathcal{L}(\theta_t), \qquad B_t = H(\theta_t) + \lambda_t I.$$

At outer iteration t, the proposed method computes an approximate solution of the damped linear system in Eq. (12) by conjugate gradient, using only matrix-free products with the operator B_t. The resulting vector d_t is then interpreted as an inexact curvature-informed search direction.

$$B_t d_t = -g_t \tag{12}$$

Starting from $d^{(0)} = 0$, the standard CG recursions are

$$\begin{aligned} r^{(0)} &= -g_t, \\ p^{(0)} &= r^{(0)}, \\ \alpha_j^{\mathrm{CG}} &= \frac{(r^{(j)})^\top r^{(j)}}{(p^{(j)})^\top B_t p^{(j)}}, \\ d^{(j+1)} &= d^{(j)} + \alpha_j^{\mathrm{CG}} p^{(j)}, \\ r^{(j+1)} &= r^{(j)} - \alpha_j^{\mathrm{CG}} B_t p^{(j)}, \\ \beta_j^{\mathrm{CG}} &= \frac{(r^{(j+1)})^\top r^{(j+1)}}{(r^{(j)})^\top r^{(j)}}, \\ p^{(j+1)} &= r^{(j+1)} + \beta_j^{\mathrm{CG}} p^{(j)}. \end{aligned}$$

After a finite number of inner iterations, the final CG iterate is used as an inexact curvature-informed search direction [6, 12].

For a candidate direction d_t, the model-based decrease is measured by the predicted reduction in Eq. (13). To globalize the iteration, the method applies Armijo backtracking: the trial step length is reduced according to Eq. (14) until the sufficient decrease condition in Eq. (15) is satisfied or a minimum threshold is reached.

$$\mathrm{pred}_t = -\left(g_t^\top d_t + \frac{1}{2} d_t^\top B_t d_t \right). \tag{13}$$

The method then applies an Armijo backtracking line search. Starting from an initial trial step length $\alpha_0 > 0$, the step size is reduced by

$$\alpha \leftarrow \beta\alpha, \qquad 0 < \beta < 1, \tag{14}$$

until

$$\mathcal{L}(\theta_t + \alpha d_t) \le \mathcal{L}(\theta_t) + c_1 \alpha\, g_t^\top d_t, \qquad 0 < c_1 < 1, \tag{15}$$

or a minimum step threshold is reached [6, 21].

Once a step is accepted, the parameters are updated by Eq. (16). The observed decrease in the objective is then measured by the actual reduction in Eq. (17), while the agreement between the local model and the nonlinear objective is quantified by the ratio in Eq. (18). This ratio is used to update the damping parameter through the rule in Eq. (19): good agreement makes the method more Newton-like, whereas poor agreement leads to stronger damping.

$$\theta_{t+1} = \theta_t + \alpha_t d_t. \tag{16}$$

The actual reduction is

$$\mathrm{ared}_t = \mathcal{L}(\theta_t) - \mathcal{L}(\theta_{t+1}), \tag{17}$$

and the reduction ratio is

$$\rho_t = \frac{\mathrm{ared}_t}{\mathrm{pred}_t}. \tag{18}$$

The damping parameter is updated by

$$\lambda_{t+1} = \begin{cases} \frac{1}{2}\lambda_t, & \rho_t > 0.75, \\ 2\lambda_t, & \rho_t < 0.25, \\ \lambda_t, & \text{otherwise}. \end{cases} \tag{19}$$

Thus, good agreement between predicted and actual reduction leads to weaker damping, whereas poor agreement leads to stronger damping.

Because the system in Eq. (12) is solved only approximately, the computed direction may occasionally fail to be sufficiently reliable. For that reason, the implementation checks whether the candidate direction is descent-like and whether the predicted reduction in Eq. (13) is positive. If either condition fails, the method replaces the candidate direction by the steepest descent safeguard in Eq. (20). The implementation therefore checks whether

$$g_t^\top d_t < 0 \qquad \text{and} \qquad \text{pred}_t > 0.$$

If either condition fails, the direction is replaced by the steepest descent direction

$$d_t = -g_t. \tag{20}$$

The formulation is methodologically close to recent inexact curvature-based approaches for classification, but differs in two ways: it addresses weighted multiclass neural classification rather than binary logistic regression, and it uses a damped Hessian-free Newton–CG mechanism rather than a generalized Gauss–Newton or QR-based Gauss–Newton construction [13–15].

2.7. Basic Computational Properties

The deterministic weighted Hessian-free method has several basic properties that clarify its optimization behavior. The basic properties below explain why the damped operator in Eq. (9), the exact system in Eq. (21), and the globalization mechanism in Eq. (15) together provide a mathematically meaningful optimization framework.

Proposition 1. *Assume that $H(\theta_t)$ is symmetric at the current iterate. If*

$$\lambda_t > -\lambda_{\min}(H(\theta_t)),$$

then

$$B_t = H(\theta_t) + \lambda_t I$$

is positive definite.

Proof. Let μ be any eigenvalue of $H(\theta_t)$ with eigenvector $u \neq 0$. Then

$$B_t u = (H(\theta_t) + \lambda_t I)u = (\mu + \lambda_t)u.$$

Hence the eigenvalues of B_t are $\mu + \lambda_t$. Since

$$\mu + \lambda_t \geq \lambda_{\min}(H(\theta_t)) + \lambda_t > 0,$$

all eigenvalues of B_t are positive. Therefore B_t is positive definite. □

Remark 2. When λ_t is large,

$$H(\theta_t) + \lambda_t I \approx \lambda_t I,$$

so the damped Newton direction behaves approximately like

$$d_t \approx -\frac{1}{\lambda_t} g_t.$$

Thus, strong damping makes the method more gradient-like and conservative.

Proposition 2. *Suppose that B_t is symmetric positive definite and that $g_t \neq 0$. Let $d_t^\star$ be the exact solution of*

$$B_t d_t^\star = -g_t. \tag{21}$$

Then $d_t^\star$ is a strict descent direction for $\mathcal{L}$ at θ_t, i.e.,

$$g_t^\top d_t^\star < 0. \tag{22}$$

Proof. Since B_t is symmetric positive definite, its inverse exists and is also symmetric positive definite. From Eq. (21),

$$d_t^\star = -B_t^{-1} g_t.$$

Therefore

$$g_t^\top d_t^\star = -g_t^\top B_t^{-1} g_t.$$

Because $g_t \neq 0$ and B_t^{-1} is positive definite, one has

$$g_t^\top B_t^{-1} g_t > 0.$$

Hence

$$g_t^\top d_t^\star < 0.$$

□

Thus, when the system in Eq. (21) is solved exactly, the direction characterized by Eq. (22) is guaranteed to be a descent direction.

Corollary 1. *Under the assumptions of the previous proposition, if $d_t^\star \neq 0$, then*

$$\mathrm{pred}_t^\star = -\left(g_t^\top d_t^\star + \frac{1}{2} (d_t^\star)^\top B_t d_t^\star \right) > 0.$$

Proof. Since $B_t d_t^\star = -g_t$, one has

$$g_t^\top d_t^\star = -(d_t^\star)^\top B_t d_t^\star.$$

Substituting this into the definition of $\mathrm{pred}_t^\star$ gives

$$\mathrm{pred}_t^\star = \frac{1}{2} (d_t^\star)^\top B_t d_t^\star.$$

Because B_t is positive definite and $d_t^\star \neq 0$, the right-hand side is strictly positive. □

This shows that, in the exact positive definite setting, the model-based quantity in Eq. (13) is strictly positive.

Proposition 3. *If $g_t \neq 0$ and the safeguard direction*

$$d_t = -g_t$$

is used, then d_t is a strict descent direction for $\mathcal{L}$ at θ_t.

Proof. One has

$$g_t^\top d_t = g_t^\top (-g_t) = -\|g_t\|^2 < 0,$$

provided $g_t \neq 0$. □

Therefore, the safeguard direction in Eq. (20) preserves descent whenever the inexact curvature step is considered unreliable.

Proposition 4. *Assume that $\mathcal{L}$ is differentiable in a neighborhood of θ_t and that d_t is a descent direction, i.e.,*

$$g_t^\top d_t < 0.$$

Then there exists $\bar{\alpha} > 0$ such that the Armijo condition holds for all $\alpha \in (0, \bar{\alpha}]$.

Proof. Since $\mathcal{L}$ is differentiable at θ_t,

$$\mathcal{L}(\theta_t + \alpha d_t) = \mathcal{L}(\theta_t) + \alpha g_t^\top d_t + o(\alpha) \qquad \text{as } \alpha \to 0^+.$$

Because $g_t^\top d_t < 0$ and $0 < c_1 < 1$, the quantity

$$\alpha g_t^\top d_t + o(\alpha)$$

is eventually no larger than

$$c_1 \alpha g_t^\top d_t$$

for sufficiently small $\alpha > 0$. Hence the Armijo inequality holds for all sufficiently small step lengths. □

Hence the sufficient decrease test in Eq. (15) is theoretically compatible with any descent direction generated either by the inexact Newton step or by the safeguard in Eq. (20).

Proposition 5. *If the backtracking procedure accepts a step length $\alpha_t > 0$, then the actual reduction satisfies*

$$\text{ared}_t = \mathcal{L}(\theta_t) - \mathcal{L}(\theta_t + \alpha_t d_t) > 0.$$

Proof. If the Armijo condition holds, then

$$\mathcal{L}(\theta_t + \alpha_t d_t) \leq \mathcal{L}(\theta_t) + c_1 \alpha_t g_t^\top d_t.$$

Since $g_t^\top d_t < 0$ and $c_1 > 0$, the right-hand side is strictly smaller than $\mathcal{L}(\theta_t)$. Therefore

$$\mathcal{L}(\theta_t + \alpha_t d_t) < \mathcal{L}(\theta_t),$$

which implies $\text{ared}_t > 0$. □

In particular, acceptance under Eq. (15) guarantees positivity of the actual reduction defined in Eq. (17).

The reduction ratio in Eq. (18) compares the observed decrease in the nonlinear objective with the decrease predicted by the damped quadratic model in Eq. (13). If ρ_t is close to one, the local model is accurate; if ρ_t is small, stronger damping through Eq. (19) is justified. The method should therefore be interpreted as an *inexact damped Newton–CG method with descent safeguarding*.

From a computational viewpoint, if J_t denotes the number of CG iterations at outer iteration

t, then the dominant cost of one outer iteration consists of one full-batch gradient evaluation, approximately J_t evaluations of the damped Hessian–vector product in Eq. (11), and several additional full-batch loss evaluations during Armijo backtracking. This leads to the per-iteration cost model in Eq. (23). Hence, if $\mathrm{Cost}_{\mathrm{grad}}$ denotes the cost of one full-batch gradient computation, $\mathrm{Cost}_{\mathrm{hvp}}$ the cost of one Hessian–vector product, and $\mathrm{Cost}_{\mathrm{ls}}$ the total cost of line-search tests, then

$$\mathrm{Cost}_t \approx \mathrm{Cost}_{\mathrm{grad}} + J_t\,\mathrm{Cost}_{\mathrm{hvp}} + \mathrm{Cost}_{\mathrm{ls}}. \tag{23}$$

An important advantage of the method is that it avoids explicit storage of the full Hessian. If the parameter vector has dimension p, storing the full Hessian would require $O(p^2)$ memory. By contrast, the matrix-free implementation stores only the parameter tensors, the gradient vector, and a small number of CG-related vectors, so the memory requirement grows essentially linearly with the number of parameters [6, 9].

This trade-off between curvature quality and per-iteration cost is also consistent with recent literature showing that modern second-order methods can be attractive in large-batch settings, provided curvature information is approximated and exploited efficiently [7, 8, 11].

2.8. Experimental Protocol

The dataset consists of student records with categorical and numerical attributes related to demographic background, prior educational experience, language and computer proficiency, and semester grade indicators. The prediction task is to classify each student into one of four academic predicate categories:

$$\{\text{Cukup},\ \text{Memuaskan},\ \text{Sangat Memuaskan},\ \text{Cumlaude}\}.$$

The predictor variables are:

1. gender,
2. school type,
3. boarding-school background,
4. admission pathway,
5. Arabic proficiency,
6. English proficiency,
7. computer proficiency,
8. major,
9. IPS 1,
10. IPS 2,
11. IPS 3,
12. IPS 4.

The preprocessing stage retains the selected predictors and target variable, treats empty strings as missing values, imputes missing categorical entries by the mode, and imputes missing numerical entries by the median. Categorical predictors are then one-hot encoded and numerical predictors are standardized. The preprocessing transformer is fitted on the training subset only and then applied to the validation and test subsets.

The dataset is partitioned by stratified sampling into training, validation, and test subsets with proportions

$$60\%\ \text{training}, \qquad 20\%\ \text{validation}, \qquad 20\%\ \text{test}.$$

A fixed random seed is used throughout, and the label distribution is recorded to verify that the class imbalance pattern is preserved.

All methods are trained on the same feedforward network with:

1. an input layer of dimension d,

2. a first dense hidden layer with 19 ReLU units,
3. a second dense hidden layer with 12 ReLU units,
4. a third dense hidden layer with 9 ReLU units,
5. an output dense layer with 4 softmax units.

No dropout is used so that the objective remains deterministic.

To improve fairness, all methods start from the same initial parameter values. One base model is initialized using a fixed random seed, and its initial weights are copied to each optimizer-specific model.

Three first-order optimizers are used as baselines: SGD with momentum, RMSProp, and Adam. Each is trained for 50 epochs using mini-batches of size 32. The settings are:

1. SGD with momentum: learning rate 0.01, momentum coefficient 0.9;
2. RMSProp: learning rate 0.001;
3. Adam: learning rate 0.001.

The proposed Hessian-free method is trained in full-batch mode for 50 outer iterations with:

1. initial damping parameter $\lambda_0 = 0.1$,
2. maximum of 30 CG iterations per outer step,
3. initial trial step length $\alpha_0 = 1.0$,
4. backtracking factor $\beta = 0.5$,
5. Armijo parameter $c_1 = 10^{-4}$,
6. minimum allowable step length 10^{-8}.

The experiments record both predictive and optimization-related outputs. For all methods, training and validation loss and accuracy are stored across epochs. For the Hessian-free method, additional outputs include the gradient norm, step norm, predicted and actual reduction, reduction ratio, accepted step length, damping parameter, number of CG iterations, final CG residual norm, and number of backtracking reductions.

To support the matrix-free formulation, the Hessian–vector product is also validated numerically by comparing the automatic-differentiation product

$$H(\theta)v$$

with the finite-difference approximation

$$\frac{g(\theta + \varepsilon v) - g(\theta)}{\varepsilon},$$

for several randomly generated vectors v and a small positive parameter ε.

Final model assessment is performed on the held-out test set. For each method, the predicted class label is

$$\hat{c}(x) = \arg\max_{1 \le k \le 4} p_{\theta,k}(x).$$

The reported measures are accuracy, weighted precision, weighted recall, and weighted F1-score. A confusion matrix is also produced for each method to examine class-wise prediction behavior. Total training time and average training time per epoch are recorded as well.

To promote reproducibility, the random seed is fixed at 42, deterministic execution is enabled whenever supported, the train–validation–test split is fixed, the preprocessing pipeline is fitted once on training data and reused on validation and test data, the network architecture is identical across methods, all methods start from the same initial weights, and the number of epochs is fixed at 50.

3. Results and Discussion

This section presents the numerical results obtained from the protocol in Section 2. Because all compared methods use the same loss, initialization, and network architecture, the observed differences can be interpreted more directly as consequences of the optimization mechanism. The discussion is organized around training dynamics, optimization diagnostics of the proposed method, comparative test performance, and computational cost.

3.1. Training and Validation Dynamics

The training dynamics of all compared methods are shown in Fig. 1. Panel (a) reports weighted training and validation loss, while panel (b) reports training and validation accuracy over 50 epochs.

Fig. 1: Training dynamics of the compared optimization methods over 50 epochs. Panel (a) shows training and validation loss, while panel (b) shows training and validation accuracy for SGD with momentum, RMSProp, Adam, and the proposed Hessian-free optimization method.

The first-order methods and the proposed Hessian-free method display clearly different trajectories. The first-order baselines perform repeated mini-batch gradient updates, whereas the Hessian-free method computes one full-batch curvature-informed update per epoch. This distinction is consistent with recent observations that second-order methods may become more attractive in larger-batch regimes, where curvature can be exploited more systematically [8].

For the proposed Hessian-free method, the recorded run shows a substantial decrease in weighted training loss, from 1.4454 at the first outer iteration to 0.2943 at the last. Over the same interval, training accuracy rises from 0.2846 to 0.8030. On the validation set, accuracy increases from 0.2958 to 0.7886, while validation loss changes from 1.2579 to 1.3574. Thus, the method achieves strong predictive improvement, although the validation loss is less monotone than the training loss. This is compatible with a full-batch second-order method driven by curvature information and line-search acceptance rather than by the smoothing effect of stochastic mini-batches.

Overall, Fig. 1 shows that the proposed method is numerically viable under the common weighted objective and follows a training trajectory that is qualitatively different from those of SGD with momentum, RMSProp, and Adam.

3.2. Optimization Diagnostics of the Hessian-Free Method

A major advantage of the revised implementation is that it records diagnostics tied directly to the mathematical formulation. These are displayed in Fig. 2, Fig. 3, Fig. 4, and summarized numerically in Table 1 and Table 2.

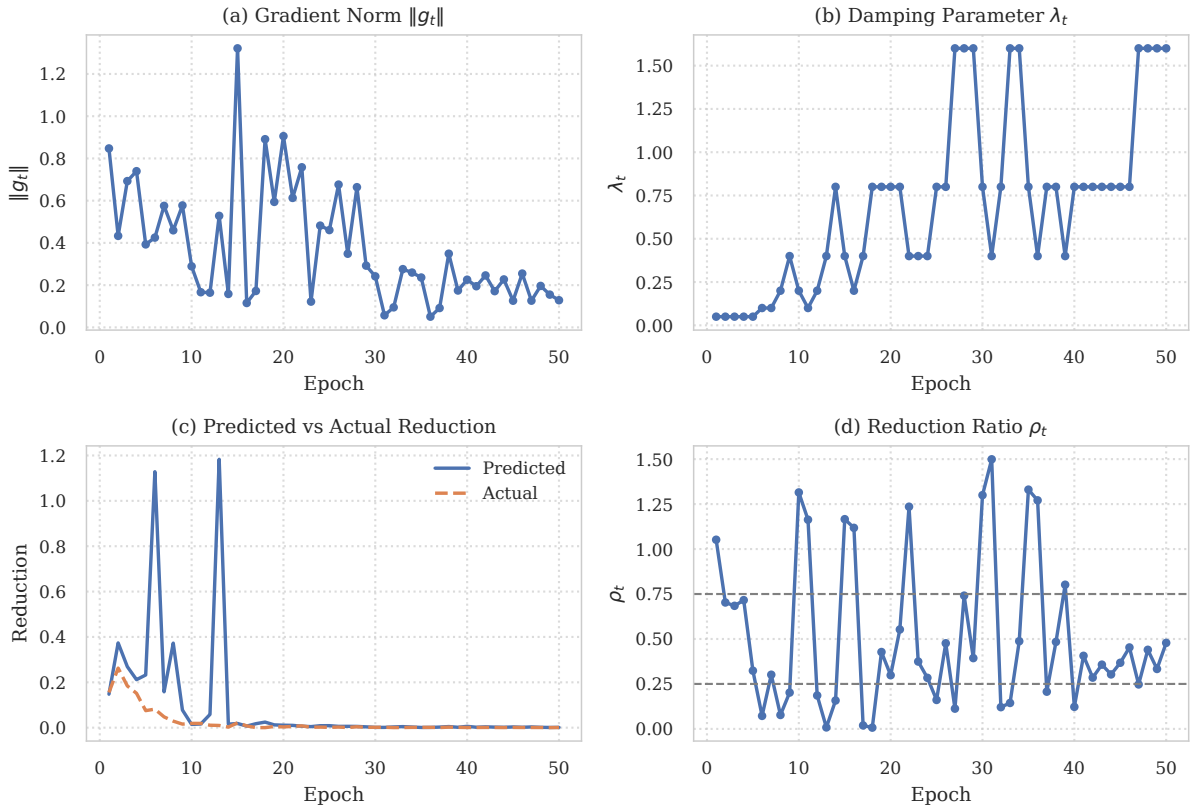


Fig. 2: Main optimization diagnostics of the proposed Hessian-free method over the outer iterations. Panel (a) shows the gradient norm $\|g_t\|$, panel (b) the damping parameter λ_t, panel (c) the predicted and actual reduction, and panel (d) the reduction ratio ρ_t.

Table 1: Summary of optimization diagnostics for the proposed Hessian-free method.

Quantity	Value
Initial gradient norm	0.8473
Final gradient norm	0.1286
Mean reduction ratio ρ_t	0.5153
Median reduction ratio ρ_t	0.3841
Accepted steps	50/50
Fallback activations	0
Backtracking-triggered iterations	24/50
Total backtracking reductions	44
Mean accepted step length	0.6977
Mean CG iterations	21.34
Mean final CG residual	0.3314
Final damping parameter	1.6

The gradient norm is the most direct indicator of first-order progress. Fig. 2(a) shows that it decreases from 0.8473 to 0.1286, suggesting that the iterates move toward a region where first-order stationarity is better approximated.

Fig. 2(b) shows the adaptive damping trajectory, which reaches 1.6 at the final iteration. This behavior is consistent with the update rule in Eq. (19), which adjusts the damping parameter according to the quality of the local model. Fig. 2(c)–(d) compare the predicted reduction in

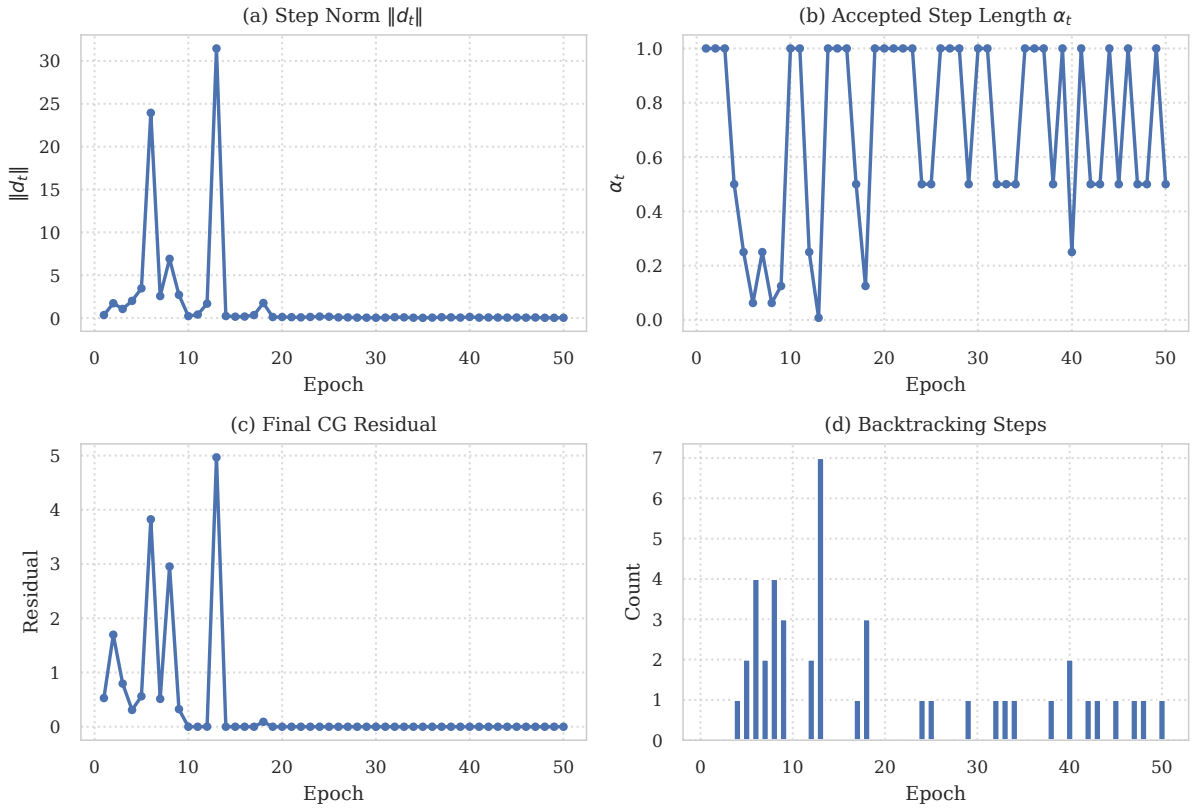


Fig. 3: Auxiliary optimization diagnostics of the proposed Hessian-free method over the outer iterations. Panel (a) shows the step norm $\|d_t\|$, panel (b) the accepted step length α_t, panel (c) the final CG residual norm, and panel (d) the number of backtracking reductions.

Table 2: Validity summary of the computed Hessian-free steps.

Quantity	Value
Iterations with $\text{pred}_t > 0$	50
Iterations with $\text{ared}_t > 0$	50
Accepted steps	50
Fallback activations	0

Eq. (13) and the actual reduction in Eq. (17), and report the corresponding ratio defined in Eq. (18). Across the 50 outer iterations, the mean value of ρ_t is 0.5153, the median is 0.3841, the minimum is 0.0070, and the maximum is 1.4990. Hence the damped quadratic model is often informative, although not uniformly accurate. Small values of ρ_t justify stronger damping in the next iteration, while larger values support more Newton-like behavior. This pattern is consistent with the trust-model interpretation of the method and with recent studies emphasizing the balance between curvature quality and computational cost [7, 11].

The auxiliary diagnostics provide further detail. The accepted step length has mean value 0.6977, minimum 0.0078125, and maximum 1.0, indicating that the full trial step is often accepted, although backtracking is occasionally substantial. Backtracking is triggered in 24 out of 50 iterations, with a total of 44 reductions. Nevertheless, every outer iteration produces an accepted step, and the gradient-fallback safeguard is never activated.

Fig. 4 further shows that the angle between the computed search direction and the steepest descent direction remains acute throughout the run:

$$\cos(\angle(-g_t, d_t)) = \frac{-g_t^\top d_t}{\|g_t\| \, \|d_t\|} > 0.$$

Thus, even though the Hessian-free step is computed only approximately, it remains aligned with

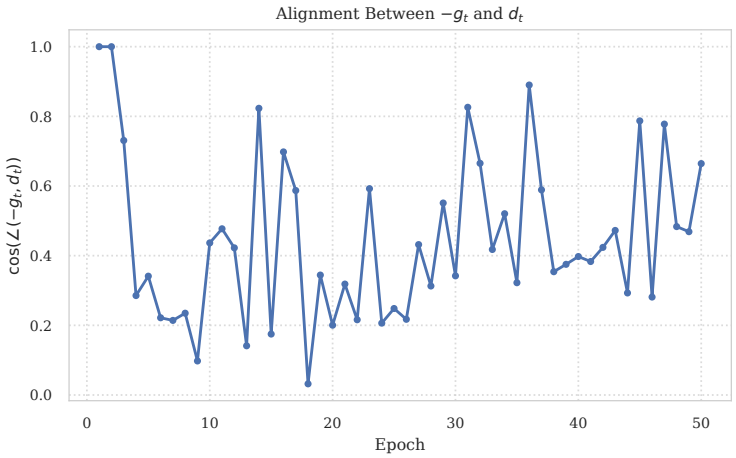


Fig. 4: Alignment between the steepest descent direction $-g_t$ and the computed search direction d_t, measured by $\cos(\angle(-g_t, d_t)) = -g_t^\top d_t/(\|g_t\|\|d_t\|)$. Positive values indicate that the computed direction remains aligned with descent.

descent. This agrees with the absence of fallback activations and indicates that the inner CG solve provides meaningful curvature-informed directions.

The absence of fallback activations indicates that the safeguard in Eq. (20) is not needed in the recorded run. Moreover, the positivity of both the predicted reduction in Eq. (13) and the actual reduction in Eq. (17) for all outer iterations is fully consistent with the descent and acceptance mechanism described in Section 2. Table 2 provides an even more direct confirmation of the descent mechanism. In all 50 iterations, the predicted reduction is positive and the actual reduction is also positive. Hence the recorded run is fully consistent with the role of the damped quadratic model and the Armijo acceptance rule. Taken together, the diagnostics support the interpretation that the implemented procedure behaves as a genuine inexact damped Newton–CG method rather than as an ad hoc second-order heuristic.

3.3. Conjugate Gradient Residual Histories

The residual histories of the inner CG iteration for selected outer epochs are shown in Fig. 5.

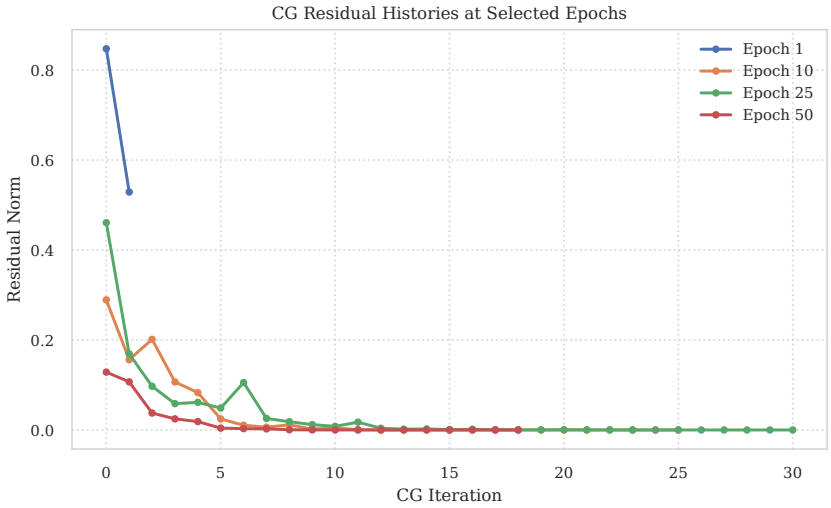


Fig. 5: Residual histories of the conjugate gradient inner iteration for selected outer epochs of the proposed Hessian-free method.

These curves correspond to the inexact solution of the damped linear system in Eq. (12). They indicate how effectively the damped Newton system is solved on representative iterations.

A decreasing residual profile shows that the matrix-free CG iteration reduces linear-system error within the explored Krylov subspace. Together with the absence of gradient-fallback activation, this suggests that the inexact inner solves are sufficiently accurate to produce meaningful curvature-informed search directions. From a computational-mathematics viewpoint, Fig. 5 strengthens the interpretation of the method as an inexact Newton–CG procedure rather than merely a heuristic second-order optimizer.

3.4. Numerical Validation of the Hessian–Vector Product

To support the matrix-free curvature formulation, the automatic-differentiation Hessian–vector product is compared with a finite-difference approximation of the gradient difference. The corresponding relative errors are shown in Fig. 6.

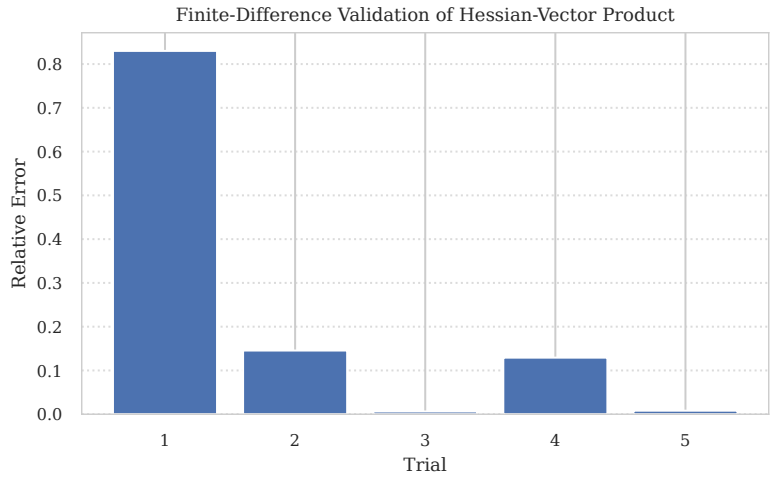


Fig. 6: Finite-difference validation of the Hessian–vector product used in the proposed Hessian-free method. The bars represent the relative error between the automatic-differentiation Hessian–vector product and the finite-difference approximation for several randomly generated test vectors.

The recorded relative errors are 0.8301, 0.1455, 0.0063, 0.1293, and 0.0081, with mean value 0.2239 and median value 0.1293. Most trials therefore show small to moderate error, although one trial exhibits a noticeably larger discrepancy. This is not unexpected in a ReLU network, where the model is only piecewise smooth and finite differences may be sensitive to local activation changes. In particular, the numerical comparison supports the practical use of the Hessian–vector identity in Eq. (10) and its damped form in Eq. (11). Overall, the validation still supports the practical consistency of the implemented Hessian–vector product, while indicating that second-order quantities in piecewise-linear networks should be interpreted with appropriate numerical caution.

3.5. Comparative Test-Set Performance

The quantitative comparison on the held-out test set is summarized in Table 3. The proposed Hessian-free method attains the best value on all four reported weighted metrics.

Table 3: Test-set performance of all compared optimization methods under the common weighted objective.

Method	Accuracy	Weighted Precision	Weighted Recall	Weighted F1-score
SGD + Momentum	0.7468	0.8458	0.7468	0.7885
RMSProp	0.7937	0.8476	0.7937	0.8094
Adam	0.7994	0.8414	0.7994	0.8138
Hessian-Free Optimization	0.8038	0.8499	0.8038	0.8188

A visual summary of these results is shown in Fig. 7.

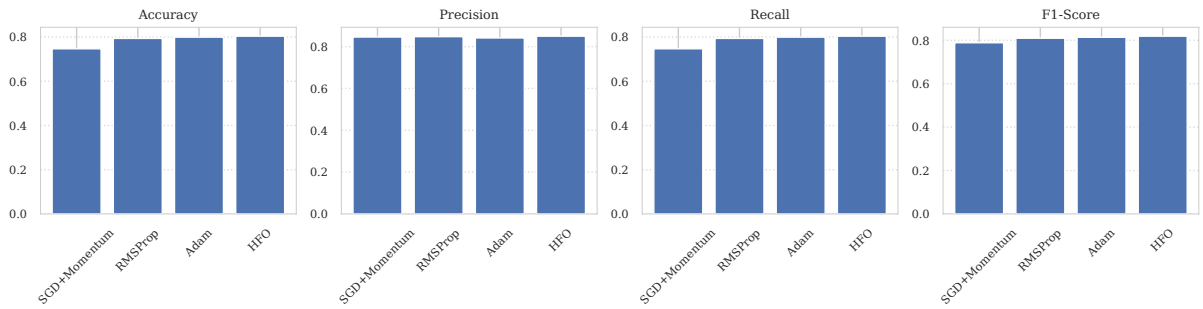


Fig. 7: Comparison of test-set accuracy, weighted precision, weighted recall, and weighted F1-score for SGD with momentum, RMSProp, Adam, and the proposed Hessian-free optimization method.

Table 3 shows that the proposed Hessian-free method reaches the highest test accuracy, 0.8038, compared with 0.7994 for Adam, 0.7937 for RMSProp, and 0.7468 for SGD with momentum. The same pattern appears for weighted F1-score, where the proposed method reaches 0.8188, compared with 0.8138 for Adam, 0.8094 for RMSProp, and 0.7885 for SGD with momentum. Its weighted precision, 0.8499, is also the highest among the four methods.

These results indicate that the proposed full-batch curvature-informed method is not only computationally feasible, but also competitive in predictive terms under the common weighted objective. In the present experiment, the use of second-order information, adaptive damping, and inexact Newton directions does not merely reproduce first-order performance; it yields the best overall weighted test-set metrics among the compared methods. This predictive improvement is accompanied by stable optimization behavior: the gradient norm decreases, descent steps are repeatedly accepted, the damping parameter responds to the reduction ratio, and the CG inner solver remains well behaved. The result is also consistent with recent classification-oriented second-order studies based on Gauss–Newton variants, suggesting that curvature-informed optimization remains competitive when extended from binary and specialized neural settings to weighted multiclass neural classification [13–15].

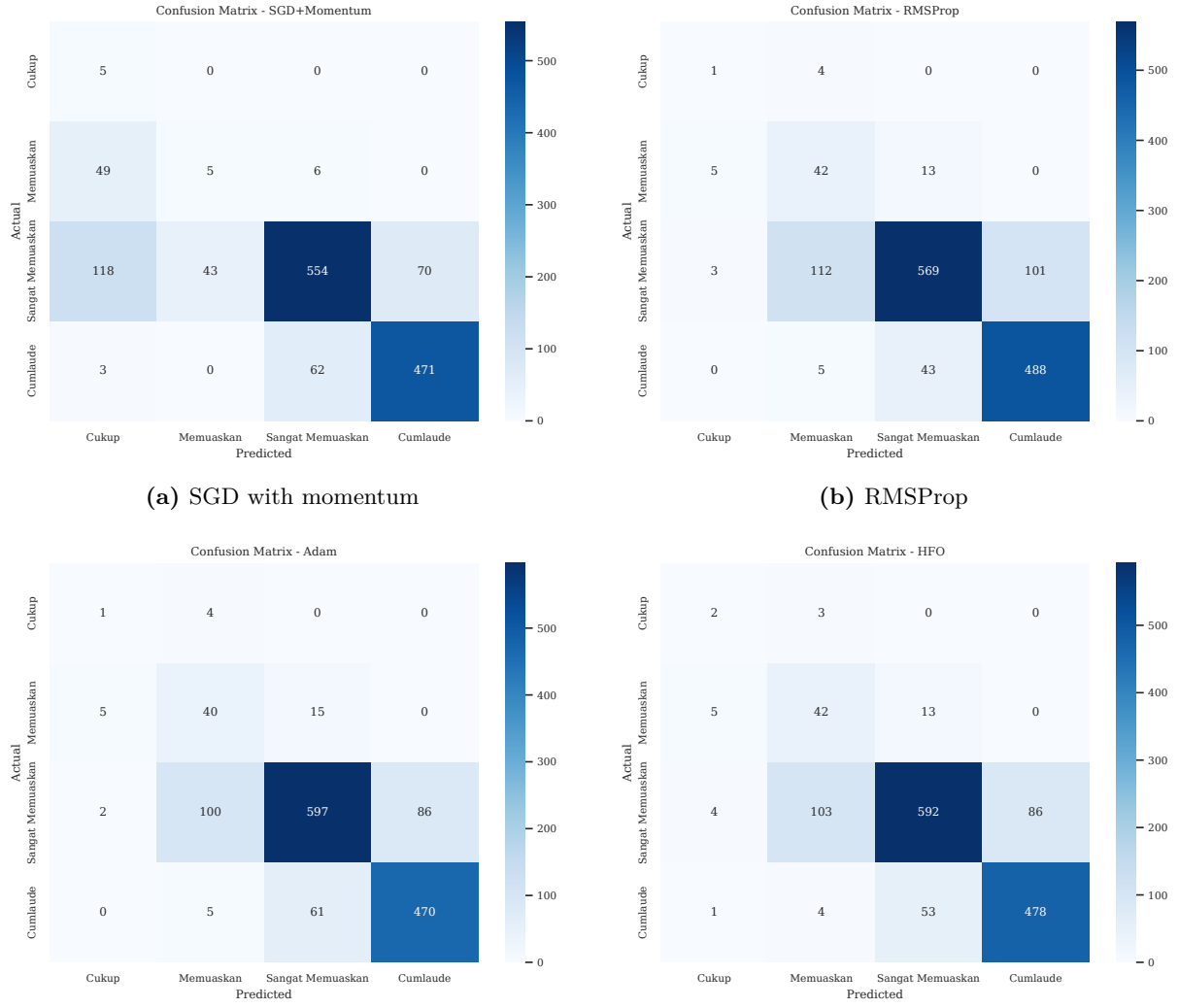
3.6. Class-wise Prediction Behavior

The class-wise behavior can be examined through the classification reports and the confusion matrices in Fig. 8. The dataset remains highly imbalanced, with only 5 test observations in the *Cukup* class and 60 in the *Memuaskan* class, compared with 785 and 536 in the two majority classes. Accordingly, the smallest classes should be interpreted cautiously.

For the minority class *Memuaskan*, the proposed method attains recall 0.7000 and precision 0.2763, which is higher in precision than RMSProp (0.2577) and Adam (0.2685), while greatly outperforming SGD with momentum in recall. For the rarest class *Cukup*, the proposed method attains precision 0.1667 and recall 0.4000. Although the support is only five samples, the result still indicates that the method does not collapse completely on the smallest class.

For the majority class *Sangat Memuaskan*, the proposed method attains precision 0.8997, recall 0.7541, and F1-score 0.8205, slightly exceeding the corresponding F1-score of Adam. For the *Cumlaude* class, the proposed method attains precision 0.8475, recall 0.8918, and F1-score 0.8691, remaining highly competitive, although SGD with momentum yields a slightly higher class-specific F1-score on this class. Overall, the proposed method also achieves the highest macro-average F1-score, namely 0.5803, which is noteworthy because macro averaging is more sensitive to minority-class performance than weighted averaging.

Hence the improvement of the proposed method is not due solely to stronger performance on the dominant classes. The class-wise results suggest a more balanced predictive profile, especially relative to SGD with momentum.



(a) SGD with momentum

(b) RMSProp

(c) Adam

(d) Hessian-free optimization

Fig. 8: Confusion matrices on the test set for the compared optimization methods. Panels (a)–(d) correspond to SGD with momentum, RMSProp, Adam, and the proposed Hessian-free optimization method, respectively. Rows represent true classes and columns represent predicted classes.

3.7. Runtime and Computational Trade-off

The runtime comparison is summarized in Fig. 9.

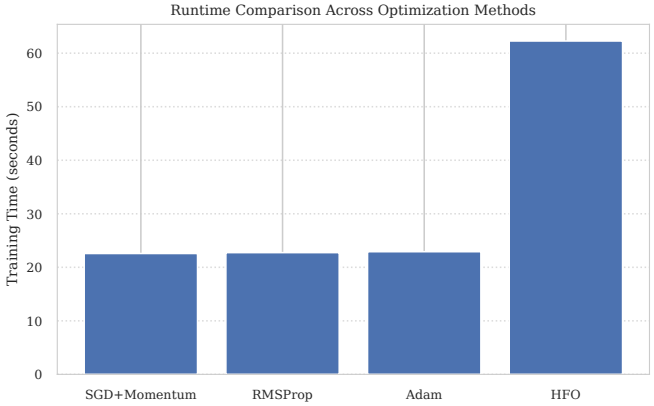


Fig. 9: Runtime comparison across optimization methods. The bars report the total training time for SGD with momentum, RMSProp, Adam, and the proposed Hessian-free optimization method.

The total training times are 22.58 seconds for SGD with momentum, 22.74 seconds for RMSProp, 22.91 seconds for Adam, and 62.27 seconds for the proposed Hessian-free method. The corresponding average time per epoch is 0.4516, 0.4548, 0.4583, and 1.2455 seconds, respectively. Thus, the proposed method is approximately 2.7 times slower per epoch than the first-order baselines.

Such a trade-off is consistent with the current second-order optimization literature, where improved optimization structure is often accompanied by higher per-iteration overhead, especially when Hessian or subspace-curvature information is used [7, 8, 11].

This runtime increase is consistent with the theoretical structure of the algorithm and, more specifically, with the per-iteration cost model in Eq. (23). Each outer iteration of the proposed method requires a full-batch gradient evaluation, repeated applications of the damped curvature operator in Eq. (11), an inner CG solve for Eq. (12), and line-search safeguards based on Eq. (15). Therefore, the improved predictive performance of the proposed method should be interpreted together with its higher computational cost.

3.8. Discussion Summary

The numerical study leads to three main observations. First, under a common weighted objective, common initialization, and deterministic architecture, the proposed Hessian-free method achieves the best overall test-set performance among the compared methods. Second, the recorded diagnostics show that the method behaves coherently as an inexact damped Newton–CG procedure: the gradient norm decreases substantially, every outer iteration produces an accepted step, the damping parameter adapts according to the reduction ratio, the computed directions remain aligned with descent, and the CG solve remains effective without triggering the fallback safeguard. Third, the improvement in predictive performance comes with a clear computational trade-off, since the Hessian-free method is substantially slower per epoch than the first-order baselines.

Taken together, these results support the conclusion that a deterministic damped Hessian-free formulation provides not only a mathematically transparent framework for multiclass neural classification, but also a practically competitive alternative to standard first-order optimization methods in the present educational dataset.

4. Conclusion

This study has presented a deterministic damped Hessian-free Newton–CG method for weighted multiclass neural classification. The method is built from a weighted categorical cross-entropy objective, a damped local quadratic model, and a matrix-free curvature operator based on Hessian–vector products. The study has emphasized the mathematical and computational structure of the method, including the inexact CG solve, Armijo backtracking, adaptive damping, and descent safeguarding.

Numerically, the proposed method proved feasible for multiclass classification on preprocessed student data with mixed categorical and numerical features. Under a common architecture, common initialization, and a common weighted objective, the Hessian-free method achieved the best overall test-set performance among the compared methods, with accuracy 0.8038, weighted precision 0.8499, weighted recall 0.8038, and weighted F1-score 0.8188. The optimization diagnostics further supported the interpretation of the method as a genuine inexact damped Newton–CG scheme: the gradient norm decreased substantially, every outer iteration produced an accepted step, the damping parameter adapted to the reduction ratio, and the inner CG solve remained stable throughout the run.

At the same time, the results confirmed the expected computational trade-off. The Hessian-free method required substantially more training time per epoch than the first-order baselines because each outer iteration involves full-batch gradient evaluation, repeated Hessian–vector products, an inner CG solve, and line-search safeguards. Its practical usefulness should therefore be judged by the balance between predictive gain and computational cost.

Overall, the present study shows that a damped Hessian-free formulation provides a mathematically transparent, reproducible, and practically competitive framework for studying second-order optimization in multiclass neural classification, while also extending recent curvature-based developments for classification and neural-network optimization toward a weighted multiclass setting [13–15]. Future work may include preconditioning for the CG inner iteration, alternative curvature approximations such as generalized Gauss–Newton or Fisher-type operators, repeated-run experiments for stronger statistical assessment, and extensions to larger architectures and broader educational datasets.

CRediT Authorship Contribution Statement

Andy Irawan: Conceptualization, Methodology, Software, Formal Analysis, Investigation, Data Curation, Visualization, Writing–Original Draft Preparation. **Zainal Abidin:** Validation, Investigation, Data Curation, Writing–Review & Editing. **Mohammad Jamhuri:** Conceptualization, Methodology, Supervision, Project Administration, Writing–Review & Editing.

Declaration of Generative AI and AI-assisted technologies

During the preparation of this manuscript, the authors used OpenAI ChatGPT for language refinement, structural editing, and drafting assistance in selected parts of the manuscript. All scientific content, mathematical formulation, numerical implementation, interpretation of results, and final written text were reviewed and verified by the authors. The authors take full responsibility for the content of this manuscript.

Declaration of Competing Interest

The authors declare no competing interests.

Funding and Acknowledgments

This research received no external funding.

The authors would like to thank all individuals and institutions who supported the completion of this study, particularly in data provision, academic discussion, and computational facilities.

Data and Code Availability

The data supporting the findings of this study contain student-related records and therefore cannot be shared publicly due to privacy and confidentiality considerations. The data may be available from the corresponding author upon reasonable request and subject to institutional permission and confidentiality requirements.

The code used to preprocess the data, train the models, and generate the numerical results reported in this study is available from the corresponding author upon reasonable request.

References

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. Official book site. No DOI is listed on the official citation page. Cambridge, MA: MIT Press, 2016. https://www.deeplearningbook.org.

[2] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep Learning”. In: *Nature* 521.7553 (2015), pp. 436–444. DOI: 10.1038/nature14539.

[3] B. T. Polyak. “Some Methods of Speeding up the Convergence of Iteration Methods”. In: *USSR Computational Mathematics and Mathematical Physics* 4.5 (1964), pp. 1–17. DOI: 10.1016/0041-5553(64)90137-5.

[4] Tijmen Tieleman and Geoffrey Hinton. *Lecture 6.5—RMSProp: Divide the Gradient by a Running Average of Its Recent Magnitude*. COURSERA: Neural Networks for Machine Learning. 2012. https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.

[5] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. Published as a conference paper at ICLR 2015. 2015. DOI: 10.48550/arXiv.1412.6980. arXiv: 1412.6980 [cs.LG]. https://arxiv.org/abs/1412.6980.

[6] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. 2nd ed. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006. DOI: 10.1007/978-0-387-40065-5. https://link.springer.com/book/10.1007/978-0-387-40065-5.

[7] Nikita Doikov, El Mahdi Chayti, and Martin Jaggi. "Second-Order Optimization with Lazy Hessians". In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. Proceedings of Machine Learning Research. 2023, pp. 8111–8148.

[8] Satoki Ishikawa and Rio Yokota. "When Does Second-Order Optimization Speed Up Training?" In: *The Twelfth International Conference on Learning Representations*. Tiny Paper. 2024. https://openreview.net/forum?id=NLrfEsSZNb.

[9] James Martens. "Deep Learning via Hessian-Free Optimization". In: *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. 2010, pp. 735–742. DOI: 10.5555/3104322.3104416. https://dl.acm.org/doi/10.5555/3104322.3104416.

[10] Barak A. Pearlmutter. "Fast Exact Multiplication by the Hessian". In: *Neural Computation* 6.1 (1994), pp. 147–160. DOI: 10.1162/neco.1994.6.1.147.

[11] Ruichen Jiang et al. "Krylov Cubic Regularized Newton: A Subspace Second-Order Method with Dimension-Free Convergence Rate". In: *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*. Vol. 238. Proceedings of Machine Learning Research. 2024, pp. 1–20.

[12] Magnus R. Hestenes and Eduard Stiefel. "Methods of Conjugate Gradients for Solving Linear Systems". In: *Journal of Research of the National Bureau of Standards* 49.6 (1952), pp. 409–436. DOI: 10.6028/jres.049.044.

[13] Mohammad Jamhuri et al. "Inexact Generalized Gauss–Newton–CG for Binary Cross-Entropy Minimization". In: *Jurnal Riset Mahasiswa Matematika* 5.2 (2025), pp. 102–122. DOI: 10.18860/jrmm.v5i2.34739.

[14] Mohammad Jamhuri et al. "Neural networks optimization via Gauss–Newton based QR factorization on SARS-CoV-2 variant classification". In: *Systems and Soft Computing* 7 (2025), p. 200195. DOI: 10.1016/j.sasc.2025.200195.

[15] Mohammad Jamhuri, Imam Mukhlash, and Mohammad Isa Irawan. "Performance Improvement of Logistic Regression for Binary Classification by Gauss-Newton Method". In: *Proceedings of the 2022 5th International Conference on Mathematics and Statistics*. ACM, 2022, pp. 12–16. DOI: 10.1145/3545839.3545842.

[16] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Deep Sparse Rectifier Neural Networks". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*. Vol. 15. Proceedings of Machine Learning Research. 2011, pp. 315–323. https://proceedings.mlr.press/v15/glorot11a.html.

[17] Kaiming He et al. "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1026–1034. DOI: 10.1109/ICCV.2015.123.

[18] Haibo He and Edwardo A. Garcia. "Learning from Imbalanced Data". In: *IEEE Transactions on Knowledge and Data Engineering* 21.9 (2009), pp. 1263–1284. DOI: 10.1109/TKDE.2008.239.

[19] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer, 2009. DOI: 10.1007/978-0-387-84858-7.

[20] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[21] Larry Armijo. "Minimization of Functions Having Lipschitz Continuous First Partial Derivatives". In: *Pacific Journal of Mathematics* 16.1 (1966), pp. 1–3. DOI: 10.2140/pjm.1966.16.1.