

Comparative Analysis of Optuna-Optimized 1D and 2D CNNs for Protein Secondary Structure Prediction

Annisa Rizqiana^{ID}, Triando Hamonangan Saragih^{ID}, Muhammad Itqan Mazdadi^{ID}, Irwan Budiman^{ID}, Adryan Maulana Saputra^{ID}

Department of Computer Science, Lambung Mangkurat University, Banjarbaru, Indonesia

Corresponding author: Triando Hamonangan Saragih (e-mail: triando.saragih@ulm.ac.id)

Abstract Protein secondary structure prediction (PSSP) is a fundamental bioinformatics task for which Convolutional Neural Networks (CNNs) offer efficient alternatives to costly experimental methods, yet systematic, tuning-controlled comparisons between 1D and 2D CNN topologies remain limited and conventional hyperparameter tuning typically relies on manual trial-and-error. This study evaluates standalone 1D and 2D CNN architectures for eight-class (Q8) PSSP under Optuna-based (TPE) hyperparameter optimization, training on CullPDB with 10-fold cross-validation and evaluating on the independent CB513 benchmark to enable paired statistical comparison, contributing a tuning-controlled comparative analysis, a reproducible optimization pipeline, cross-dataset generalization validation, and a complexity-performance trade-off analysis. Results show that Optuna's benefit is architecture-dependent: for the 1D CNN, tuning yielded a 56% smaller model (18,600 vs. 41,992 parameters) with statistically equivalent performance to the manual baseline ($p > 0.6$), while for the 2D CNN, tuning yielded significantly higher F1-score (0.4009 vs. 0.39) and AUC (0.8240 vs. 0.8032, both $p < 0.001$, Cohen's $d > 1.7$) than its baseline, at the cost of a substantially larger model (765,768 parameters); the Optuna-optimized 2D CNN configuration significantly outperformed all other configurations, indicating that this gain reflects the joint effect of representation and tuning rather than either alone, consistent with Optuna identifying a receptive field better matched to the joint sequence-feature structure than a narrow, manually-chosen kernel. These findings were obtained under a search constrained by the available cloud GPU environment and under severe Q8 class imbalance that leaves rare classes such as pi-helix poorly predicted, both of which are discussed as limitations. Automated hyperparameter optimization thus measurably improves training efficiency and, combined with a 2D representation, yields a statistically validated and practically substantial performance gain, with the choice between 1D and 2D topologies should be guided by the accuracy, interpretability, and computational-cost requirements of the downstream application.

Keywords Protein Secondary Structure Prediction, Convolutional Neural Network, Optuna Optimization

1. Introduction

Proteins are essential macromolecules whose biological function is strongly associated with their structural organization, commonly categorized into primary, secondary, tertiary, and quaternary levels [1]. Protein secondary structure, local folding patterns such as alpha-helices, beta-sheets, and beta-turns, stabilized primarily by hydrogen bonding [2], serves as an intermediate representation between amino acid sequences and three-dimensional conformations [3], and is difficult to determine directly. Experimental methods such as Nuclear Magnetic Resonance (NMR) or X-ray Crystallography remain expensive, labor-intensive, and time-consuming [4], [5], leaving the number of experimentally determined structures far behind the rapidly growing number of available

sequences. Computational prediction methods have therefore emerged as efficient, scalable alternatives [6], [7], [8], with accurate protein secondary structure prediction (PSSP) carrying substantial practical value for downstream applications such as drug discovery, protein engineering, and vaccine design. PSSP is a crucial bioinformatics task [9], [10], [11], and a wide range of approaches has been explored to capture the relationship between primary sequence and secondary-structure folding patterns. Earlier studies relied on traditional machine learning methods such as Support Vector Machines (SVM) and Decision Trees as a hybrid model [6]. More recently, deep learning approaches, deployed either as standalone or hybrid combinations of Recurrent Neural Networks [12], Convolutional Neural Networks (CNN) [13], [14], [15], Long Short-Term Memory (LSTM) [14], Bidirectional

LSTM [16], [17], [18], and Transformer-based architectures [16], [19], have demonstrated superior performance in capturing complex sequence-to-structure relationships. Among these, CNNs have been widely adopted for their ability to capture local sequence motifs and amino acid propensities effectively [20].

Despite the adoption, three related gaps remain. First, comparative studies that systematically evaluate one-dimensional against two-dimensional CNN representations for PSSP are limited, and architectural configurations such as kernel size and filter count are, in most existing CNN-based studies, determined through manual trial-and-error rather than systematic search [21], an approach that introduces human bias, is computationally inefficient, and often traps the network in a suboptimal performance region. Second, more advanced architectures, attention-based networks, hybrid CNN-BiLSTM and CNN-TCN models, and Transformer-based protein language models, typically layer additional representational components onto a convolutional backbone whose own hyperparameters remain manually selected, making it difficult to isolate the contribution of the representation choice (1D versus 2D) from the effects of these additional components and from inconsistent tuning practices across studies. Third, automated hyperparameter optimization frameworks such as Optuna, which employs a Tree-structured Parzen Estimator (TPE) that learns from previous trials to guide subsequent configurations toward more promising regions [22], [23], and whose superiority over conventional methods such as grid and random search has been validated across multiple domains [24], remain largely unexplored in the specific context of PSSP, particularly for independently comparing 1D and 2D CNN architectures under systematic optimization. Taken together, these gaps motivate a standalone, tuning-controlled comparison of 1D and 2D CNN representations as a foundational reference point: establishing how much of a model's performance can be attributed to the representation itself, under a fair, systematic hyperparameter-selection procedure, before introducing additional architectural complexity.

To address these gaps, this study proposes a comparative framework in which standalone 1D and 2D CNN architectures are each integrated with Optuna-based hyperparameter optimization, trained on the CullPDB benchmark and independently evaluated on the CB513 benchmark. This study aims to evaluate the effectiveness of 1D and 2D CNN architectures for eight-state (Q8) PSSP and to investigate the impact of Optuna-based hyperparameter optimization on model performance and training efficiency. The primary contributions of this study are as follows. First, we present a tuning-controlled comparative analysis of

standalone 1D and 2D CNN topologies for Q8 PSSP, in which both representations are optimized under an identical hyperparameter-search protocol, isolating the effect of the representation choice from the effect of tuning quality. Second, we implement and document a reproducible Optuna/TPE-based hyperparameter optimization pipeline, including the full search space, objective function, and best configurations for each architecture, reducing reliance on manual trial-and-error and enabling more systematic hyperparameter selection than fixed-heuristic tuning. Third, we validate the generalization capability of the optimized models through strict cross-dataset testing, training exclusively on CullPDB and evaluating on the independent CB513 benchmark. Fourth, we provide a quantified trade-off analysis between model complexity (parameter count) and predictive performance across the four evaluated configurations, offering practical guidance on when the added computational cost of a 2D representation and/or hyperparameter optimization is justified.

This study is structured as follows: Section II reviews the related literature on PSSP using deep learning approaches. Section III discusses the dataset used and proposed methods. Section IV presents CNN accuracy results and a comparative analysis across all configurations. Section V discusses the interpretation, comparison with prior studies, and limitations of these findings. Section VI concludes by revisiting the study's objectives, summarizing the main findings, and outlining directions for future work.

II. Related Works

Deep learning approaches, particularly Convolutional Neural Networks (CNNs), have dominated recent advances in PSSP. Guo et al. [25] demonstrated ensemble learning using Atrous Spatial Pyramid Networks with 1D CNN encoders, achieving competitive Q3 and Q8 accuracy on CB513 and CullPDB benchmarks. Similarly, Yuan et al. [26] proposed a bidirectional temporal convolutional ensemble model showing that multiscale convolution significantly enhances feature extraction over single-architecture CNNs, while Jin et al. [27] introduced an improved channel attention with multiscale convolution module within a GAN framework, collectively affirming that CNN architecture configuration directly governs prediction quality. The comparative performance of 1D and 2D CNNs on sequential biological data has attracted growing research interest. 1D CNNs efficiently capture local inter-residue patterns along the sequence dimension, whereas 2D CNNs enable richer processing of matrix-based representations such as Position-Specific Scoring Matrices (PSSM). Cheng et al. [28] demonstrated that deep residual 2D CNNs combined with PSSM features push Q3 accuracy beyond 80% on standard benchmarks, while Kim and Kwon [29] showed through AttSec that 2D

convolutional feature maps extracted from attention representations yield richer spatial encodings than purely 1D approaches. Bongirwar and Mokhade [30] further confirmed that no single architecture universally dominates without careful hyperparameter tuning tailored to the dataset characteristics.

More recently, several studies have pushed Q8 performance on the CB513 benchmark further by combining convolutional backbones with additional representational mechanisms. Cheng et al. [31] combined CNN, residual connections, bidirectional

Table 1. Feature and label indices in the CullPDB and CB513 dataset representations.

Index	Data Representation
0-21	Amino acid sequences
22-30	8 class secondary structure
31	N-Terminal
32	C-Terimnal
33-34	Solvent accessibility
35-56	PSSM's profile

Table 2. Index mapping of amino acid residues and secondary-structure labels.

Idx	Sym	Idx	Sym	Idx	Sym
0	A	11	N	22	L
1	C	12	Q	23	B
2	E	13	P	24	E
3	D	14	S	25	G
4	G	15	R	26	I
5	F	16	T	27	H
6	I	17	W	28	S
7	H	18	V	29	T
8	K	19	Y	30	NoSeq
9	M	20	X		
10	L	21	NoSeq		

recurrent units, and attention to reach 67.1% Q8 accuracy on CB513; Alanazi et al. [32] leveraged pre-trained protein language model embeddings to reach 84.6–86.6% Q3 accuracy with corresponding Q8 gains; and Sharma et al. [33] combined convolutional and bidirectional gated recurrent layers to reach 70.4% Q8 accuracy on CB513. While these results demonstrate the performance ceiling achievable by layering additional architectural components onto a CNN backbone, in each case the underlying convolutional representation (1D or 2D) and its hyperparameters are fixed choices inherited from prior work rather than the object of systematic study, making it difficult to attribute their reported gains specifically to the added components versus the quality of the underlying CNN configuration.

Automated hyperparameter optimization is essential for maximizing deep learning model performance on complex biological prediction tasks. Zubair et al. [34] applied systematic hyperparameter tuning on CNN and LSTM models for PSSP using stratified k-fold cross-validation, demonstrating that optimized configurations substantially improve prediction accuracy. Alsenan et al. [35] investigated optimization strategies on CNN-based RNA-protein binding models and found that automated tuning critically affects performance, particularly in selecting convolutional window sizes and dropout values. Despite these advances, and despite the growing use of attention-based, residual, bidirectional-recurrent, and protein-language-model-based extensions to CNN backbones for Q8 PSSP, studies that specifically isolate and compare 1D and 2D CNN representations under integrated, systematic Optuna-based hyperparameter optimization remain scarce. This represents the specific research gap that the present study addresses: establishing how much of a model's Q8 PSSP performance can be attributed to the convolutional representation itself, independent of the additional architectural components used by the more complex models cited above.

III. Method

A. Dataset

This study utilizes public protein datasets, specifically CullPDB and CB513, as referenced in [36]. The CullPDB dataset is a pre-processed dataset represented in a 2D array of 5365 rows and 39900 columns. Each row represents a protein sequence that was originally a flattened matrix of 700 x 57, where 700 denotes the maximum length of the amino acid sequence, and 57 represents the feature dimensions. Similarly, the CB513 dataset consists of 513 protein sequence data stored in a 2D array of 514 rows and 39900 columns. Both datasets have no redundancy to each other. Using CullPDB as the training set and CB513 as an independent testing benchmark is a well-established standard protocol in the protein secondary structure prediction (PSSP) domain. Several recent state-of-the-art studies have widely adopted these two datasets to validate deep learning architectures [27], [37], [38].

The features and labels are detailed in Table 1. In Table 2, the indices for amino acid residues (0-21) and the eight classes of secondary structure labels (22-30) are mapped to their respective symbols. The symbol 'NoSeq' is utilized as padding to maintain a consistent sequence length across the dataset. Each protein sample is processed as a pair of aligned sequences: the primary structure serves as the input features, while the secondary structure serves as the target labels. Beyond the amino-acid and secondary-structure-label

channels shown in Table 2, the remaining columns of the original 57-channel CullPDB/CB513 encoding (indices 31-56) comprise three additional groups: N-/C-terminal position indicators (indices 31-32), relative and absolute solvent accessibility (indices 33-34), and a PSSM (position-specific scoring matrix) evolutionary profile (indices 35-56, 22 columns). In this study, we retain only the PSSM columns as model input. To provide a clear representation of the dataset's alignment, a concrete example of a processed protein instance from the benchmark is illustrated in Table 3. Each protein sample is an aligned sequence pair, where the primary structure serves as the input feature vector and the corresponding secondary structure represents the target labels, as in [27] and the evolutionary mapping methods in [37].

a fixed-length sequence of $L = 700$ residue positions (shorter sequences are zero-padded). For every position, the original 57-channel CullPDB/CB513 encoding is split into: (i) an amino-acid identity vector, one-hot encoded over the 20 standard amino acids plus a padding token (22 channels, indices 0–21), and (ii) a PSSM (position-specific scoring matrix) evolutionary-profile vector (22 channels, indices 35–56). Of the intervening channels, indices 22-30 encode the eight-class secondary-structure label itself (used to construct the target tensor Y and, via its NoSeq marker at index 30, the sequence-validity mask M described below), while indices 31-32 (terminal indicators) and 33-34 (solvent accessibility) are auxiliary descriptors that are not used as model input in this study. Concatenating the two retained components yields the input tensor represented in Eq. (1).

$$X = [X_{aa}; X_{pssm}] \in \mathbb{R}^{(L \times 44)} \quad (1)$$

A binary validity mask, derived from the label channel, marks padded positions and is used as a sample weight during training and evaluation, so padded positions do not contribute to the loss or any reported metric. The target tensor is constructed as $Y = \text{onehot}(\arg\max(X_{\text{label}}))$ in $\mathbb{R}^{(L \times 8)}$, derived from the eight-class one-hot encoding at indices 22-29 (excluding the NoSeq marker at index 30). The sequence-validity mask M in $\{0,1\}^L$ is derived from the same NoSeq marker: $M_i = 0$ if position i is padding (NoSeq = 1 at index 30), and $M_i = 1$ otherwise. This mask is applied as a sample weight during training and evaluation for both the 1D and 2D architectures, ensuring that padded positions do not contribute to the loss or to any reported metric.

For the 1D CNN model, this feature tensor $X \in \mathbb{R}^{(L \times 44)}$ is consumed directly, allowing 1D convolutional filters to slide along the sequence-length axis to capture local evolutionary and biochemical patterns. For the 2D CNN model, X is reshaped into a single-channel two-dimensional map, $X' \in \mathbb{R}^{(L \times 44 \times 1)}$, by adding a channel axis, so that both the sequence-position axis (length L) and the feature axis (width 44) are treated as spatial dimensions for a 2D convolution. This lets each output value aggregate information from neighbouring sequence positions and neighbouring feature channels simultaneously within the convolutional kernel's receptive field, rather than combining all 44 input channels at every position as the 1D convolution does. This 2D-convolutional formulation operates on the sequence-length $\times$ feature-channel plane of a single protein and is distinct from the residue-residue pairwise (contact-map-style) 2D representations used in some prior PSSP work, in which the two spatial axes both index residue position (an $L \times L$ map) and each cell encodes a pairwise or evolutionary coupling between two residues. The present architecture does not construct such a residue-

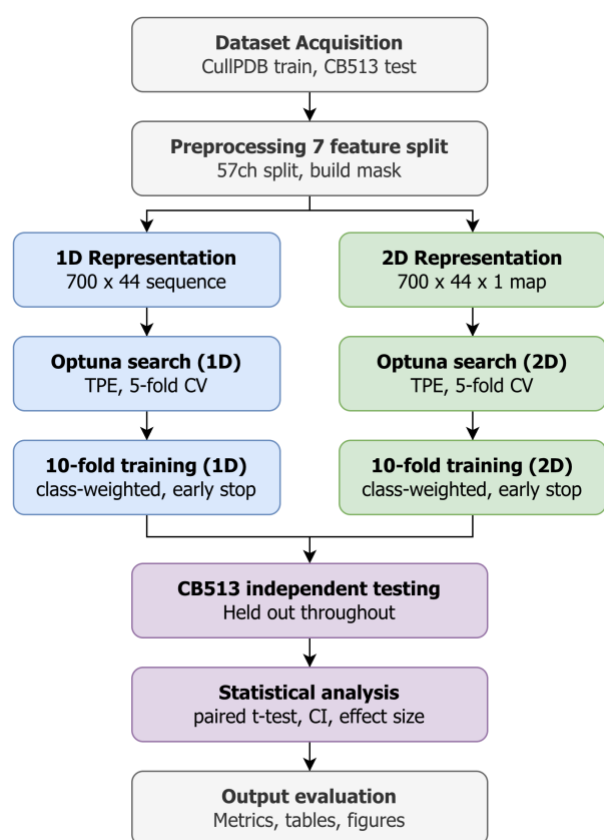


Fig. 1. End-to-end experimental workflow for 1D and 2D CNN-based protein secondary structure prediction, including preprocessing, Optuna optimization, cross-validation, CB513 testing, and statistical analysis.

B. Feature Representation

Since this study independently evaluates and compares the performance of 1D and 2D CNNs, the raw protein sequences described in Section III-A are transformed into two distinct computational representations. Each protein sample is represented as

pair map, and constructing and evaluating a true residue-pair 2D representation is identified as a direction for future work.

C. Convolutional Neural Network

Convolutional Neural Network (CNN) workflow in this study is designed as shown in Fig. 1, where the figure summarizes the complete experimental workflow employed in this study, from raw dataset acquisition through final statistical evaluation. Standalone 1D and 2D CNNs are explained below (Fig. 2 and Fig. 3).

1. 1D CNN Architecture

The 1D CNN architecture in this study is designed to capture local sequential patterns and structural motifs directly from protein primary structure. This mechanism aligns with the fundamental capability of 1D CNNs to extract localized features and transient motifs from one-dimensional sequential data, a technique previously proven effective in time-series anomaly detection [39] and astrophysical signal classification [40]. Unlike 2D image processing that relies on spatial pixels, the 1D convolutional operation slides along a single dimension representing the sequence length of the protein, allowing the model to capture relevant biochemical properties without losing sequential context. Two stacked 1D convolutional layers ("same" padding, ReLU activation) are used, followed by a dropout layer and two time-distributed dense layers, the first a ReLU projection, the second an 8-way softmax classifier applied independently at every sequence position to produce the per-residue Q8 secondary-structure prediction. Fig. 2 illustrates the structural workflow of the proposed standalone 1D CNN model and the integration with Optuna optimization. Dashed lines from the "Optuna search" box branch out to three layers with automatically determined values: two Conv layers (filter and kernel size) and a Dropout layer (rate). During the training phase, the specific structural hyperparameters of this 1D CNN, including the number of filters, kernel sizes, dropout rates, and the learning rate, are not fixed manually for the Optuna-tuned configuration; instead, they are dynamically searched and optimized using the Optuna framework as described below. Formally, the 1D convolution operation applied at position t of a single filter is defined as Eq. (2) [41].

$$y_t = \sum_{m=0}^{k-1} w_m \cdot x_{t+m} + b \quad (2)$$

where k is the kernel size, w_m denotes the learnable filter weights, b is the bias term, and x is the input feature sequence. The convolution output is then passed through a Rectified Linear Unit (ReLU) activation (Eq. (3) [42]) to introduce non-linearity:

$$\text{ReLU}(y_t) = \max(0, y_t) \quad (3)$$

2. 2D CNN Architecture

Unlike the 1D sequential approach, the 2D convolution operation processes data in a matrix form, allowing the

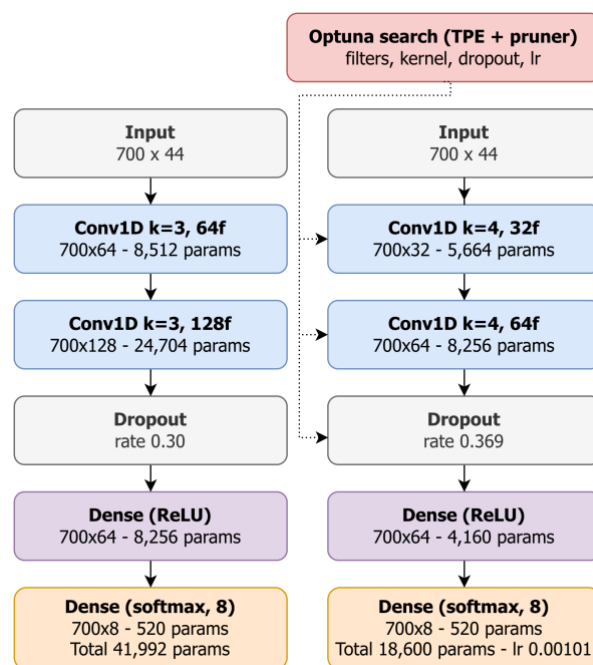


Fig. 2. Architectures of the baseline 1D CNN (left) and Optuna-optimized 1D CNN (right).

model to learn local dependencies jointly across neighbouring sequence positions and neighbouring feature channels within the convolutional kernel's receptive field. This is distinct from a residue-residue contact-map representation, in which both spatial axes would index residue position; the present formula instead operates on the sequence-position x feature channel plane of a single protein. This transition to a two-dimensional convolutional formulation allows the network to reveal spatial correlations between adjacent positions and feature channels that are otherwise inaccessible to 1D convolutions, a methodological advantage widely demonstrated in other domains, such as converting time-series signals into 2D time-frequency or sensor-position arrays to capture richer features [39], [40].

The 2D CNN architecture in this study applies two-dimensional convolution over the sequence-feature map described above, allowing the model to jointly learn local dependencies across neighbouring sequence positions and neighbouring feature channels. Two stacked 2D convolutional layers ("same" padding, ReLU activation) are used, followed by dropout. The resulting feature maps are then reshaped back into a per-position sequence representation before being passed through the same time-distributed dense classification head used in the 1D architecture. To achieve optimal performance, the architectural hyperparameters of the 2D CNN, the number of filters, kernel size, dropout rate, and learning rate, are likewise autonomously explored by the

Optuna framework for the Optuna-tuned configuration. The structural components and workflow of the proposed standalone 2D CNN model and 2D with Optuna integration are represented in Fig. 3. Analogous to the 1D operation, the two-dimensional convolution applied at spatial position (p,q) using a square kernel of size $k \times k$ is formulated as Eq. (4) [41].

$$Z_{p,q} = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} W_{m,n} \cdot M_{p+m,q+n} + b \quad (4)$$

where $W_{m,n}$ represents the learnable 2D kernel weights and $Z_{p,q}$ is the resulting feature map, which is likewise followed by the ReLU activation defined in Eq. (3). Beyond the convolutional layers described above, both the standalone 1D and 2D CNN topologies share the following common building blocks. To mitigate overfitting, dropout (Eq. (5)) is applied by randomly deactivating a proportion p of neural activations during training [43].

$$\tilde{h} = h \odot \text{Bernoulli}(1 - p) / (1 - p) \quad (5)$$

where h is the pre-dropout activation and $\tilde{h}$ is the resulting masked activation. The (reshaped, in the 2D case) feature sequence is then passed through a time-distributed fully connected (dense) layer (Eq. (6)) applied independently at each sequence position [44]:

$$u_t = \text{ReLU}(W_{fc} h_t + b_{fc}) \quad (6)$$

where W_{fc} and b_{fc} are the dense layer weights and biases. Finally, a second time-distributed dense layer applies the softmax function (Eq. (7)) to produce a probability distribution over the eight secondary structure classes for each residue [44]:

$$\hat{Y}_{\{t,c\}} = \exp(z_{\{t,c\}}) / \sum_{c'=1}^8 \exp(z_{\{t,c'\}}) \quad (7)$$

where $z_{\{t,c\}}$ is the logit for class c at position t and $\hat{Y}_{\{t,c\}}$ is the predicted probability that residue t belongs to secondary structure class c out of the eight DSSP classes.

3. Hyperparameter Optimization using Optuna

To ensure an unbiased and rigorous evaluation, the architectural and training hyperparameters for both standalone 1D CNN and 2D CNN models are systematically optimized using the Optuna framework. Optuna is a next-generation hyperparameter optimization software that utilizes a Bayesian optimization approach, specifically the Tree-structured Parzen Estimator (TPE) algorithm, to efficiently navigate the multi-dimensional search space [45]. Previous research highlighted Optuna's ability to effectively search for model configurations and was shown to outperform unoptimized models on all evaluation metrics [46]. In this study, Optuna is used to optimize the parameters in CNNs, such as the number of filters, kernel size, dropout rate, learning rate, and batch size of each CNN. Each trial is internally validated via 5-fold cross-validation on the CullPDB training pool, and the search is formulated as a

maximization problem over the search space Θ (Eq. (8)).

$$\theta^* = \arg \min_{\theta \in \Theta} L_{\text{val}}(f(\theta)) \quad (8)$$

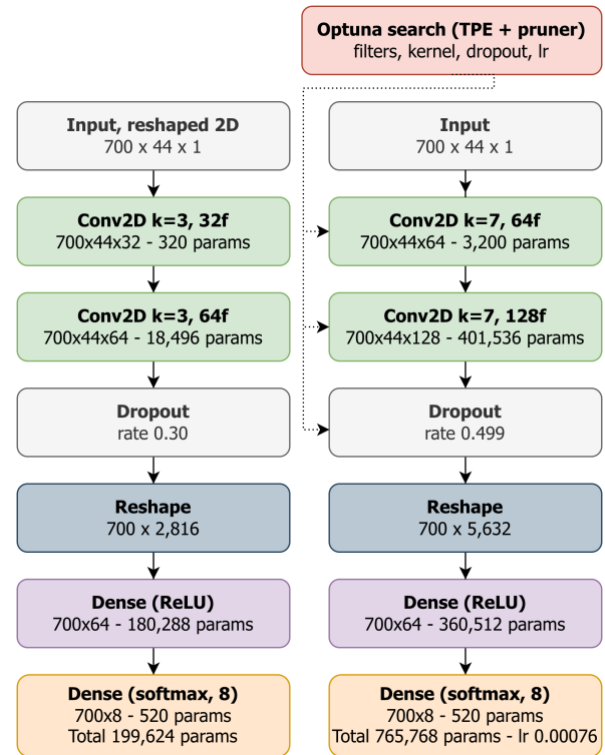


Fig. 3. Architectures of the baseline 2D CNN (left) and Optuna-optimized 2D CNN (right).

where θ denotes a candidate hyperparameter configuration (number of layers, filters, kernel size, dropout rate, learning rate), $f(\theta)$ is the CNN trained under that configuration, and L_{val} is the validation cross-entropy loss used as the objective. To efficiently navigate this search space, Optuna's Tree-structured Parzen Estimator (TPE) models the conditional densities of hyperparameters associated with low-loss and high-loss trials separately, using a loss threshold θ [47] (Eq. (9)):

$$p(\theta) = \begin{cases} l(\theta) & \text{if } L_{\text{val}} < \theta^* \\ g(\theta) & \text{if } L_{\text{val}} \geq \theta^* \end{cases} \quad (9)$$

where $l(\theta)$ and $g(\theta)$ are kernel density estimates fitted respectively to the better-performing and worse-performing trials observed so far. The next candidate configuration is then selected by maximizing the ratio between these two densities, which is asymptotically equivalent to maximizing the Expected Improvement (EI) [47] (Eq. (10)):

$$\theta_{\text{next}} = \arg \max_{\theta \in \Theta} \frac{l(\theta)}{g(\theta)} \quad (10)$$

This region with the highest $l(\theta)/g(\theta)$ ratio represents the most promising area of the search space, allowing Optuna to progressively concentrate trials toward

Algorithm 1: Optuna-Based Hyperparameter Search and Cross-Validated TrainingInput: CullPDB dataset D_{train} , CB513 dataset D_{test} , architecture type A in $\{1D, 2D\}$ Output: Best hyperparameters θ^* , ten fold-trained models, per-fold CB513 metrics

```

1:  $X, Y, \text{Mask} \leftarrow \text{Preprocess}(D_{\text{train}})$ 

2: // Stage 1: Optuna hyperparameter search (5-fold CV, TPE + Median Pruner)
3: repeat
4:  $\theta \leftarrow \text{SuggestParameters}(\text{search space})$ 
5:  $\text{objective} \leftarrow \text{Mean}(\text{5-fold cross-validation accuracy on } D_{\text{train}} \text{ using } \theta)$ 
6: prune trial early if underperforming (Median Pruner)
7: until target number of completed trials reached or trial budget exhausted
8:  $\theta^* \leftarrow \text{BestParameters}()$ 

9: // Stage 2: Final evaluation (10-fold CV, outer)
10: for fold = 1 to 10 do
11: train model on 9 folds using  $\theta^*$ , with class-weighted loss and early stopping
12: predict on held-out fold and on CB513 ( $D_{\text{test}}$ ) // CB513 never used above this line
13: record per-fold precision, recall, F1, and AUC
14: end for

15: // Stage 3: Statistical analysis
16: compute 95% confidence intervals across the 10 folds
17: for each pair of the four evaluated configurations do
18: paired t-test, Holm-Bonferroni correction, Cohen's d
19: end for
20: return  $\theta^*$ , trained models, per-fold metrics

```

configurations likely to minimize the validation loss while still allocating a fraction of trials to unexplored regions. Pseudocode of Optuna optimization is presented in [Algorithm 1](#).

4. Experimental Setup

This study evaluates four deep learning configurations for Q8 (eight-state) PSSP: standalone CNN 1D, CNN 1D with Optuna optimization, standalone CNN 2D, and CNN 2D with Optuna optimization. All four configurations were trained exclusively on the CullPDB dataset and evaluated on the independent CB513 benchmark, following the strict cross-dataset protocol established in Section III-A. All experiments used the same dataset split to ensure a fair, directly comparable evaluation across configurations [36], [37]. To account for training variance and to support statistical validation of the reported differences between configurations, each configuration is evaluated using 10-fold cross-validation on CullPDB ($\text{random_state} = 42$, held constant across all four configurations to enable paired statistical comparison), with the fixed, held-out CB513 set evaluated separately by each of the 10 fold-trained models, yielding 10 independent performance measurements per configuration for statistical comparison; CB513 itself remains fully excluded from training and hyperparameter tuning throughout. Because the Q8 label distribution is heavily imbalanced (the rarest class, π -helix, constitutes well under 0.1% of residues), class-weighted categorical cross-entropy

loss is used during training, with per-class weights computed via scikit-learn's balanced class-weighting scheme (inversely proportional to class frequency) [48] and combined multiplicatively with the sequence-validity mask described in the Feature Representation subsection ([Eq. \(11\)](#)).

$$w_c = \frac{N}{K \times n_c} \quad (11)$$

Cross-entropy loss was used as the optimization objective during training, consistent with the multi-class classification nature of the PSSP task and the probabilistic output layer of each CNN architecture [44] ([Eq. \(12\)](#)):

$$L_{\text{CE}} = -\frac{1}{N} \sum_{n=1}^N \sum_{c=1}^8 y_{n,c} \log(\hat{y}_{n,c}) \quad (12)$$

where N is the number of residues in a batch, $y_{n,c}$ is the ground-truth one-hot label indicating whether residue n belongs to class c , and $\hat{y}_{n,c}$ is the corresponding softmax probability. Each fold of each model configuration was trained for a maximum of 50 epochs, with early stopping (patience = 5 epochs on validation loss, best weights restored) determining the actual stopping point per fold; convergence epochs therefore vary across folds rather than following a single fixed value per configuration, and are reported as a distribution rather than a single number. All models were trained using the Adam optimizer with a batch size of 16 (for baseline configurations) or the Optuna-selected batch size (16 or 32) for the tuned

configurations. Experiments were run on Google Colaboratory with GPU acceleration.

For the Optuna optimization runs, the target was 5 completed trials per architecture, with a maximum budget of 15 total trial attempts (allowing for pruned or failed trials) and a Median Pruner (1D CNN: 5 startup trials, 10 warmup steps; 2D CNN: 2 startup trials, 1 warmup step). The 1D CNN search reached its target of 5 completed trials with no pruning or failures. The 2D CNN search reached the maximum budget of 15 total attempts with only 3 of the target 5 trials completing successfully (4 trials pruned, 2 trials failed, most likely due to GPU memory exhaustion given the larger filter/kernel combinations explored); the reported best 2D configuration is therefore drawn from a smaller effective sample than originally planned, and this is disclosed as a limitation in the Discussion. All experiments were conducted on Google Colaboratory's GPU runtime, whose capped, shared memory allocation is the most probable cause of these failures, and whose non-persistent sessions prevented the TPE sampler seed and individual per-trial objective values from being retained beyond the best configuration.

It is important to clarify the relationship between the hyperparameter search and the final evaluation protocol. The Optuna search and the 10-fold cross-validation used for final performance estimation both draw their training and validation splits from the same CullPDB pool, using independent K-fold partitions (5-fold for the Optuna objective, 10-fold for final evaluation) rather than a fully nested cross-validation design in which hyperparameter search would be repeated separately within each outer fold. This design was adopted due to the substantial computational cost of the search itself, which made a fully nested search (requiring the entire search to be repeated once per outer fold) computationally impractical within the scope of this study. Critically, this design does not affect the primary reported performance metrics: the CB513 benchmark, which underlies every accuracy, precision, recall, F1, and AUC figure reported in this study, was never accessed by either the Optuna search or the outer 10-fold training loops and therefore remains a fully independent test set throughout. The potential effect of this non-nested design is limited to the internal comparability of hyperparameter choices across CullPDB-validation folds, not to the CB513-based results that form the basis of this study's conclusions.

IV. Result

This study evaluates and compares four deep learning model configurations for protein secondary structure prediction: CNN 1D, CNN 1D with Optuna hyperparameter optimization, CNN 2D, and CNN 2D with Optuna optimization, on the eight-class (Q8) DSSP labeling scheme. All experiments were

conducted using identical 10-fold cross-validation splits on CullPDB (random_state = 42, held constant across all four configurations), and each fold-trained model additionally evaluated on the independent CB513 benchmark. The classification task involves predicting protein secondary structure classes, an inherently imbalanced multi-class problem, as certain structural elements appear far more frequently than others in protein datasets.

A. Overall Performance Comparison

Table 4 summarizes the aggregate performance of the four evaluated configurations. The Optuna-optimized 2D CNN configuration achieved the highest performance across all CB513 metrics (Accuracy 0.9016, Precision 0.4093, Recall 0.4386, F1 0.4009, AUC 0.8240), at the cost of substantially higher model complexity (765,768 parameters, roughly 41× more than the CNN 1D baseline). Across all four models, a consistent pattern is observed: F1-score and Precision remain considerably lower than the aggregate accuracy figure, a discrepancy characteristic of imbalanced multi-class classification, where a model can achieve a high accuracy-like aggregate by predicting majority classes well while performing poorly on minority classes.

B. Evaluation Metrics

All evaluation metrics are formally defined below in terms of the confusion-matrix quantities true positive (TP), false positive (FP), true negative (TN), and false negative (FN) counts, computed per class and macro-averaged across the eight DSSP secondary structure classes. Accuracy, Precision, Recall, and F1-score are computed following the standard definitions [49] presented in Eq. (13)-(16); AUC is derived from the Receiver Operating Characteristic (ROC) curve [50] presented in Eq. (17)-(18), computed using a one-vs-rest scheme across the eight classes and macro-averaged [49] as Eq.(19).

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (13)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (14)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (15)$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (16)$$

$$TPR = \frac{TP}{TP + FN}, \quad FPR = \frac{FP}{FP + TN} \quad (17)$$

$$AUC = \int_0^1 TPR \, d(FPR) \quad (18)$$

$$Metric_{macro} = \frac{1}{K} \sum_{k=1}^K Metric_k \quad (19)$$

C. Performance of CNN 1D and Optuna-optimized 1D CNN

The CNN 1D baseline model achieved a CB513 accuracy of 0.8813 across the 10 folds, with a CB513 precision of 0.3769, recall of 0.3937, F1-score of 0.3544, and AUC of 0.8005. When Optuna-based hyperparameter optimization was applied (5 completed trials, 5-fold cross-validation per trial), the resulting configuration, 32 filters, kernel size 4, dropout 0.369, learning rate 0.00101, achieved marginally different aggregate metrics (CB513 F1 0.3569, AUC 0.7997) that were not statistically distinguishable from the baseline (paired t-test, $p = 0.607$ for F1, $p = 0.613$ for

comparison), at the cost of a substantially larger model (765,768 parameters).

E. Statistical Validation Across Folds

$$t = \frac{\bar{d}}{\frac{s_d}{\sqrt{n}}} \tag{20}$$

To assess whether the observed performance differences reflect genuine effects rather than random variation, paired t-tests (as shown in Eq. (20) [51])($\alpha = 0.05$) were conducted between all six model pairs using the 10 per-fold F1 and AUC values as matched

Table 4. Performance of the four model configurations on the independent CB513 benchmark.

Model	Params	CB513 Accuracy	CB513 Precision	CB513 Recall	CB513 F1	CB513 AUC
CNN 1D	41,992	0.8813	0.3769	0.3937	0.3544	0.8005
Optuna-optimized 1D CNN	18,600	0.8825	0.3794	0.3971	0.3569	0.7997
CNN 2D	199,624	0.8844	0.3791	0.3988	0.3599	0.8032
Optuna-optimized 2D CNN	765,768	0.9016	0.4093	0.4386	0.4009	0.8240

Table 3. Mean F1-score and AUC with standard deviations and 95% confidence intervals across 10 folds.

Model	F1 (mean ± SD)	F1 (mean 95% CI)	AUC (mean ± SD)	AUC (mean 95% CI)
CNN 1D	0.3542 ± 0.0099	0.3542 [0.3471, 0.3613]	0.8038 ± 0.0123	0.8038 [0.7951, 0.8126]
Optuna-optimized 1D CNN	0.3562 ± 0.0045	0.3562 [0.3530, 0.3594]	0.8057 ± 0.0028	0.8057 [0.8038, 0.8077]
CNN 2D	0.3599 ± 0.0110	0.3599 [0.3520, 0.3678]	0.8071 ± 0.0108	0.8071 [0.7993, 0.8148]
Optuna-optimized 2D CNN	0.4008 ± 0.0071	0.4008 [0.3957, 0.4059]	0.8282 ± 0.0040	0.8282 [0.8254, 0.8311]

AUC). Notably, however, Optuna converged on a substantially smaller model: 18,600 parameters versus 41,992 for the baseline, a 56% reduction, while maintaining statistically equivalent predictive performance. This indicates that, for the 1D architecture, Optuna's principal benefit in this study was model compression rather than an accuracy gain.

D. Performance of CNN 2D

The CNN 2D baseline model achieved a CB513 accuracy of 0.8844, with a CB513 precision of 0.3791, recall of 0.3988, F1-score of 0.3599, and AUC of 0.8032, not statistically distinguishable from the CNN 1D baseline ($p = 0.192$ for F1). When Optuna optimization was applied to the 2D architecture (3 of a targeted 5 trials completed successfully within a 15-trial budget; the resulting configuration, 64 filters, kernel size 7, dropout 0.499, learning rate 0.00076, achieved a CB513 accuracy of 0.9016, F1-score of 0.4009 and AUC of 0.8240, the highest among all four configurations. Unlike the 1D case, this improvement was statistically significant relative to all three other configurations ($p < 0.001$ in every pairwise

samples, exploiting the fact that identical cross-validation splits were used across all four configurations. Table 5 reports the mean ± standard deviation of F1 and AUC across folds for each model, and Table 6 reports the corresponding paired t-test results. To further characterize the practical magnitude of these differences and control for the risk of false positives across six comparisons, Table 6 additionally reports Cohen's d effect sizes and Holm-Bonferroni-corrected p-values. The results reveal a clear and interpretable pattern: neither the change in representation alone (CNN 1D to CNN 2D, without Optuna) nor Optuna tuning alone (applied only to the 1D architecture) produced a statistically significant improvement in F1 or AUC. However, the combination of the 2D representation with Optuna-based hyperparameter optimization produced a statistically significant improvement over all three other configurations. This indicates that the performance gain observed in this study is attributable specifically to the joint effect of representation choice and systematic tuning, not to either factor in isolation, directly addressing the question of whether the small

performance differences reported in earlier PSSP literature are statistically meaningful. After Holm-Bonferroni correction for multiple comparisons, the same three comparisons involving Optuna-optimized 2D CNN remain statistically significant, each with a large effect size (Cohen's d ranging from -1.75 to -4.69), indicating that this performance advantage is both statistically robust and practically substantial.

F. Per-Class Performance and Class Imbalance

Table 7 compares per-class F1 and AUC between the CNN 1D baseline and the best-performing Optuna-optimized 2D CNN configuration on CB513, alongside each class's support. The Q8 label distribution is severely imbalanced: the π -helix class (I) accounts for only 0.04% of CB513 residues (30 out of approximately 85,000), while α -helix (H) and β -strand (E) together account for over half the dataset. This imbalance is reflected directly in per-class performance: F1 for the I class remains below 0.002 even in the best-performing

classes (e.g., class B, 1.4% support) without fully resolving the extreme scarcity of class I.

G. Comparative Analysis Across All Configurations

Table 8 summarizes the Optuna search outcomes for both tuned architectures. Both searches used a Tree-structured Parzen Estimator (TPE) sampler with a Median Pruner and internally validated each trial via 5-fold cross-validation. The CNN 1D search reached its target of 5 completed trials without any pruning or failures. The CNN 2D search reached the maximum trial budget (15 total attempts) with only 3 of the target 5 trials completing successfully; 4 trials were pruned early, and 2 failed, most likely due to GPU memory exhaustion given the larger kernel/filter combinations attempted. The reported best 2D configuration is therefore drawn from a smaller effective sample than originally planned; this is disclosed here as a limitation rather than presented as a fully realized 5-trial search.

Table 5. Paired statistical comparisons of F1-score and AUC across model configurations, including Holm-Bonferroni-adjusted p-values and Cohen's d effect sizes.

Comparison	Metric	t(9)	p (raw)	p (holm)	Cohen's d	Effect
CNN 1D vs Optuna-optimized 1D CNN	F1	-0.53	0.607	0.716	-0.17	negligible
CNN 1D vs Optuna-optimized 1D CNN	AUC	-0.53	0.613	1.000	-0.17	negligible
CNN 1D vs CNN 2D	F1	-1.41	0.192	0.575	-0.45	small
CNN 1D vs CNN 2D	AUC	-0.74	0.476	1.000	-0.24	small
Optuna-optimized 1D CNN vs CNN 2D	F1	-0.97	0.358	0.716	-0.31	small
Optuna-optimized 1D CNN vs CNN 2D	AUC	-0.38	0.710	1.000	-0.12	negligible
CNN 1D vs Optuna-optimized 2D CNN	F1	-10.09	<0.001	<0.001	-3.19	large
CNN 1D vs Optuna-optimized 2D CNN	AUC	-6.06	<0.001	0.001	-1.92	large
Optuna-optimized 1D CNN vs Optuna-optimized 2D CNN	F1	-14.83	<0.001	<0.001	-4.69	large
Optuna-optimized 1D CNN vs Optuna-optimized 2D CNN	AUC	-14.72	<0.001	<0.001	-4.66	large
CNN 2D vs Optuna-optimized 2D CNN	F1	-8.86	<0.001	<0.001	-2.80	large
CNN 2D vs Optuna-optimized 2D CNN	AUC	-5.53	<0.001	0.001	-1.75	large

model, while F1 for the H and E classes exceeds 0.71 and improves substantially under the Optuna-optimized 2D CNN configuration (H: 0.7165 increased to 0.8239; E: 0.6407 increased to 0.7188). Class-weighted training (via balanced class weighting, applied multiplicatively with the sequence-validity mask) was used specifically to mitigate this imbalance; the results indicate this measure improves performance on moderately represented minority

H. Comparison with Related Q8 Methods from the Literature

Table 9 presents a same-task (Q8, CB513) comparison between the configurations developed in this study and representative Q8-reporting deep learning approaches from the literature. All compared figures use the same eight-class DSSP labeling scheme and are evaluated on the same CB513 benchmark. Although all compared methods share the same eight-class DSSP labelling scheme and are evaluated on the CB513 benchmark,

Table 6. Per-class F1-score and AUC comparison between the baseline 1D CNN and Optuna-optimized 1D CNN on CB513.

Class	Support	CNN 1D F1	Optuna-optimized 2D CNN F1	CNN 1D AUC	Optuna-optimized 2D CNN AUC
L (Coil)	17,920 (21.1%)	0.4208	0.4635	0.8167	0.8363
B (Bridge)	1,181 (1.4%)	0.0876	0.1190	0.7701	0.7862
E (Strand)	18,016 (21.3%)	0.6407	0.7188	0.8872	0.9275
G (3-10 Helix)	3,132 (3.7%)	0.2133	0.2725	0.7893	0.8479
I (Pi-Helix)	30 (0.04%)	0.0008	0.0019	0.6203	0.5818
H (Alpha-Helix)	26,157 (30.9%)	0.7165	0.8239	0.9037	0.9546
S (Bend)	8,316 (9.8%)	0.3129	0.3446	0.7890	0.8072
T (Turn)	10,013 (11.8%)	0.4423	0.4629	0.8276	0.8508

Table 7. Optuna search outcomes and selected hyperparameters for the 1D and 2D CNN architectures.

Model	Complete/Pruned/Failed/Total	Best Hyperparameters
Optuna-optimized 1D CNN	5 / 0 / 0 / 5 (target reached)	filters=32, kernel=4, dropout=0.369, lr=0.00101, batch=16
Optuna-optimized 2D CNN	3 / 4 / 2 / 13 (max budget hit, target not reached)	filters=64, kernel=7, dropout=0.499, lr=0.00076, batch=16

interpret direct rankings across these studies with caution, as the underlying training protocols are not fully equivalent. Reported figures may differ in training-set composition (e.g., CullPDB variants of differing size and homology-reduction thresholds), input feature sets (PSSM-only versus combined PSSM/HMM/physicochemical profiles), sequence-length handling and padding conventions, and the precise definition of accuracy (e.g., residue-level versus sequence-averaged, or whether padded positions are fully excluded). Given these differences, the comparison in Table 9 is intended to situate this study’s results within the broader range of reported Q8 performance on CB513, rather than to establish a strict, protocol-equivalent ranking against prior work.

V. Discussion

A. Why Optuna Benefits the 2D Architecture but Not the 1D Architecture

A direct comparison of all four models shows that Optuna-based optimization did not improve performance across both architectures: for the 1D architecture, Optuna produced a smaller, statistically equivalent model, whereas for the 2D architecture, Optuna produced a significantly better-performing (but substantially larger) model. This pattern underscores that the benefit of automated hyperparameter search is architecture-dependent, and that reporting only aggregate "before/after Optuna" comparisons, without statistical testing, risks overstating the consistency of such gains. We interpret this asymmetry as follows. The 1D CNN baseline already operates on a relatively simple, low-dimensional feature interaction space. At each sequence position, the 1D convolution

aggregates information across all 44 input channels using a kernel with a small number of parameters per channel, so the baseline architecture's capacity is not obviously undersized relative to the 44-channel input. Optuna's search over 1D hyperparameters (filters, kernel size, dropout, learning rate) found a substantially smaller configuration (18,600 vs. 41,992 parameters) that matched baseline performance, suggesting that the baseline 1D architecture was already over-parameterized for the amount of signal available in the sequence-only representation, and that the primary benefit of search in this regime is compression rather than performance gain. In contrast, the 2D CNN's baseline configuration uses relatively small kernels (3×3) over the sequence-position × feature-channel plane, which may be insufficient to capture cross-channel interaction patterns that require a wider receptive field. The Optuna-tuned 2D configuration selected a substantially larger kernel (7×7) and more filters (64/128 vs. 32/64), roughly quadrupling the receptive field extent along both the sequence and feature axes. This suggests that the 2D representation contains structure, specifically, joint dependencies across neighbouring sequence positions and feature channels, that a narrow 3×3 kernel under-captures, and which a systematically searched wider kernel can better exploit. Under this interpretation, Optuna's role for the 2D architecture was not primarily compression (as for 1D) but capacity matching: identifying a receptive field size better suited to the underlying 2D structure than the manually chosen 3×3 baseline. This is consistent with the substantially larger effect size and higher final parameter count observed for the 2D +

Optuna configuration relative to the other three configurations.

B. Quantitative Comparison with Prior Studies

Table 9 lists nine prior Q8 studies evaluated on CB513 alongside their reported accuracy. Table 10 below extends this comparison along dimensions not captured by Table 9 but requested for meaningful cross-study evaluation: feature representation, model size, hyperparameter search strategy, and class-imbalance handling. Several similarities and

(MLPRNN) and Cheng et al. [31] (RECNN), rely on manually selected configurations without reporting a search procedure. Regarding class imbalance, none of the nine prior studies explicitly report a class-weighting or resampling strategy, despite the Q8 label distribution being severely imbalanced by construction of the CullPDB/CB513 dataset itself. This suggests that the per-class weaknesses discussed in Section V.D, particularly for the pi-helix class, likely recur across this literature even when not explicitly reported or analyzed,

Table 8. Comparison with related Q8 protein secondary structure prediction methods evaluated on CB513.

Method	Year	Q8 Accuracy (CB513)
This study — CNN 1D / Optuna-optimized 1D CNN / CNN 2D / Optuna-optimized 2D CNN	2026	88.1% / 88.3% / 88.4% / 90.2%
Cascaded CNN + BGRU [55]	2016	69.4%
Ensemble of diverse NN architectures [56]	2018	70.7%
CRRNN (1D CNN + residual BGRU) [52]	2018	71.4% (single) / 74.0% (ensemble)
Hybrid CNN-BRNN, 42-D profile features [53]	2020	73.5%
MLPRNN (stacked BGRU + MLP) [58]	2021	70.6%
Cascaded Feature Learning Model [54]	2023	70.74%
Ensemble BTCN+BLSTM [26]	2023	77.91%
CNN + BiGRU [57]	2024	70.4%
RECNN (CNN + residual + BGRU + attention) [31]	2025	67.1%

differences with this study are notable. Regarding feature representation, this study’s 44-dimensional input (22 one-hot AA and 22 PSSM channels) is narrower than several prior approaches: Zhang et al [52] (CRRNN) used a 50-dimensional hybrid profile, while Kumar et al. [53] and Geethu and Vimina [54] incorporated additional solvent-accessibility and physicochemical descriptors alongside PSSM. This narrower representation may partly explain why this study’s accuracy is not uniformly higher than these methods, since part of their reported performance may derive from richer input information rather than architectural differences alone.

Regarding model size, most prior studies do not report explicit parameter counts, limiting direct efficiency comparison. Zhang et al. [52] report their CRRNN ensemble at approximately 7.74 million parameters, and Li and Yu [55] report an 18.1 MB ensemble model, both substantially larger than the standalone Optuna-optimized 2D CNN configuration in this study (765,768 parameters). In contrast, Drori et al. [56], Kumar et al. [53], Lu [57], and Cheng et al. [31] (RECNN) do not report parameter counts or model size, precluding a complete efficiency comparison across the full set of nine studies. Regarding hyperparameter search, none of the nine prior studies report a systematic, automated search comparable in documentation to the Optuna/TPE procedure used here; Yuan et al. [26] mention a grid search over hyperparameters but do not report the search space or number of configurations evaluated, while the remaining eight studies, including Lyu et al. [58]

since Q8 accuracy alone does not reveal per-class performance.

Because F1-score is not consistently reported across these nine studies (most report only aggregate Q8 accuracy in Table 9), a direct F1-based ranking against this study’s macro-F1 (0.4009 for Optuna-optimized 2D CNN) is not possible without re-evaluating each prior model under identical conditions, itself a nontrivial undertaking given inconsistent public availability of trained models and evaluation code across this literature. We therefore limit quantitative comparison in Table 9 to accuracy and note this as a boundary of what can be responsibly compared given the current state of reporting practice in this literature.

C. Explaining the Unusually High Accuracy

The reported CB513 accuracy (88.1-90.2%) is substantially higher than both the macro-averaged F1-score (0.35-0.40) obtained in this study and the Q8 accuracy range (67-78%) reported by the prior studies compared in Table 10. We attribute this gap primarily to two related factors: the severe class imbalance of the Q8 label distribution, and the accuracy-focused objective used during hyperparameter optimization. First, the Q8 label distribution in CB513 is highly skewed: the three most frequent classes (coil, strand, and alpha-helix) together account for 73.3% of all residues, while the rarest class, pi-helix, accounts for only 0.035% (30 of 84,765 residues), a dominance ratio of roughly 2,070 to 1. Because overall accuracy is computed as the proportion of correctly classified residues across the entire dataset, it is numerically dominated by performance on the majority classes; a

model that classifies the three dominant classes well will achieve high aggregate accuracy even if it performs poorly on rare classes such as pi-helix. Macro-averaged F1-score, in contrast, weights every class equally regardless of its support, so the same poor minority-class performance that has negligible impact on accuracy substantially depresses the macro-F1 score. This divergence is directly visible in the per-class results (Table 7): F1 for the pi-helix class remains below 0.002 across all configurations, even though these same classes contribute almost nothing to the overall accuracy figure due to their extremely low support. Second, the Optuna hyperparameter search in this study was explicitly formulated to maximize mean validation accuracy as its objective (Eq. (8)), not F1-score or any class-balanced metric. Given the class distribution described above, a hyperparameter configuration that improves classification of the dominant classes will increase the accuracy objective almost regardless of its effect on minority classes, since minority-class residues contribute negligibly to the accuracy calculation. This creates a systematic incentive, inherent to the search objective itself rather than any single configuration, for Optuna to select hyperparameters that are well-suited to the majority classes while remaining comparatively insensitive to precision and recall on rare classes such as pi-helix. This is consistent with the pattern observed across all four configurations in this study, where substantial gains in accuracy (e.g., 88.1% to 90.2% for Optuna-optimized 2D CNN) were accompanied by

comparatively modest gains in macro-F1 (0.3544 to 0.4009), and where the most Optuna-tuned configuration (Optuna-optimized 2D CNN) did not achieve a proportionally larger improvement on the rarest classes (Table 7).

We note this as an important methodological consideration for future work: because accuracy and macro-F1 respond very differently to class imbalance, the choice of optimization objective materially shapes which aspects of model performance are improved by automated hyperparameter search. Future iterations of this pipeline could adopt a class-balanced objective (e.g., macro-F1 or balanced accuracy) during the Optuna search to more directly target minority-class performance, potentially at some cost to the aggregate accuracy figure reported here. We recommend that readers interpret the accuracy figures in this study primarily as a measure of majority-class performance, and treat macro-averaged precision, recall, F1-score, and AUC as the more informative metrics for assessing overall model quality on this imbalanced task.

D. The Pi-Helix Problem: Interpretation and Recommendations

Across all four configurations, the π -helix (I) class was essentially unrecognized, with F1-scores between 0.0008 and 0.0019 despite AUC values of 0.58-0.62 for this class, indicating that the model retains some discriminative signal (better than random ranking) but fails to translate this into confident positive classifications at the operating threshold implied by softmax argmax decoding. This gap between AUC and

Table 9. Comparison of this study with previous Q8/CB513 methods in terms of feature representation, model size, hyperparameter optimization, and class-imbalance handling.

Study	Year	Feature Rep.	Model Size	HP Search	Imbalance Handling
Cascaded CNN + BGRU [55]	2016	PSSM+One-hot (22-dim)	18.1 MB, ensemble of 10	Manual	Not addressed
Ensemble of diverse NN architectures [56]	2018	PSSM+One-hot	Not reported, ensemble	Manual	Not addressed
CRRNN (1D CNN + residual BGRU) [52]	2018	50-dim hybrid profile	7.74M params, ensemble of 10	Manual	Not addressed
Hybrid CNN-BRNN, 42-D profile features [53]	2020	42-dim hybrid profile	Not reported	Manual	Not addressed
MLPRNN (stacked BGRU + MLP) [58]	2021	PSSM+HMM	Reductive (smaller by design)	Manual	Not addressed
Cascaded Feature Learning Model [54]	2023	PSSM+PSEM+solvent+physchem	Not reported, multi-stage transfer	Manual	Not addressed
Ensemble BTCN+BLSTM [26]	2023	PSSM+one-hot (41 dim)	Not reported, 6-model ensemble	Manual	Not addressed
CNN + BiGRU [57]	2024	PSSM+one-hot	Not reported	Manual	Not addressed
RECNN (CNN + residual + BGRU + attention) [31]	2025	PSSM+one-hot+attention	Not reported	Manual	Not addressed
This study (all 4 configs)	2026	PSSM+one-hot (44 dim)	18,600-765,768 params (config-dependent)	Optuna/TPE, 5-fold CV	Class-weighted loss

F1 for a single class is itself diagnostic: it suggests a threshold/decision-boundary problem compounded by extreme scarcity, rather than a complete absence of learnable signal. Three factors plausibly compound to produce this outcome. First, extreme scarcity: the π -helix class constitutes only 30 of approximately 84,765 CB513 residues (0.04%), meaning that even a well-calibrated classifier faces a severe base-rate challenge; at this prevalence, the softmax output for class I must overcome very strong priors toward the seven other classes to be selected as the argmax prediction for any residue. Second, the class-weighted loss function used in this study reweights the loss contribution of each class but does not directly address the decision boundary at inference time; class weighting influences gradient magnitude during training but does not guarantee that the resulting decision boundary favors minority-class predictions at prediction time. Third, π -helices are biologically rare and structurally short (typically spanning only a few consecutive residues), meaning that even the training data itself may contain too few contiguous π -helix examples for the model to learn robust sequence motifs specific to this class, independent of the class-weighting scheme applied.

We recommend three directions for future work to address this specific weakness. First, focal loss (Lin et al., 2017) directly modifies the loss landscape to down-weight easy (typically majority-class) examples and up-weight hard (typically minority-class) examples during training, which may more directly influence the decision boundary than class-weighted cross-entropy alone. Second, hierarchical classification, first predicting a coarser 3-state label (H/E/C) and then refining within the H-superclass to distinguish α -helix (H), 3-10 helix (G), and π -helix (I), may allow the model to first establish the broad structural category before attempting the much harder fine-grained distinction, potentially improving minority-class recall within each coarse category. Third, given the extreme rarity of π -helix residues even in the full CullPDB training pool, targeted data augmentation or transfer learning from structurally related classes (e.g., pretraining on 3-10 helix examples before fine-tuning on π -helix) may partially compensate for the intrinsic scarcity of training examples for this class.

E. Practical Implications

The results offer concrete guidance for practitioners choosing between 1D and 2D CNN representations for Q8 PSSP under an Optuna-based optimization pipeline. When model compactness and computational efficiency are prioritized, for example, in resource-constrained deployment settings, the Optuna-

optimized 1D CNN configuration is recommended: Optuna identified a configuration with 56% fewer parameters than the manually-configured 1D baseline while maintaining statistically equivalent predictive performance (indicating no accuracy cost for this substantial efficiency gain). When predictive performance is the primary objective and computational budget is less constrained, the Optuna-optimized 2D CNN configuration is recommended: it achieved the only statistically significant and practically substantial (large effect size) improvement over all three other configurations, at the cost of a model 41 times larger than the CNN 1D baseline. Practitioners without the computational budget to run a full Optuna search should note that manually configured 2D and 1D baselines in this study were statistically indistinguishable from each other, suggesting that representation choice alone, without accompanying systematic tuning, is unlikely to yield a meaningful accuracy advantage; the benefit observed here arose specifically from the combination of representation and tuning, not either factor in isolation.

F. Limitations

A primary limitation of this study stems from severe dataset class imbalance across CullPDB and CB513, particularly regarding extremely rare secondary structure classes such as the π -helix (<0.1% of residues). Although class-weighted loss partially mitigated this issue for moderately rare classes, extreme scarcity hindered the model's capacity to accurately predict minority structures, highlighting the need for future targeted oversampling or hierarchical classification schemes. Furthermore, the architectural scope was intentionally restricted to standalone CNN topologies, omitting attention mechanisms, recurrent layers, and protein language models, to isolate representation choices; thus, overall performance reflects a baseline benchmark rather than parity with complex hybrid architectures. Methodologically, a key constraint was directing the Optuna hyperparameter search exclusively toward maximizing overall accuracy. Because the search space optimized solely for accuracy, the selected hyperparameter configurations may be suboptimal for other critical evaluation metrics, such as macro F1-score or minority-class recall, which better capture performance under severe class imbalance. Finally, computational resource limits restricted the hyperparameter trial budget, which introduces a potential risk of slight optimistic bias in internal CullPDB estimates, though final evaluations on the isolated CB513 test set remain unaffected.

VI. Conclusion

This study evaluated the effectiveness of standalone 1D and 2D CNN architectures for eight-class (Q8) protein secondary structure prediction under Optuna-

based hyperparameter optimization, using 10-fold cross-validation and paired statistical testing. The results show that Optuna's benefit is architecture-dependent: for the 1D CNN, hyperparameter optimization reduced the model from 41,992 to 18,600 parameters (a 56% reduction) while maintaining statistical performance relative to the manually configured baseline (0.3544 and 0.3569 on F1 and 0.8005 and 0.7997 on AUC for each model). For the 2D CNN, hyperparameter optimization increased CB513 accuracy from 88.4% to 90.2%, F1 from 0.3599 to 0.4009, and AUC from 0.8032 to 0.8240. The Optuna-optimized 2D CNN therefore achieved the best overall performance among the four evaluated configurations, with the observed gain attributable specifically to the joint effect of the 2D representation and systematic tuning rather than either factor alone. This best-performing result, however, should be interpreted within the conditions under which it was obtained: the Optuna search for the 2D architecture was limited and incomplete, with only 3 of a targeted 5 trials completing successfully due to computational constraints; the Q8 label distribution exhibits severe class imbalance, leaving rare classes such as pi-helix poorly predicted despite class-weighted training; and the comparability of this study's accuracy figures to prior literature cannot be fully guaranteed, as evaluation protocols vary across studies and were not independently re-verified under identical conditions. These conditions temper the extent to which the observed improvement should be read as general superiority of the 2D + Optuna configuration. Future work should prioritize four directions. First, fully nested cross-validation should be implemented to eliminate any residual risk of optimistic bias arising from the current non-nested search. Second, the Optuna search budget should be expanded, particularly for the 2D architecture, to determine whether additional trials yield further gains. Third, evaluation on external, independent datasets beyond CB513 would help assess the generalizability of the reported results. Fourth, a systematic computational-efficiency analysis would complement the parameter-count-based efficiency comparison presented in this study and provide a more complete picture of the practical trade-offs between configurations.

Declarations

Acknowledgments

None.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data Availability

The datasets analyzed in this study (CullPDB and CB513) are publicly available. These are from Zhou and Troyanskaya's research [36].

Ethics Approval

Not applicable.

Consent for Publication

Not applicable.

Competing Interests

The authors declare that they have no competing interests.

Author Contributions

Annisa Rizqiana: Conceptualization, Methodology, Data Acquisition, Writing (original draft)

Triando Hamonangan Saragih: Supervision, Methodology, Software, Validation, Analysis, Writing (review & editing)

Muhammad Itqan Mazdadi: Methodology, Validation, Writing (review & editing)

Irwan Budiman: Analysis, Validation, Writing (review & editing)

Adryan Maulana Saputra: Software, Writing (review & editing)

References

- [1] N. Ajomiwe, M. Boland, S. Phongthai, M. Bagiyal, J. Singh, and L. Kaur, "Protein Nutrition: Understanding Structure, Digestibility, and Bioavailability for Optimal Health," *Foods*, vol. 13, no. 11, p. 1771, Jun. 2024, doi: 10.3390/foods13111771.
- [2] O. S. Zakharov, A. V. Rudik, D. A. Filimonov, and A. A. Lagunin, "Prediction of Protein Secondary Structures Based on Substructural Descriptors of Molecular Fragments," *Int. J. Mol. Sci.*, vol. 25, no. 23, p. 12525, Dec. 2024, doi: 10.3390/ijms252312525.
- [3] J. Zhang, J. Qian, Q. Zou, F. Zhou, and L. Kurgan, "Recent Advances in Computational Prediction of Secondary and Supersecondary Structures from Protein Sequences," in *Methods in Molecular Biology*, vol. 2870, Humana Press, 2024, pp. 1–19. doi: 10.1007/978-1-0716-4213-9_1.
- [4] J. H. Wat *et al.*, "Sequence-Specific Protein Secondary-Structure Assignment with Isotope Reverse-Labeled Amide I Spectroscopy," *J. Phys. Chem. B*, vol. 129, no. 46, pp. 11959–11968, Nov. 2025, doi: 10.1021/acs.jpcc.5c06200.
- [5] N. Fatima Ali, S. Khan, and S. Zahid, "A critical address to advancements and challenges in

- computational strategies for structural prediction of protein in recent past," *Comput. Biol. Chem.*, vol. 117, Art. No. 108430, Aug. 2025, doi: 10.1016/j.compbiolchem.2025.108430.
- [6] H. M. Afify, M. B. Abdelhalim, M. S. Mabrouk, and A. Y. Sayed, "Protein secondary structure prediction (PSSP) using different machine algorithms," *Egyptian Journal of Medical Human Genetics*, vol. 22, no. 1, Art. no. 54, Jun. 2021, doi: 10.1186/s43042-021-00173-w.
- [7] B. Huang *et al.*, "Protein Structure Prediction: Challenges, Advances, and the Shift of Research Paradigms," *Genomics Proteomics Bioinformatics*, vol. 21, no. 5, p. 913, Oct. 2023, doi: 10.1016/j.gpb.2022.11.014.
- [8] Y. Qin, Z. Chen, Y. Peng, Y. Xiao, T. Zhong, and X. Yu, "Deep learning methods for protein structure prediction," *MedComm - Future Medicine*, vol. 3, no. 3, p. e96, Sep. 2024, doi: 10.1002/mef2.96.
- [9] Y. Zhao and Y. Liu, "OCLSTM: Optimized convolutional and long short-term memory neural network model for protein secondary structure prediction," *PLoS One*, vol. 16, no. 2, p. e0245982, Feb. 2021, doi: 10.1371/journal.pone.0245982.
- [10] Y. Chen, G. Chen, and C. Y. C. Chen, "PSSP-MFFNet: A Multifeature Fusion Network for Protein Secondary Structure Prediction," *ACS Omega*, vol. 9, no. 5, p. 5985, Feb. 2024, doi: 10.1021/acsomega.3c10230.
- [11] Y. N. H. Al-Shameri, İ. Kösesoy, H. Gündüz, and Ö. F. Yilmaz, "Scalable and Efficient Protein Secondary Structure Prediction Using Autoencoder-Reduced ProtBERT Embeddings," *Applied Sciences*, vol. 15, no. 13, Art. no. 7112, Jun. 2025, doi: 10.3390/app15137112.
- [12] R. R. Ema, A. Khatun, M. A. Hossain, M. R. Akhond, N. Hossain, and M. Y. Arafat, "Protein Secondary Structure Prediction using Hybrid Recurrent Neural Networks," *Journal of Computer Science*, vol. 18, no. 7, pp. 599–611, Jul. 2022, doi: 10.3844/jcssp.2022.599.611.
- [13] A. Rizqiana and Afiahayati, "N-Grams Modeling for Protein Secondary Structure Prediction: Exploring Local Features and Optimal CNN Parameters," *BIO Web Conf.*, vol. 75, p. 01008, Nov. 2023, doi: 10.1051/bioconf/20237501008.
- [14] J. Cheng, Y. Liu, and Y. Ma, "Protein secondary structure prediction based on integration of CNN and LSTM model," *J. Vis. Commun. Image Represent.*, vol. 71, p. 102844, Aug. 2020, doi: 10.1016/j.jvcir.2020.102844.
- [15] V. M. Sutanto, Z. I. Sukma, and A. Afiahayati, "Predicting Secondary Structure of Protein Using Hybrid of Convolutional Neural Network and Support Vector Machine," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 1, p. 232-243, Feb. 2021, doi: 10.22266/ijies2021.0228.23.
- [16] B. Dong *et al.*, "SERT-StructNet: Protein secondary structure prediction method based on multi-factor hybrid deep model," *Comput. Struct. Biotechnol. J.*, vol. 23, pp. 1364–1375, Dec. 2024, doi: 10.1016/j.csbj.2024.03.018.
- [17] V. Bongirwar and A. S. Mokhadde, "A Hybrid Bidirectional Long Short-Term Memory and Bidirectional Gated Recurrent Unit Architecture for Protein Secondary Structure Prediction," *IEEE Access*, vol. 12, pp. 115346–115355, 2024, doi: 10.1109/access.2024.3444468.
- [18] L. Zhao, J. Li, W. Zhan, X. Jiang, and B. Zhang, "Prediction of protein secondary structure by the improved TCN-BiLSTM-MHA model with knowledge distillation," *Sci. Rep.*, vol. 14, no. 1, p. 16488, Dec. 2024, doi: 10.1038/s41598-024-67403-0.
- [19] T. Wu, W. Cheng, and J. Cheng, "Improving Protein Secondary Structure Prediction by Deep Language Models and Transformer Networks," *Methods Mol. Biol.*, vol. 2867, pp. 43–53, 2025, doi: 10.1007/978-1-0716-4196-5_3.
- [20] D. P. Ismi, R. Pulungan, and Afiahayati, "Deep learning for protein secondary structure prediction: Pre and post-AlphaFold," *Comput. Struct. Biotechnol. J.*, vol. 20, pp. 6271–6286, Jan. 2022, doi: 10.1016/j.csbj.2022.11.012.
- [21] L. Zhao, J. Li, B. Zhang, and X. Jiang, "Combining knowledge distillation and neural networks to predict protein secondary structure," *Scientific Reports*, vol. 15, no. 1, Art. no. 32031, Aug. 2025, doi: 10.1038/s41598-025-17513-0.
- [22] M. A. K. Raiaan *et al.*, "A systematic review of hyperparameter optimization techniques in Convolutional Neural Networks," *Decision Analytics Journal*, vol. 11, p. 100470, Jun. 2024, doi: 10.1016/j.dajour.2024.100470.
- [23] P. Srinivas and R. Katarya, "hyOPTXg: OPTUNA hyper-parameter optimization framework for predicting cardiovascular disease using XGBoost," *Biomed. Signal Process. Control*, vol. 73, p. 103456, Mar. 2022, doi: 10.1016/j.bspc.2021.103456.
- [24] A. Khalifa, F. Ben Said, and Y. H. Kacem, "Optuna-Optimized Meta-Learner and Ensemble Learning Models for Student Performance Prediction," *Procedia Comput. Sci.*, vol. 270, pp. 1826–1835, Jan. 2025, doi: 10.1016/j.procs.2025.09.303.
- [25] Y. Guo, J. Wu, H. Ma, S. Wang, and J. Huang, "Deep Ensemble Learning with Atrous Spatial

- Pyramid Networks for Protein Secondary Structure Prediction," *Biomolecules*, vol. 12, no. 6, Art. no. 774, Jun. 2022, doi: 10.3390/biom12060774.
- [26] L. Yuan, Y. Ma, and Y. Liu, "Ensemble deep learning models for protein secondary structure prediction using bidirectional temporal convolution and bidirectional long short-term memory," *Front. Bioeng. Biotechnol.*, vol. 11, p. 1051268, Feb. 2023, doi: 10.3389/fbioe.2023.1051268.
- [27] X. Jin, L. Guo, Q. Jiang, N. Wu, and S. Yao, "Prediction of protein secondary structure based on an improved channel attention and multiscale convolution module," *Front. Bioeng. Biotechnol.*, vol. 10, p. 901018, Jul. 2022, doi: 10.3389/fbioe.2022.901018.
- [28] J. Cheng, Y. Xu, and Y. Zhao, "Prediction of protein secondary structure based on deep residual convolutional neural network," *Biotechnology & Biotechnological Equipment*, vol. 35, no. 1, pp. 1881–1890, 2021, doi: 10.1080/13102818.2022.2026815.
- [29] Y. Kim and J. Kwon, "AttSec: protein secondary structure prediction by capturing local patterns from attention map," *BMC Bioinformatics*, vol. 24, no. 1, Art. no. 183, May 2023, doi: 10.1186/s12859-023-05310-3.
- [30] V. Bongirwar and A. S. Mokhade, "An improved multi-scale convolutional neural network with gated recurrent neural network model for protein secondary structure prediction," *Neural Comput. Appl.*, vol. 36, no. 24, pp. 15063–15074, Aug. 2024, doi: 10.1007/s00521-024-09822-8.
- [31] L. Cheng *et al.*, "RECNN: A Novel Deep Learning Model to Predict 8-state Protein Secondary Structure," *Communications in Computer and Information Science*, vol. 2387 CCIS, pp. 172–181, 2025, doi: 10.1007/978-981-96-1907-8_17.
- [32] W. Alanazi, D. Meng, and G. Pollastri, "Porter 6: Protein Secondary Structure Prediction by Leveraging Pre-Trained Language Models (PLMs)," *International Journal of Molecular Sciences*, vol. 26, no. 1, p. 130, Dec. 2024, doi: 10.3390/ijms26010130.
- [33] L. Sharma, A. Deepak, A. Ranjan, and G. Krishnasamy, "A novel hybrid CNN and BiGRU-Attention based deep learning model for protein function prediction," *Stat. Appl. Genet. Mol. Biol.*, vol. 22, no. 1, Jan. 2023, doi: 10.1515/sagmb-2022-0057, PMID: 37658681.
- [34] M. Zubair *et al.*, "A Deep Learning Approach for Prediction of Protein Secondary Structure," *Computers, Materials & Continua*, vol. 72, no. 2, pp. 3705–3718, Mar. 2022, doi: 10.32604/cmc.2022.026408.
- [35] S. Alsenan, I. Al-Turaiki, M. Aldayel, and M. Tounsi, "Role of Optimization in RNA-Protein-Binding Prediction," *Curr. Issues Mol. Biol.*, vol. 46, no. 2, pp. 1360–1373, Feb. 2024, doi: 10.3390/cimb46020087.
- [36] J. Zhou and O. G. Troyanskaya, "Deep Supervised and Convolutional Generative Stochastic Network for Protein Secondary Structure Prediction," in *Proceedings of The 31st International Conference on Machine Learning (ICML)*, vol. 2, 2014, pp. 745–753.
- [37] Y. Gao, Y. Zhao, Y. Ma, and Y. Liu, "Prediction of Protein Secondary Structure Based on WS-BiLSTM Model," *Symmetry 2022, Vol. 14, Page 89*, vol. 14, no. 1, p. 89, Jan. 2022, doi: 10.3390/sym14010089.
- [38] Q. Wang *et al.*, "Prior knowledge facilitates low homologous protein secondary structure prediction with DSM distillation," *Bioinformatics*, vol. 38, no. 14, pp. 3574–3581, Jul. 2022, doi: 10.1093/bioinformatics/btac351.
- [39] S. M. Shahid, S. Ko, and S. Kwon, "Performance Comparison of 1D and 2D Convolutional Neural Networks for Real-Time Classification of Time Series Sensor Data," *International Conference on Information Networking*, vol. 2022-January, pp. 507–511, 2022, doi: 10.1109/icoin53446.2022.9687284.
- [40] S. Sasaoka *et al.*, "Comparative study of 1D and 2D convolutional neural network models with attribution analysis for gravitational wave detection from compact binary coalescences," *Physical Review D*, vol. 109, no. 4, p. 043011, Feb. 2024, doi: 10.1103/physrevd.109.043011.
- [41] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998, doi: 10.1109/5.726791.
- [42] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pp. 807–814, June. 2010.
- [43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.
- [44] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Feedforward Network," in *Deep Learning*, MIT Press, pp. 164–220, 2016. [Online]. Available: <https://www.deeplearningbook.org/contents/mlp.html>

- [45] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A Next-generation Hyperparameter Optimization Framework," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2623–2631, Jul. 2019, doi: 10.1145/3292500.3330701.
- [46] N. Parekh, A. Sen, P. Rajasekaran, J. D. D. Jayaseeli, and P. Robert, "Network Intrusion Detection System Using Optuna," *Proceedings of 5th International Conference on IoT Based Control Networks and Intelligent Systems, ICICNIS 2024*, pp. 312–318, 2024, doi: 10.1109/icicnis64247.2024.10823298.
- [47] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for Hyper-Parameter Optimization," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 24, pp. 2546–2554, 2011.
- [48] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: <http://scikit-learn.sourceforge.net>.
- [49] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, Jul. 2009, doi: 10.1016/j.ipm.2009.03.002.
- [50] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.
- [51] T. G. Dietterich, "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, Oct. 1998, doi: 10.1162/089976698300017197.
- [52] B. Zhang, J. Li, and Q. Lü, "Prediction of 8-state protein secondary structures by a novel deep learning architecture," *BMC Bioinformatics*, vol. 19, no. 1, pp. 1–13, 2018, doi: 10.1186/s12859-018-2280-5.
- [53] P. Kumar, S. Bankapur, and N. Patil, "An enhanced protein secondary structure prediction using deep learning framework on hybrid profile based features," *Appl. Soft Comput.*, vol. 86, p. 105926, Jan. 2020, doi: 10.1016/j.asoc.2019.105926.
- [54] S. Geethu and E. R. Vimina, "Protein Secondary Structure Prediction Using Cascaded Feature Learning Model," *Appl. Soft Comput.*, vol. 140, p. 110242, Jun. 2023, doi: 10.1016/j.asoc.2023.110242.
- [55] Z. Li and Y. Yu, "Protein Secondary Structure Prediction Using Cascaded Convolutional and Recurrent Neural Networks," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI-16)*, vol. 2016-January, pp. 2560–2567, Apr. 2016.
- [56] I. Drori et al., "High Quality Prediction of Protein Q8 Secondary Structure by Diverse Neural Network Architectures," in 32nd Conference on Neural Information Processing Systems (NIPS 2018), Nov. 2018, arXiv preprint: 1811.07143.
- [57] Y. Lu, "Protein Secondary Structure Prediction Using Convolutional Bidirectional GRU," *Journal of Mathematics Research*, vol. 16, no. 4, p. p11, Aug. 2024, doi: 10.5539/jmr.v16n4p11.
- [58] Z. Lyu, Z. Wang, F. Luo, J. Shuai, and Y. Huang, "Protein Secondary Structure Prediction With a Reductive Deep Learning Method," *Front. Bioeng. Biotechnol.*, vol. 9, p. 687426, Jun. 2021, doi: 10.3389/fbioe.2021.687426.

Author Biography



Annisa Rizqiana is a lecturer in the Department of Computer Science at Universitas Lambung Mangkurat (ULM), South Kalimantan, Indonesia. She earned her bachelor's degree in computer science from ULM and her master's degree in computer science from Universitas Gadjah Mada (UGM). Her research interests focus on bioinformatics, data science, and machine learning, particularly in applying computational methods to biological data and complex datasets. In teaching activities, she teaches courses that focus on mathematical concepts as the foundation of programming, including mathematical logic, discrete mathematics, relations and functions, and basic algebra to support algorithm development and computational problem solving. She is also actively involved in supervising undergraduate students and academic research.



Triando Hamonangan Saragih is currently a lecturer at the Department of Computer Science, Lambung Mangkurat University. He is deeply involved in academia, with a primary focus on various aspects of Data Science. His academic journey began with completing his undergraduate studies in Informatics at Brawijaya University, Malang, which he

successfully completed in 2016. To deepen his understanding of Computer Science, he continued his education at the master's level at the same university and earned a master's degree in 2018. His research interests center on Data Science, where he continues to develop her expertise in data analysis, predictive modeling, and the application of data-driven technologies. As an academic, he is active in scientific research and collaboration to develop innovative solutions applicable across various fields. He also contributes to the development of science through publications and academic discussions. Email: triando.saragih@ulm.ac.id Orcid ID: 0000-0003-43463323



Adryan Maulana Saputra is currently a Computer Science Student at the Faculty of Mathematics and Natural Science, Lambung Mangkurat University, having commenced his studies in 2023. He is interested in machine learning and software development, with a focus on website development. He also enjoys computer repair as both a job and a hobby. During his academic years, he developed a small computer business while learning about the world of computers.



Muhammad Itqan Mazdadi is a lecturer in the Department of Computer Science at Lambung Mangkurat University with research interests primarily in Data Science and Computer Networking. He began his academic journey at Lambung Mangkurat University, where he earned his undergraduate degree in Computer Science in 2013. To further deepen his expertise, he pursued and successfully completed a master's degree in informatics at the Islamic University of Indonesia, Yogyakarta. Currently, he serves as Secretary in the Computer Science Department at Lambung Mangkurat University, contributing to both academic and administrative responsibilities. His research focuses on advancements in data-driven technologies and network systems. As a lecturer and researcher, he actively engages in academic development, fostering innovation in his field. His dedication to education and research enables him to support students and collaborate on projects that enhance technological progress. Email: mazdadi@ulm.ac.id; ORCID ID: 0000-0002-8710-4616.



Irwan Budiman successfully finished his bachelor's degree in the Informatics Department at the Islamic University of Indonesia. Subsequently, he assumed the role of a lecturer in Computer Science at Universitas Lambung Mangkurat starting in 2008. Additionally, in 2010, he pursued a master's degree in information systems at Diponegoro University. Currently, Irwan Budiman serves as chair of the computer science study program at Universitas Lambung Mangkurat. His area of research expertise lies in Data Science.