

One Filter Is All You Need: Understanding Pattern Recognition in 1D Convolutional Neural Networks

VASSILIOS ANDREADAKIS, IOANNIS SMYRNAKIS

Dept. of Mechanical Engineering
Hellenic Mediterranean University
Estavromenos, 71410, Heraklion
GREECE

Abstract: - Pattern recognition in sequential data is a fundamental problem encountered across digital communications, cryptography, and computational biology. Although convolutional neural networks (CNNs) have demonstrated strong performance in sequence classification tasks, the mechanisms by which they achieve this remain insufficiently understood. In this work, we examine how a simple one-dimensional CNN performs binary classification on sequences containing hidden target patterns, using artificially generated data under tightly controlled conditions that isolate network behavior from domain-specific noise. We demonstrate, through both theoretical analysis and empirical evaluation, that a single convolutional filter aligned with the target pattern is sufficient for perfect classification. We further show that constraining the fully connected layer to have equal weights yields a substantial reduction in the number of parameters and extends accurate classification from sequences of 1,200 bits to 20,000 bits. The effects of sequence length and target pattern length on classification performance are also investigated. These results provide insight into the mechanisms underlying the effectiveness of shallow 1D-CNNs in sequence classification and suggest practical guidelines for the design of interpretable models.

Key-Words: - 1-d Convolutional Neural Networks, Pattern Recognition, Binary Sequence Classification, Target Pattern Detection, Interpretable Deep Learning, Parameter Reduction

Received: April 2, 2026. Revised: June 9, 2026. Accepted: July 9, 2026. Published: October 8, 2026.

1 Introduction

The detection of specific patterns within sequences (e.g. numerical strings, binary bit streams, or biological data) constitutes a central problem in numerous applied domains. In digital communications, the recognition of predefined bit sequences is essential for identifying synchronization markers and detecting transmission errors, [1],[2]. In cybersecurity, the identification of malicious patterns within network traffic serves as a critical defense mechanism, [3]. In genomics, the detection of binding motifs in DNA and protein sequences has become fundamental to modern molecular biology, [4].

Conventional approaches to such tasks have relied on rule-based methods and statistical models, [5], including Hidden Markov Models, [6], Random Forests, [7], and Support Vector Machines, [8], all of which require substantial hand-crafted feature engineering and domain-specific expertise. While these methods have proven effective within their respective scopes, they are generally limited in their capacity to extract complex, higher-order features directly from raw sequence data.

More recent studies also show that CNNs are

still highly relevant for pattern recognition and classification tasks. For example, EfficientNetV2, [9] improved the efficiency of convolutional models by reducing training cost while maintaining strong performance. ConvNeXt, [10] further showed that modern CNN architectures can still compete with transformer-based models. In addition, RepVGG, [11] demonstrated that simple VGG-style convolutional networks can achieve strong performance when combined with structural re-parameterization. These works support the idea that convolutional filters remain useful for detecting local patterns, which is the main focus of this study.

The emergence of deep learning, and in particular one-dimensional convolutional neural networks (1D-CNNs), has substantially expanded the available toolkit, [12]. By applying convolutional filters across an input sequence, these networks are capable of detecting localized patterns without the need for manual preprocessing. Applications of 1D-CNNs now span time-series classification in finance, network traffic monitoring, [3], and binary stream analysis, [13]. In computational biology, DeepBind was introduced, the first CNN-based framework to successfully predict protein-DNA and protein-RNA

binding directly from sequence data, demonstrating that shallow networks can learn meaningful binding motifs, [4]. DeepSEA was subsequently developed, a deeper architecture capable of simultaneously predicting hundreds of biological properties, [14]. A systematic evaluation of CNN design choices across 690 datasets was then conducted, finding that wider networks with additional filters generally improved performance, whereas deeper architectures provided benefit only for more challenging tasks, [15].

However, the majority of prior work has focused on improving predictive accuracy rather than on elucidating the mechanisms through which these networks operate. The present study addresses this gap. We employ a deliberately simple 1D-CNN architecture and a controlled, artificially generated dataset in order to isolate the network's behavior from confounding factors. This approach enables the derivation of insights that would otherwise be obscured by biological or experimental noise.

More specifically, this work investigates several interrelated questions: the minimum number of convolutional filters required for successful classification, the necessary size of the training set, the maximum sequence length the model can accommodate, and the influence of target pattern length on performance. A principal finding is that classification is accomplished by a single filter and that the structure of this filter directly reveals the target pattern.

2 Triplet Detection: A Baseline Experiment Using 1D-CNN

2.1 Sequence Generation and Class Definition

To investigate the learning behavior of a 1D-CNN in a controlled setting, we designed a binary classification task based on triplet detection. All possible binary triplets of size $k = 3$, from "000" to "111" (eight in total), were enumerated. These triplets were then randomly concatenated to form sequences of $n = 20$ triplets each, yielding sequences of 60 bits. One specific triplet was designated as the target pattern.

Two classes of sequences were generated. In Class 1, the target triplet appears exactly once at a random position within the sequence. In Class 0, the target triplet is entirely absent. The objective of the classification task is to identify whether a given sequence includes the target pattern and also to reveal the which pattern is.

2.2 Architecture and Training Configurations

The architecture of the proposed 1D-CNN is intentionally minimal, consisting of a single

convolutional layer with filter size 3 and stride 3, such that each triplet is evaluated independently without overlap. A Rectified Linear Unit (ReLU) activation function is applied following the convolution. The resulting feature map is then passed to a fully connected output layer comprising a single neuron with a sigmoid activation function. An output value greater than or equal to 0.5 indicates Class 1; a value below 0.5 indicates Class 0. The model was trained using binary cross-entropy as the loss function and the Adam optimizer for backpropagation. This architecture was selected to prioritize interpretability: a single convolutional layer permits direct examination of filter-input interactions. Specifically, the convolution filter F acts on a triplet $\mathbf{m} = (m_1, m_2, m_3)$ as

$$F(\mathbf{m}) = \mathbf{w}^c \cdot \mathbf{m} + b^c \quad (1)$$

where $\mathbf{w}^c$ denotes the filter weights and b^c the convolution bias. In conjunction with the ReLU function, the bias acts as a threshold: inner products at or below $-b^c$ are set to zero, while values exceeding this threshold pass through unchanged. The output neuron computes a weighted sum of the filtered values

$$L(\mathbf{z}) = \sum_{i=1}^n w_i z_i + b \quad (2)$$

where $z_i = \max(0, F(\mathbf{m}^i))$. A sigmoid function applied to this sum produces the final classification output. The total number of trainable parameters is $4 * f + n + 1$, where f denotes the number of filters and n the number of triplets in the sequence.

2.3 Training and Observations

The model was trained with 4, 8, and 16 randomly initialized filters, using training sets of 100, 200, and 300 samples per class, over 500 epochs. Consistent achievement of 100% test accuracy on 60-bit sequences required 16 filters and 300 training samples for each class. Configurations with fewer filters or smaller training sets yielded inconsistent results.

This outcome raises a natural question regarding whether all 16 filters are necessary for classification. Detailed inspection of the trained models revealed that this is not the case. The number of randomly initialized filters was varied among 4, 8, and 16, while the training-set size per class was varied among 100, 200, and 300 samples, with an equal number of samples assigned to Class 0 and Class 1. Their effects on classification accuracy were then evaluated (Fig. 1). Four consistent observations emerged from this analysis:

First, in the majority of cases, only a single filter was responsible for the classification. The remaining filters either captured irrelevant features or remained effectively inactive. Second, the functional filter consistently mirrored the target pattern, i.e. its learned weights closely corresponded to the structure of the target. This observation is particularly significant, as the target pattern is not known a priori; the filter thus serves as a means of identifying it. Third, the absolute values of the functional filter's weights were approximately equal. Fourth, the weights of the fully connected layer also converged to approximately equal values.

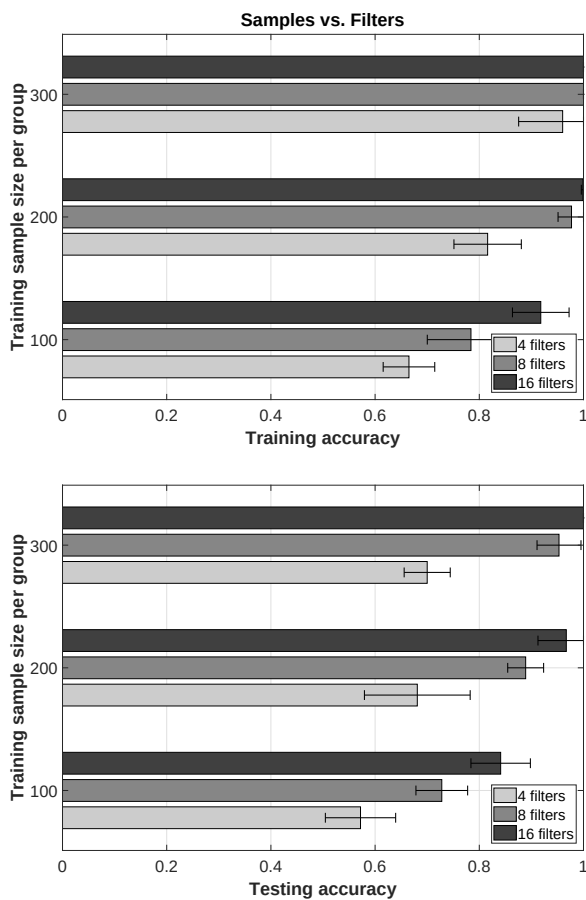


Figure 1: Training (up) and testing (down) accuracy vs. number of samples and filters. Source: created by the authors.

Then, why use multiple filters? The necessity of employing multiple filters during training can therefore be attributed to random initialization. A larger number of filters increases the probability that at least one will be initialized sufficiently close to the target pattern for gradient descent to converge to the correct solution. With fewer filters, the likelihood of no filter occupying a favorable region of the weight space increases, resulting in classification failure.

2.4 Operation of a 1D-CNN Filter

To illustrate the mechanism concretely, consider the input sequence [101, 110, 000, 111, 001, 010, 011, 100], where the objective is to detect the pattern [101]. Training produced a filter with weights [+1.163, -1.159, +1.169] and bias [-1.174]. Table 1 presents the result of applying this filter to each triplet in the sequence. As shown in Table 1, following the

Idx	Input	Dot product	After bias	ReLU output
1	[101]	+2.332	+1.158	1.158
2	[110]	+0.004	-1.170	0
3	[000]	+0.000	-1.174	0
4	[111]	+1.173	-0.001	0
5	[001]	+1.169	-0.005	0
6	[010]	-1.159	-2.333	0
7	[011]	+0.010	-1.164	0
8	[100]	+1.163	-0.011	0

Table 1: Convolution output for filter [1.163, -1.159, 1.169] with bias [-1.174], applied to each triplet in the sequence. Only the target pattern [101] yields a nonzero ReLU output. Source: created by the authors.

application of ReLU, all outputs are reduced to zero with the exception of the one corresponding to the target pattern [101]. The filter has effectively learned to activate exclusively in the presence of the target. It is worth noting that several non-target triplets (indices 4, 5, and 8) produce bias-adjusted values that are only marginally negative. While this poses no difficulty in a noise-free setting, it does indicate a potential vulnerability to noise perturbation, a point that warrants further investigation.

3 From Triplets to General Pattern Search

In the previous example, the sequence was partitioned into non-overlapping triplets, each scanned with stride 3. A more general formulation considers the case where the target pattern may straddle adjacent triplets, that is, it may occur at any arbitrary position within the sequence. This more general case is reducible to the non-overlapping formulation described above.

The reduction proceeds as follows. A new sequence of separated triplets is constructed from the original: for each position i in the original sequence, a triplet is formed from the element at position i and the two subsequent elements. A stride-1 search in the original sequence is thereby equivalent to a stride-3 search in this derived sequence. This construction generalizes to multiplets of arbitrary size k . Moreover, it extends to sequences whose elements are drawn from a finite alphabet of p symbols rather

than binary digits. By assigning a unique binary code of length $q \approx \log_2 p$ to each symbol, any such sequence can be converted to a binary string, and the classification framework described here can be applied directly.

More precisely, if each symbol is encoded using a $q - \text{bit}$ binary representation, the original sequence of n symbols is transformed into a binary string of length nq . A $\text{stride} - k$ pattern search in the original sequence corresponds to a $\text{stride} - kq$ search for a $\text{length} - kq$ binary pattern in the derived string. Consequently, any classification task defined over sequences with elements from a finite alphabet (like DNA sequences, or other symbolic encodings) can be reduced to a binary search problem of the form studied in this work. The analysis of binary sequences with stride equal to the target pattern length is therefore fully general.

3.1 Why a Single Filter is Sufficient for Classification

The question arises whether an effective convolutional filter can be constructed in a principled way, rather than relying on training to discover one from random initialization. In the controlled setting considered here, the answer is positive. By re-encoding the binary input such that 0 is replaced by -1 , every k -bit pattern has the same norm, equal to $\sqrt{k}$. As a result, the inner product between a filter and an input pattern depends on their angular alignment. This means that the response is maximized when the filter is aligned with the target pattern. This observation leads to the following theorem.

Theorem 1. Let $\mathbf{m}^* \in \{0, 1\}^k$ be a target binary pattern, and let $\tilde{\mathbf{m}}^* \in \{-1, +1\}^k$ be its re-encoded form, where each 0 is replaced by -1 and each 1 is kept as $+1$. Then there exists a filter $\mathbf{w}^c$ and a bias b^c such that, for any input pattern $\tilde{\mathbf{m}} \in \{-1, +1\}^k$,

$$\text{ReLU}(\mathbf{w}^c \cdot \tilde{\mathbf{m}} + b^c) > 0$$

if and only if

$$\tilde{\mathbf{m}} = \tilde{\mathbf{m}}^*.$$

Proof. Choose the filter to be equal to the re-encoded target pattern,

$$\mathbf{w}^c = \tilde{\mathbf{m}}^*.$$

Since each entry of $\tilde{\mathbf{m}}$ is either -1 or $+1$, every k -bit pattern has norm

$$\|\tilde{\mathbf{m}}\| = \sqrt{k}.$$

The inner product between the filter and an input pattern can therefore be written as

$$\mathbf{w}^c \cdot \tilde{\mathbf{m}} = \|\mathbf{w}^c\| \|\tilde{\mathbf{m}}\| \cos \theta,$$

where θ is the angle between the two vectors. This value is maximized when $\theta = 0$, which occurs when the input pattern is identical to the target.

More directly, if $\tilde{\mathbf{m}} = \tilde{\mathbf{m}}^*$, then

$$\mathbf{w}^c \cdot \tilde{\mathbf{m}} = k.$$

If $\tilde{\mathbf{m}} \neq \tilde{\mathbf{m}}^*$, then at least one position differs, so the inner product is at most

$$k - 2.$$

Now choose the bias as

$$b^c = -(k - 1).$$

For the target pattern,

$$\mathbf{w}^c \cdot \tilde{\mathbf{m}}^* + b^c = k - (k - 1) = 1 > 0.$$

For any non-target pattern,

$$\mathbf{w}^c \cdot \tilde{\mathbf{m}} + b^c \leq (k - 2) - (k - 1) = -1 < 0.$$

Therefore, after applying ReLU, only the target pattern produces a nonzero response. Hence,

$$\text{ReLU}(\mathbf{w}^c \cdot \tilde{\mathbf{m}} + b^c) > 0 \quad \text{if and only if} \quad \tilde{\mathbf{m}} = \tilde{\mathbf{m}}^*.$$

This proves the claim. $\square$

This result also has practical significance beyond its theoretical implications. Because the optimal filter aligns with the target pattern, the target can be identified directly from the learned filter weights. The trained CNN thus not only performs classification but also reveals the structure of the pattern it has detected.

4 Performance Degradation with Increasing Sequence Length

Having established the classification mechanism for short triplet sequences, we now consider its behaviour at greater scale. The pattern length was increased to $k = 20$ bits, and the sequence length was varied from 400 bits ($n = 20$ patterns) to 2,000 bits ($n = 100$ patterns). The number of possible 20-bit patterns exceeds 10^6 , yielding a substantially more challenging classification environment. Training was conducted using 1,000 sequences per class, 4 convolutional filters with stride 20, and 1,500 epochs with L2 loss and the Adam optimizer. The model achieved perfect test accuracy for sequences of 400 and 800 bits (Fig. 2). However, performance degraded for longer sequences: by 1,200 bits, testing accuracy had begun to decline, and for sequences exceeding 1,400 bits it fell well below 0.95. This degradation is attributable to the excessive number of parameters relative to the available training data,

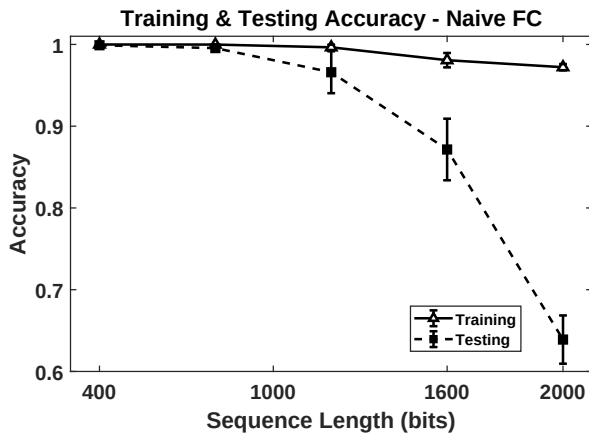


Figure 2: Training and testing accuracy of a simple CNN with 4 filters on sequences of varying length, demonstrating that classification fails for sequences longer than 800 bits despite perfect accuracy on short inputs. Source: created by the authors.

leading to overfitting. Hence, the primary objective would be to reduce the number of parameters.

At this point, the aforementioned observation that *the weights of the fully connected layer also converged to approximately equal values* becomes relevant: in well-trained models, the weights of the fully connected layer converge to approximately equal values. This behaviour has a clear explanation: since the target pattern is equally likely to appear at any position within the sequence, there is no basis for the network to assign different importance to different positions. The fully connected weights should therefore exhibit translational invariance, and the trained network confirms this expectation.

5 Exploiting Translational Symmetry in the Output Layer

Motivated by the last observation, we modified the architecture to constrain all weights in the fully connected layer (*FC*) to be equal by construction. This modification has a substantial impact on the parameter count: instead of $n + 1$ parameters per filter in the *FC* layer (n position-specific weights and one bias), the constrained model requires only 2 (one shared weight and one bias). For a sequence comprising 1,000 patterns, this represents a reduction from 1,001 parameters to 2.

All remaining hyperparameters and the training set size were held constant; only the sequence length was varied. The improvement in classification performance was substantial (Fig. 3). With equal *FC* weights, the network maintained accuracy at or above 0.95 for sequences up to 18,000 bits in length, as shown in Table 2. In contrast, the naive model

achieved comparable accuracy only for sequences of 1,200 bits or shorter (Fig. 2). This represents an approximately sixteenfold increase in the maximum classifiable sequence length, achieved through the removal of parameters rather than the addition of further complexity.

Model variant	Max length (acc ≥ 0.95)	FC parameters per filter
Naive FC	$\sim 1,200$ bits	$n + 1$
Equal-weight FC	$\sim 18,000$ bits	2

Table 2: Comparison of the naive and equal-weight architectures. The equal-weight constraint yields a substantial extension of the maximum classifiable sequence length. Source: created by the authors.

An additional benefit of this approach relates to model interpretability. Under the equal-weight constraint, training consistently yields a single dominant filter that aligns with the target pattern. The resulting architecture is therefore minimal: a single convolutional layer performs pattern detection, followed by a one-neuron output layer in which all weights are identical. Although multiple filters remain useful at initialization to provide gradient descent with a favorable starting configuration, the trained network invariably converges to a single effective filter. The equal-weight constraint thus not only improves scalability but also enables the direct identification of the target pattern from the learned weights. (Fig. 3)

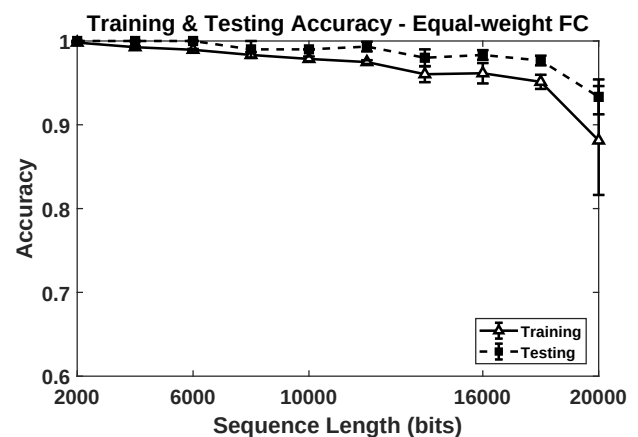


Figure 3: Training and testing accuracy after introducing equal weights in the fully connected layer, enabling accurate classification of much longer sequences compared to the baseline model. Source: created by the authors.

6 Short vs. Long Patterns: A Trade-Off Analysis

A further question of interest concerns the relationship between the length of the target pattern and the network's ability to detect it reliably. To investigate this, the equal-weight model was trained with 600 samples per class, while the sequence length was varied from 2,000 to 10,000 bits and the target pattern length from 3 to 63 bits. The upper limit of 63 bits was chosen on the grounds that, beyond this length, the number of possible patterns (2^{63}) becomes so large that the probability of a specific pattern occurring by chance is negligible. For relatively short sequences comprising 100 concatenated patterns, the network achieved near-perfect accuracy across all pattern lengths examined. With 200 patterns, accuracy remained consistently above 0.90. However, when the number of patterns was increased to 300 or more, a notable decline in performance was observed for shorter target patterns, particularly in the range of 6 to 18 bits. This behavior is consistent with the reduced distinctiveness of short patterns: when the target is brief, a greater number of non-target patterns will share structural similarity with it, increasing the frequency of near-matches and thereby confusing the classifier.

At the opposite extreme, longer target patterns exhibited a different form of instability, characterized by erratic fluctuations in accuracy. This is attributable to data sparsity: as the pattern length increases, the combinatorial space of possible patterns grows exponentially, and the training set becomes increasingly unlikely to contain a sufficiently diverse set of examples for robust filter learning.

7 Discussion and Conclusion

This paper aimed to provide a mechanistic account of how a simple 1D-CNN performs sequence classification. The use of artificially generated binary sequences under controlled conditions allowed the isolation of the network's behavior from confounding factors that typically accompany real-world datasets.

The principal finding is that classification is accomplished through the construction of a single convolutional filter that responds exclusively to the target pattern and produces zero output for all others. This filter aligns with the target, rendering the learned representation directly interpretable: the target pattern can be recovered from inspection of the filter weights. While only one such filter is theoretically required, multiple randomly initialized filters are necessary in practice to ensure convergence to the correct solution during training.

A second contribution concerns the exploitation

of the translational symmetry inherent in the classification task. By constraining the fully connected layer weights to be equal, reflecting the fact that the target pattern may appear with equal probability at any position, the maximum classifiable sequence length was extended from approximately 1,200 bits to 20,000 bits, accompanied by a substantial reduction in parameters. This result illustrates how understanding the mechanisms of the network can inform architectural design and lead to improved performance.

In addition, pattern length experiments demonstrated that the network is effective across a broad range of target lengths, with performance declining at the extremes due to pattern ambiguity (short patterns) and data sparsity (long patterns), respectively.

Several directions for future work emerge from these findings. The extension of this framework to multi-pattern detection, in which the network must identify one of several possible targets, is a natural next step. The applicability of symmetry-based parameter reduction to other architectures, including 2D-CNNs for image classification, also warrants investigation. Finally, the sensitivity of the working filter to near-threshold patterns suggests potential vulnerability to noise, which we are currently addressing through the development of trainable denoising layers.

Although the use of artificially generated sequences is essential for isolating and interpreting the classification mechanism, it also represents an important limitation of the present study. The proposed construction assumes that the target pattern is well defined, appears in a controlled manner, and is not corrupted by noise. In real-world sequence data, these assumptions may not stand, as patterns may occur with insertions, deletions, shifts, or partial corruption, and non-target regions may contain near-matches that produce activations close to the decision threshold. As illustrated by the triplet example, several non-target inputs can yield pre-activation values only slightly below zero. In the presence of noise, such cases could become false positives. Conversely, noise affecting the true target pattern could reduce its activation and lead to false negatives. Therefore, while a single aligned filter is sufficient in the ideal conditions considered here, real-world applications may require additional mechanisms, such as multiple filters representing pattern variants, regularization or denoising layers that explicitly account for uncertainty.

Acknowledgment:

This publication is financed by the Project "Strengthening and optimizing the operation of

MODY services and academic and research units of the Hellenic Mediterranean University”, funded by the Public Investment Program of the Greek Ministry of Education and Religious Affairs.

Declaration of Generative AI and AI-assisted Technologies in the Writing Process

The authors wrote, reviewed and edited the content as needed and they have not utilised artificial intelligence (AI) tools. The authors take full responsibility for the content of the publication.

References:

- [1] Cao, Congzhe, Duanshun Li, and Ivan Fair, Deep learning-based decoding of constrained sequence codes, *IEEE Journal on Selected Areas in Communications*, Volume 37, Issue 11, 2019, pp. 2532-2543, <https://doi.org/10.1109/JSAC.2019.2933954>.
- [2] Tandler, Daniel, Dörner, Sebastian, Cammerer, Sebastian, and ten Brink, Stephan, On recurrent neural networks for sequence-based processing in communications, *In 53rd Asilomar Conference on Signals, Systems, and Computers of IEEE*, 2019, pp. 537-543, <https://doi.org/10.1109/IEEECONF44664.2019>.
- [3] Qazi, Emad Ul Haq, Abdulrazaq Almorjan, and Tanveer Zia, A one-dimensional convolutional neural network (1D-CNN) based deep learning system for network intrusion detection, *Applied Sciences*, Volume 12, Issue 16, 2022, 7986, <https://doi.org/10.3390/app12167986>.
- [4] Alipanahi, Babak, DeLong, Andrew, Weirauch, Matthew T., and Frey, Brendan, Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning, *Nature Biotechnology*, Volume 33, Issue 8, 2015, pp. 831-838, <https://doi.org/10.1038/nbt.3300>.
- [5] Lin, Wenyi, Hasenstab, Kyle, Moura Cunha, Guilherme, and Schwartzman, Armin, Comparison of handcrafted features and convolutional neural networks for liver MR image adequacy assessment, *Scientific Reports*, Volume 10, Issue 1, 2020, 20336, <https://doi.org/10.1038/s41598-020-77264-y>.
- [6] Yoon, Byung-Jun, Hidden Markov models and their applications in biological sequence analysis, *Current Genomics*, Volume 10, Issue 6, 2009, pp. 402-415, <https://doi.org/10.2174/138920209789177575>.
- [7] Chen, Xi, and Ishwaran, Hemant, Random forests for genomic data analysis, *Genomics*, Volume 99, Issue 6, 2012, pp. 323-329, <https://doi.org/10.1016/j.ygeno.2012.04.003>.
- [8] Sonnenburg, Soren, Rätsch, Gunnar, and Schäfer, Christin, *Learning interpretable SVMs for biological sequence classification*, ISBN: 978-3-540-25866-7, Springer, Berlin, Heidelberg, 2005, pp. 389-407, <https://doi.org/10.1007/11415770>.
- [9] Tan, Mingxing, and Le, Quoc, EfficientNetV2: Smaller models and faster training, *International conference on machine learning*, 2021, pp. 10096-10106, <https://doi.org/10.48550/arXiv.2104.00298>.
- [10] Liu, Zhuang, Mao, Hanzhi, Wu, Chao-Yuan, Feichtenhofer, Christoph, Darrell, Trevor, and Xie, Saining, A ConvNet for the 2020s, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11966-11976, <https://doi.org/10.1109/CVPR52688.2022.01167>.
- [11] Ding, Xiaohan, Zhang, Xiangyu, Ma, Ningning, Han, Jungong, Ding, Guiguang, and Sun, Jian, RepVGG: Making VGG-style ConvNets great again, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13728-13737, <https://doi.org/10.1109/CVPR46437.2021.01352>.
- [12] Kiranyaz, Serkan, Avci, Onur, Abdeljaber, Osama, Ince, Turker, Gabbouj, Moncef, and Inman, Daniel J., 1D convolutional neural networks and applications: A survey, *Mechanical Systems and Signal Processing*, Volume 151, 2021, 107398, <https://doi.org/10.1016/j.ymsp.2020.107398>.
- [13] Lin, Wei-Cheng, and Yeh, Yi-Ren, Efficient malware classification by binary sequences with one-dimensional convolutional neural networks, *Mathematics*, Volume 10, Issue 4, 2022, 608, <https://doi.org/10.3390/math10040608>.
- [14] Zhou, Jian, and Troyanskaya, Olga G., Predicting effects of noncoding variants with deep learning-based sequence model, *Nature Methods*, Volume 12, Issue 10, 2015, pp. 931-934, <https://doi.org/10.1038/nmeth.3547>.
- [15] Zeng, Haoyang, Edwards, Matthew D., Liu, Ge, and Gifford, David K., Convolutional neural network architectures for predicting DNA-protein binding, *Bioinformatics*, Volume 32, Issue 12, 2016, pp. i121-i127, <https://doi.org/10.1093/bioinformatics/btw255>.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

Vassilios Andreadakis conceived and implemented the study, performed the experiments, analyzed the results, and drafted the manuscript.

Ioannis Smyrnakis provided supervision, guidance in the interpretation of results, and contributed to manuscript review and editing.

Both authors read and approved the final manuscript.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflicts of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0

https://creativecommons.org/licenses/by/4.0/deed.en_US