Justification and Efficiency of Low-parameter Architectures of Artificial Neural Networks for Recognition of the Ukrainian Dactyl Alphabet in Conditions of Limited Computing Resources

Olena Liubymenko[1,2 *,†] , Ilya Ivanchenko[1,†] Oleksandr Shtepa [1,†]

[1] *Donetsk National Technical University, 76, Sambirska, St., Drohobych, Lviv region, 82100,* Ukraine
[2] *Lutsk National Technical University, 75, Lvivska street, Lutsk, Volyn region, 43018, Ukraine*[1]

Abstract

The paper investigates the effectiveness of using low-parameter architectures of direct propagation artificial neural networks for real-time recognition of Ukrainian dactyl alphabet gestures. Unlike resource-intensive deep learning frameworks, the proposed algorithmic pipeline is based on basic matrix calculations of the NumPy library. The development combines the extraction of three-dimensional coordinates of key points of the hand using the MediaPipe Hands toolkit, algebraic 3D-normalization of features and classification using a two-layer perceptron. On the generated dataset of 16513 samples, an overall classification accuracy of 98.18% was achieved on a completely isolated test sample. A detailed spatial analysis of the residual error was performed. The feasibility of using such a model for deployment on hardware-limited platforms was analytically proven due to the radical minimization of RAM overhead.

Keywords

computer vision, gesture recognition, artificial neural network, MediaPipe, Ukrainian dactyl alphabet, multimodal interface, unitary coding, inclusive technologies.

1. Introduction

The evolution of human-computer interaction (HCI) interfaces indicates a stable transition from traditional contact input devices to contactless intelligent control systems. The integration of computer vision algorithms for sign language recognition is one of the most important steps in building inclusive educational and communication ecosystems for people with hearing impairments. This issue has acquired particular social importance and scientific relevance in Ukraine. As O. Bilanova, L. Ivashchenko, and S. Kulbida indicate in their study of intermodal contacts of Ukrainian Sign Language (USSL), the events of 2022–2026 stimulated intensive terminologization, accelerated standardization of new gestures, and the rejection of lexemes shared with the aggressor language [1]. This phenomenon of lexical shift requires the urgent implementation of mobile and flexible technological solutions to support the updated linguistic paradigm in times of crisis.

In addition to sociolinguistic challenges, a serious engineering difficulty is the visual-morphological specificity of the Ukrainian Dactylic Alphabet (UDA). Unlike American Sign Language (ASL), which consists of 26 Latin graphemes, UDA operates with 33 Cyrillic letters. Such an expansion of the feature space leads to a noticeable narrowing of interclass distances and a densification of spatial clusters.

Analysis of the morphological structure of UDA allows us to distinguish two fundamental classes of destructive factors that complicate the work of feedforward network classifiers:

[1]*RS-2026: Resilient Systems: Secure Digital Technologies and Critical Infrastructure, June 30, 2026, Drohobych, Ukraine*
* Corresponding author.
† These authors contributed equally.
olena.liubymenko@donntu.edu.ua (O. Liubymenko); illia.ivanchenko.kita@donntu.edu.ua (I. Ivanchenko), oleksandr.shtepa@donntu.edu.ua (O.Shtepa)
0000-0002-5935-6891 (O. Liubymenko); 0009-0001-1298-9771 (I. Ivanchenko); 0000-0002-3860-4331 (O.Shtepa)

1. Microstructural coordinate differentiations (Class A) - configurations where letters differ by a minimal shift of one finger or the appearance of a small spatial element in a fixed plane. For example, a "tail" in the lower part of the letter "ІІЧ" relative to "ІІ", or a slight displacement of the thumb for "Г" compared to "Г". Overcoming this problem requires the mathematical model to be highly accurate in affine transformations and scaling of local features.

2. Spatial occlusion and self-overlap (Class B) — gestures in which differential characteristics (such as the position of the little finger or the dynamics of the hand) are physically overlapped by other parts of the palm during frontal monocular shooting. Prominent examples are the pairs “И”—“Ю”, “Д”—“К” or “Т”—“М”.

Traditional commercial systems often level out this fine division, trying to overcome both types of interference exclusively by increasing the massiveness of neural networks [2]. For a long time, the global scientific discourse suffered from the so-called “ASL-bias”, when most open architectures were optimized exclusively for the American dactyl [2]. Today, existing high-precision classifiers operate mainly on the basis of heavy deep learning frameworks (TensorFlow, PyTorch) [3]. Their inference generates significant memory overhead and requires powerful graphics processing units (GPU) [7]. Such high hardware requirements hinder the mass deployment of systems on budget peripheral devices (Edge Devices) in educational institutions of Ukraine.

Therefore, the current study aims to solve the outlined problem: to evaluate the efficiency and stability of a low-parameter deep neural network (DNN), implemented exclusively by means of low-level matrix operations of the NumPy library [8]. The proposed architecture is focused on stable identification of 33 spatial gestures of the UDA in real time using standard RGB cameras and exclusively the central processing unit (CPU).

2. Related Work and Problem Statement

Modern approaches to cybersecurity management emphasize the importance of systematic vulnerability management within Information Security Management Systems (ISMS). In addition to the ISMS requirements defined by ISO/IEC 27001 and ISO/IEC 27005, the ISO/IEC 27002 standard provides recommendations for implementing technical and organizational security controls, including those related to vulnerability management, incident response, and security monitoring [4]. Practical aspects of incident handling and cyber threat response are also addressed in NIST guidelines (SP 800-61, SP 800-137) [5,6]. In parallel, contemporary approaches to quantitative risk assessment, such as the FAIR (Factor Analysis of Information Risk) model, consider both the likelihood of threat realization and its impact on assets [7].

The development of sign language translation systems has overcome a significant evolution from bulky hardware solutions (such as the CyberGlove sensor gloves) and infrared depth scanners (Microsoft Kinect) to flexible image processing algorithms. The integration of Google's open MediaPipe Hands toolkit has significantly democratized the development of artificial intelligence, providing the ability to localize the spatial skeleton of the hand without rigidly binding to the background or lighting. However, since MediaPipe outputs only raw coordinates, developers need to integrate additional algebraic classifiers. Recognition of localized sign languages. In world practice, methods for adapting MediaPipe features to the specifics of national languages are actively being studied. In particular, M. Rodriguez and co-authors implemented a Mexican Sign Language (MSL) classifier without forced hand framing, achieving an accuracy of 92% based on Support Vector Machine (SVM) for static signs [2]. A. Cardano and colleagues proposed a system for recognizing Filipino Sign Language (FSL), where MediaPipe is supplemented with recurrent GRU networks to track interframe dynamics [3]. Another approach was demonstrated by the group of Z. Alejo, who, thanks to transfer learning methods, brought the accuracy of FSL identification to 98.75% [4]. The problem of Indian Sign Language (ISL) was investigated by A. Nadaf, applying multimodal fusion of camera coordinates and electromyography (sEMG) signals using Parallel SENet (accuracy 97.1%) [5]. However, the use of external sensors contradicts our criterion of zero cost of infrastructure deployment.

Spatial normalization and computational complexity. The importance of reprocessing MediaPipe spatial vectors was empirically confirmed by M. Gil-Martín. His research proves that linear transfer of the center of the Cartesian system to the wrist node ensures feature invariance to linear displacements in the frame [6]. In the context of inference optimization, J. Fernandez's research proves the concept of "framework tax" - for small amounts of input data, the fixed overhead of heavy frameworks dominates the time of the calculations themselves [9]. This ideally confirms the thesis of critical initialization delays and justifies the use of NumPy matrix calculations for inference optimization. The concept of "CPU-only" deployment based on linear algebra libraries, as demonstrated in the works of H. Jin [10] and S. Radcliffe [11], demonstrates a radical reduction in RAM overhead, allowing inference to be performed with low asymptotic complexity.

The state of recognition of Ukrainian dactyl. National developments have long been fragmentary. A. Bauer initiated the creation of an offline corpus in the laboratory conditions of the University of Cologne [12]. An applied aspect is highlighted in the work of O. Marmur, who used a heavy CNN+RNN for the classification of 108 words [13]. The most relevant benchmark is the study of A. Glybovets and M. Bikchentaev: recognition of 33 letters of the UDA based on MediaPipe and LSTM networks. Analyzing a time series of 65 frames, the authors recorded an accuracy of 98.4% [14]. The current study proves that for static dactyls, the preservation of long-term temporal memory is redundant: processing single frames with a multilayer perceptron (MLP) in combination with a UX filter provides higher performance and comparable recognition quality.

3. Methodology and mathematical apparatus of the study (Methods)

A multi-level computational pipeline was designed to build an effective sign language translation system. The processing starts with capturing the BGR video stream (via OpenCV), converting it to RGB, and localizing the anatomical parameters of the palm.

3.1. Algebraic formalism of spatial 3D feature normalization

Data from the MediaPipe Hands detector form a set of 21 key points (landmarks). Point P_i is described by a three-dimensional vector in the Cartesian system:

$$P_i = (x_i, y_i, z_i), i \in [0,20] \quad (1)$$

where i is the joint index, and P0 denotes the anatomical center of the wrist.

In order to ensure the stability of the classifier to the position of the hand relative to the camera and individual anthropometry, the coordinates undergo a two-stage normalization to obtain scale and translational invariance [6]. In the first stage, spatial centering is performed. The origin of coordinates is transferred to the wrist P0. The displacement vector for each node is calculated as:

$$x_i' = x_i - x_0, y_i' = y_i - y_0, z_i' = z_i - z_0 \quad (2)$$

In the second stage, scale standardization is applied. Isotropic scaling eliminates the effect of the distance to the lens. The global coefficient S is calculated - the maximum absolute value among all frame coordinates:

$$S = \max_{j \in [0,20]} \left(max\left(|x_j'|, |y_j'|, |z_j'| \right) \right) \quad (3)$$

The final normalized features take the form:

$$x_i{}^{norm} = \frac{x_i{}'}{S}, \qquad y_i{}^{norm} = \frac{y_i{}'}{S}, \qquad z_i{}^{norm} = \frac{z_i{}'}{S} \quad (4)$$

They are reduced to a flat row vector X of dimension 1 × 63:

$$X = [x_0{}^{norm}, y_0{}^{norm}, \dots, z_{20}{}^{norm}] \in R^{1 \times 63} \quad (5)$$

3.2. Topology and parametric initialization of ANN

Classification is performed by a densely connected network (MLP) with an architecture of 63 ⟶ 128 ⟶ 64 ⟶ 33. The number of hidden neurons is optimized to create an information bottleneck that cuts off excess spatial noise MediaPipe.

For the training process, single vectors are combined into a mini-batch input feature matrix X ∈ R ^ {m × 63}, where the batch dimension is fixed at m = 128. The choice of such a subset size is justified by the need to stabilize stochastic gradient descent (SGD) and maximize the load on vector registers of modern CPUs. The weight matrices Wl are initialized strictly according to the He Initialization method, which mathematically guarantees the preservation of the signal dispersion between layers when using ReLU activation functions, avoiding the problem of explosion or fading gradients [7]:

$$W_l \sim \mathcal{N}(0, {\sigma_l}^2), \;\; \sigma_l = \sqrt{\left(\frac{2}{N_{l-1}}\right)} \tag{6}$$

where N $_{l-1}$ is the dimension of the input of layer l. The displacement vectors b_l are initialized with zeros. All calculations are performed using the float32 data type (single precision), which provides an optimal compromise between the accuracy of representing real numbers and the speed of performing matrix operations on x86_64 and ARM architectures.

3.3. Forward Pass Modeling

The inference pipeline is implemented using basic NumPy tensor multiplications [8]. Forward pass for a matrix X is written as a system of matrix equations:

$$Z_1 = X \cdot W_1 + b_1, A_1 = max(0, Z_1) \tag{7}$$
$$Z_2 = A_1 \cdot W_2 + b_2, A_2 = max(0, Z_2)$$

where $W_1 \in R^{\{63 \times 128\}}$, Z_1, $A_1 \in R^{\{m \times 128\}}$; $W_2 \in R^{\{128 \times 64\}}$, Z_2, $A_2 \in R^{\{m \times 64\}}$. The output layer forms a posterior probability distribution for 33 classes through a normalized exponential Softmax function:

$$Z_3 \; = \; A_2 \; \cdot \; W_3 \; + \; b_3 \tag{8}$$
$$\hat{Y} = Softmax(Z_3) \Rightarrow \hat{Y}_{i,k} = \frac{e^{Z_{3,i,k}}}{\sum_j e^{Z_{3,i,j}}}$$

The estimate of the discrepancy with the true label matrix Y (unitary One-Hot Encoding) is calculated through the loss minimization function — categorical cross-entropy [7]:

$$L = -\frac{1}{m} \cdot \sum_{i=1}^{m} \sum_{k=1}^{33} Y_{i,k} \cdot ln\left(\hat{Y}_{i,k}\right) \tag{9}$$

3.4. Mathematical formalism of error backpropagation

The optimization of the objective function L is based on chain rule differentiation [7, 10, 11]. The gradient for the linear combination of the output layer takes the form:

$$LdZ_3 = \frac{1}{m} \cdot \left(\hat{Y} - Y\right) \in \mathbb{R}^{m \times 33} \tag{10}$$

Gradients of the parameters of the output layer l=3, where 1mT denotes the unit vector for summation along the batch axis:

$$dW_3 \; = \; {A_2}^T \; \cdot \; dZ_3, \;\; db_3 \; = \; {1_m}^T \; \cdot \; dZ_3 \tag{11}$$

Error propagation to the hidden layers (taking into account the derivative of the ReLU activation):

$$dA_2 \; = \; dZ_3 \; \cdot \; {W_3}^T \tag{12}$$
$$dZ_2 = dA_2 \odot \mathbb{I}(Z_2 > 0)$$

$$dW_2 = {A_1}^T \cdot dZ_2,\ db_2 = {1_m}^T \cdot dZ_2$$
$$dA_1 = dZ_2 \cdot {W_2}^T$$
$$dZ_1 = dA_1 \odot \mathbb{I}(Z_1 > 0)$$
$$dW_1 = X^T \cdot dZ_1,\ db_1 = {1_m}^{\boldsymbol{T}} \cdot dZ_1$$

Weights are adjusted at each iteration:

$$W_l = W_l - \alpha_t \cdot dW_l,\ b_l = b_l - \alpha_t \cdot db_l \quad (13)$$

The dynamic learning rate αt decreases according to the geometric law (α_0 = 0.001, γ = 0.95, step 30 epochs):

$$\alpha_t = \alpha_0 \cdot \gamma^{\lfloor t/30 \rfloor} \quad (14)$$

3.5. Probabilistic decision-making model

To integrate the classifier into the final multimodal interface, a frame duplication filtering algorithm with a hard probabilistic threshold Θ = 0.7 is implemented:

$$c_* = max(\hat{Y}), k_* = argmax(\hat{Y}) \quad (15)$$

$$Decision(t) = k_*, \quad if \quad c_* \geq 0.7; \quad \text{otherwise} \quad \emptyset, \quad (16)$$

where symbol k* enters the output buffer only if it differs from the previously stored character, and the gesture configuration is continuously held by the user for at least 2.0 seconds.

3.6. Generalized algorithm of system operation

The general processing pipeline is formalized into a single algorithmic block to ensure the reproducibility of the experiment and confirmation of constant spatial complexity.

Algorithm 1. Resource-efficient pipeline for spatial data processing and ANN inference.

Input: Current video frame Ft; trained weight matrices W1, W2, W3 and displacement vectors b1, b2, b3; probability threshold Θ = 0.7; minimum delay time window t_{delay} = 2.0 s.

Output: Updated keyboard output text buffer B.

1. Capture frame F_t from the webcam using cv2.VideoCapture methods.
2. Convert the color space of the frame F_t from the BGR configuration to RGB.
3. Pass the array F_t to the detection engine mp.solutions.hands. Hands.
4. Extract the spatial coordinates of the 21 key points P = {P_0, P_1, ..., P_20}, where P_i = (x_i, y_i, z_i).
5. If the results.multi_hand_landmarks object is empty (no hand is detected), then:
6. Go to the next iteration frame F_t+1.
7. Read the coordinates of the wrist base point P_0(x_0, y_0, z_0).
8. For each anatomical node i from 0 to 20, perform:
9. Calculate the relative shift along the axes: x_i' = x_i - x_0; y_i' = y_i - y_0; z_i' = z_i - z_0.
10. Create a temporary intermediate array of shifted coordinates P'.
11. Find the scalar normalization coefficient: S = max_j (max(|x_j'|, |y_j'|, |z_j'|)).
12. Calculate the final standardized coordinates: P_i^norm = P_i' / S.
13. Perform the operation of flattening the array P^norm into a flat row vector X ∈ R^{1 × 63}.
14. Calculate the activation of the first hidden layer: A_1 = max(0, X · W_1 + b_1).
15. Calculate the activation of the second hidden layer: A_2 = max(0, A_1 · W_2 + b_2).
16. Generate the output probability vector: Y_hat = Softmax(A_2 · W_3 + b_3).
17. Determine the maximum probability c* = max(Y_hat) and its index k* = argmax(Y_hat).
18. If c* >= Θ, then:
19. Calculate the duration of the gesture hold Δt_hold = t_current - t_last_change.
20. If k* != S_last AND Δt_hold >= t_delay, then:
21. Update the text string: B = B + MapIndexToLetter(k*).
22. Update the system state: S_last = k*; t_last_change = t_current.

23. Return the current state of the text buffer B.

A natural property of any feedforward network is a constant asymptotic complexity of O(1) in time. The absence of recursive calculations in the inference phase and direct NumPy matrix operations allow the use of hardware vectorization (SIMD), which explains the high speed of algebraic calculations on Edge devices compared to classifiers that use memory.

4. Results and Discussion

In order to prevent data leakage and guarantee the methodological purity of the experiment, the collected array of 16513 samples was subjected to a random partitioning procedure using class stratification. The data were divided into three isolated subsets: training (70%, 11561 samples), validation (15%, 2476 samples) and test (15%, 2476 samples). The validation set was used exclusively to monitor cross-entropy during epochs and the operation of the Early Stopping mechanism. The final statistical metrics shown in Table 1 are calculated strictly on an independent test set (Test Set), which was not analyzed by the network optimizer.

Table 1
Evaluation of the effectiveness of ANNs on the test set (2476 samples)

Evaluation Metric	Value (%)
Accuracy (Global Accuracy)	98,18
Macro-Precision (Classifier Accuracy)	98,22
Macro-Recall (Completeness)	98,20
Macro-F1 Score (Harmonious Measure)	98,19

The achieved global accuracy rate (98.18%) confirms the stability of the developed mathematical apparatus of 3D-normalization. Comparative analysis with the domestic benchmark of A. Hlybovets (98.4% for 33 letters) [14] reveals significant architectural advantages of our model. Analog [14] is based on a massive LSTM architecture that analyzes sequences of 65 frames, generating computational lag during processing. Our MLP forward propagation model demonstrates comparable accuracy (only 0.22% less), but significantly reduces computational complexity, making inference delays impossible.

At the international level, the solution is competitive compared to the work of Z. Aleho (98.75% for FSL) [4]. However, unlike [4], which relies on transfer learning and powerful GPUs, the proposed architecture is optimized for edge computing (Edge AI) [9, 10]. The analytical estimate of the forward pass is expressed in terms of the number of fused multiply-add operations:

$$Ops=(63\times128)+(128\times64)+(64\times33)=18368 \text{ operations.}$$

The model has a total of 18593 parameters. Considering the base float32 format, the amount of RAM required is:

$$Memory = 18593 \times 4 \text{ bytes} \approx 74.4 \text{ KB.}$$

This amount can easily be accommodated in the ultra-fast cache (L1/L2 Cache) of the processor. Moreover, the promising integer quantization of weights (INT8), which is a standard for Edge deployments, will allow compressing the model size to ~18.6 KB, which will further accelerate inference on x86/ARM architectures. The rejection of TensorFlow/PyTorch graph engines and their overheads guaranteed uninterrupted operation. Physical testing on a hardware bench (HP Victus laptop, AMD Ryzen 5 8645HS processor, 16 GB RAM) confirmed that the average inference time of one frame solely based on the central processor without involving a discrete video card was ~0.1 ms. CPU utilization (CPU Usage) during continuous recognition of a 30 FPS stream remained minimal, which proves the energy efficiency of ANNs and the absence of a “bottleneck” effect.

It is worth noting the stable methodological connection between the theoretical issues of UDA and the obtained results. The obtained error matrix (Fig. 1) experimentally confirms the hypothesis regarding the two-component nature of the geometric complexity of the UDA. The high accuracy of

the classification of letters assigned to class A (only 2 errors in the pair "Г"—"Г" and the complete absence of false positives in the pair "Ш"—"Щ") proves the high efficiency of the algorithm of algebraic 3D-normalization of coordinates. Mathematical centering and isotropic scaling allowed to standardize the space and make local features invariant to translations.

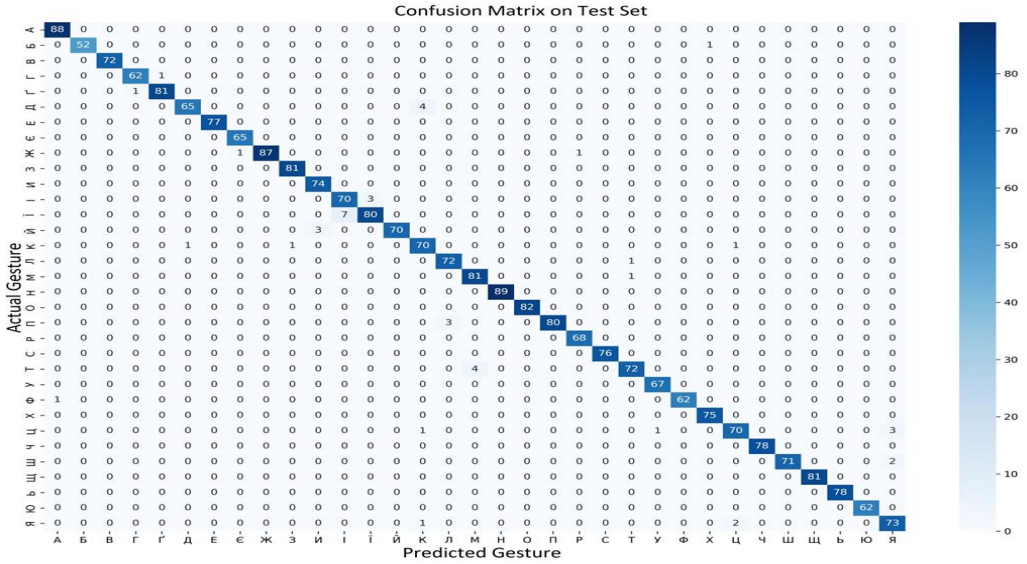

Figure 1: Confusion Matrix of low-parameter ANN on an isolated test sample (2476 samples).

Instead, the localization of the residual error (1.82%) within class B — mainly between the letters "И" and "Ю" (misclassified 10 times in total), "Т" — "М" (5 errors) and "Д" — "К" (5 errors) — clearly outlines the physical limits of the application of monocular computer vision.

Since under the conditions of the frontal shooting angle, the projection of 3D coordinates onto the 2D sensor matrix occurs, there is a partial loss of information about the depth coordinate (Z axis) and inevitable spatial occlusion of key MediaPipe points. The feature vectors of such letters become mathematically collinear when the fingers are overlapped by the palm (their cosine distance approaches unity). Thus, the developed low-parametric model has completely exhausted the mathematical potential of separating static spatial features at the level of individual frames, which logically justifies the feasibility of the future transition to dynamic recurrent (LSTM/GRU) models for capturing hand micromovements [14].

Testing the effectiveness of the gesture recognition system is carried out in real time and is the culminating stage, demonstrating its functionality. After successfully training the model or loading it from a previously saved file, the user initiates the launch of the camera. The system begins to continuously analyze the video stream, providing instant feedback (Fig. 2).

The effectiveness of the system was assessed by visually observing the speed and accuracy of recognizing various gestures performed by the user in dynamics. Thanks to careful data normalization, the system demonstrates relative stability to minor changes in the position of the hand, its orientation and size in the frame. The system's response time allows for real-time recognition, which is critically important for interactive applications.

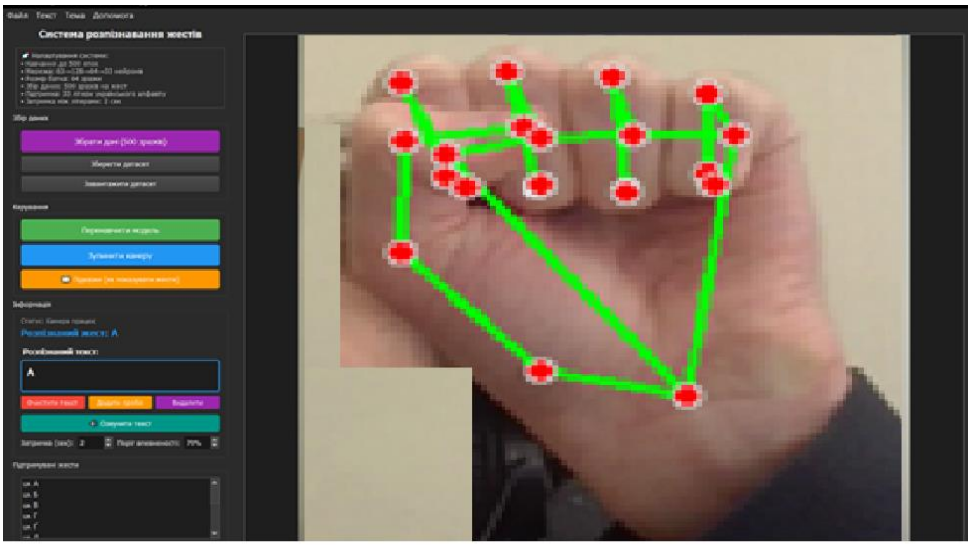

Figure 2: Real-time letter "A" recognition test.

5. Conclusions

A resource-efficient spatial data processing pipeline for non-contact identification of 33 letters of the Ukrainian dactyl alphabet in real time based on RGB cameras has been designed and implemented.

A two-stage algebraic model of 3D coordinate normalization has been formalized, which experimentally proved its effectiveness in solving the problem of microstructural coordinate differentiation (Class A), completely eliminating the mixing of complex Cyrillic letters with minimal joint displacements.

Implementation of direct propagation ANN based on NumPy linear algebra allowed minimizing RAM consumption to ~74.4 KB and achieving an inference speed of ~0.1 ms on the central processor. Integration of integer quantization INT8 will expand the system's potential for guaranteed hardware independence from GPU graphics accelerators.

On a fully isolated test sample (2476 samples), a global classification accuracy of 98.18% was confirmed. The proposed approach outperforms existing recurrent analogues in terms of speed and computational ease while maintaining high quality of static dactyl recognition.

Empirical testing of the system in dynamic mode confirmed its high functional suitability. Thanks to the applied algebraic 3D-normalization of features, the classifier is resistant to spatial fluctuations of the brush (changes in position, orientation and scale in the frame), and the minimum reaction time fully satisfies the requirements for interactive real-time applications.

Acknowledgements

We would like to express our sincere gratitude to all members of the research group whose dedicated and diligent work made a significant contribution to this study. We hope that the presented results will serve as a foundation for further research and contribute to strengthening the cyber resilience of critical systems.

Declaration of Generative AI Use

During the preparation of this work, the author(s) used ChatGPT-4 and Gemini in order to:
Grammar and spelling check, Improve writing style. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] S. Kulbida, O. Bilanova, L. Ivashchenko, Ukrainian Sign Language - Implementation of a New State Standard in Grades 5-6 of the New Ukrainian School, Special Child: Education and Upbringing 109 (2023) 33–60. doi:10.33189/ectu.v109i1.127.

[2] M. Rodriguez, O. Oubram, A. Bassam, N. Lakouari, R. Tariq, Mexican Sign Language Recognition: Dataset Creation and Performance Evaluation Using MediaPipe and Machine Learning Techniques, Electronics 14 (2025) 1423. doi:10.3390/electronics14071423.

[3] A. Cardano, F. R. Columna, J. Villaverde, Development of Transactional Filipino Sign Language Recognition System Using MediaPipe and Gated Recurrent Units, Engineering Proceedings 134 (2026) 47. doi:10.3390/engproc2026134047.

[4] Z. A. Alejo, N. C. J. R. Fuentes, M. P. Z. Lungay, et al., Innovative Filipino Sign Language Translation and Interpretation with MediaPipe, Engineering Proceedings 134 (2026) 75. doi:10.3390/engproc2026134075.

[5] A. I. Nadaf, S. Pardeshi, R. Gupta, Efficient gesture recognition in Indian sign language using SENet fusion of multimodal data, Scientific Reports 13 (2025) 114–122. doi:10.62110/sciencein.jist.2025.v13.1145.

[6] M. Gil-Martín, M. R. Marini, I. Martín-Fernández, S. Esteban-Romero, L. Cinque, Hand Gesture Recognition Using MediaPipe Landmarks and Deep Learning Networks, in: Proceedings of the 17th International Conference on Agents and Artificial Intelligence, SciTePress, 2025, pp. 24–30. doi:10.5220/0013053500003890.

[7] J. Heaton, Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning: The MIT Press, 2016, 800 pp, ISBN: 0262035618, Genetic Programming and Evolvable Machines 19 (2017). doi:10.1007/s10710-017-9314-z.

[8] C. R. Harris, K. J. Millman, S. J. van der Walt, et al., Array programming with NumPy, Nature 585 (2020) 357–362. doi:10.1038/s41586-020-2649-2.

[9] J. Fernandez, J. Kahn, C. Na, Y. Bisk, E. Strubell, The Framework Tax: Disparities Between Inference Efficiency in NLP Research and Deployment, in: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, 2023, pp. 1588–1600.

[10] H. Jin, Build an ML framework from scratch, 2024. URL: https://haifengjin.com/build-an-ml-framework-from-scratch/.

[11] S. Radcliffe, SmallPebble: A minimalistic autodiff library in Python, 2026. URL: https://github.com/sradc/SmallPebble.

[12] A. Bauer, R. Poryadin, Our stories – Experiences of Ukrainian Deaf people leaving Ukraine in 2022 and coming to Germany. Conversations in Ukrainian Sign Language, Data Center for the Humanities, Universität zu Köln, 2023. doi:10.18716/dch/a.00000015.

[13] R. Rastgoo, K. Kiani, S. Escalera, Sign Language Recognition: A Deep Survey, Expert Systems with Applications 164 (2021) 113794. doi:10.1016/j.eswa.2020.113794.

[14] A. Hlybovets, M. Bikchentayev, Recognizing gestures of the Ukrainian dactylic alphabet, Problems of Control and Informatics 68 (2023) 86–100. doi:10.34229/1028-0979-2023-3-9.