INVESTIGATION AND DEVELOPMENT OF METHODS AND ALGORITHMS FOR ANALYZING VIDEO-EEG MONITORING OF DELAYED CEREBRAL ISCHEMIA AFTER SUBARACHNOID HAEMORRHAGE

A. A. Morozov[1], O. S. Sushkova[1,*], M. V. Sinkin[2], I. V. Okuneva[2], Yu. V. Obukhov[1]

[1] Kotel'nikov Institute of Radio Engineering and Electronics of RAS, Mokhovaya 11-7, 125009 Moscow, Russia - morozov@cplire.ru (A.A.M.), o.sushkova@mail.ru (O.S.S.), yuvobukhov@mail.ru (Yu.V.O.)
[2] Department of Neurosurgery of the Sklifosovsky Research Institute for Emergency Medicine of Moscow Healthcare Department, Bolshaya Sukharevskaya Square 3, 129090 Moscow, Russia - mvsinkin@gmail.com (M.V.S.), okunevaiv@mail.ru (I.V.O.)

Commission II, WG II/8

KEY WORDS: Video-EEG, Neural-Symbolic Computing, Object-Oriented Logic Programming, Actor Prolog, Event Recognition, Facial Image Analysis, Posture Analysis, Intensive Care Unit.

ABSTRACT:

This work aims to study and develop methods and tools for analyzing video-EEG monitoring of delayed cerebral ischemia after subarachnoid haemorrhage. A study was made of methods and algorithms for processing video images, which are supposed to be used to recognize life events and medical care for a patient, which can lead to artefacts in the patient's EEG records. An approach to video monitoring of the patient was proposed and tested on real video monitoring data using object-oriented logic programming methods in combination with neural network methods for image analysis.

1. INTRODUCTION

Delayed cerebral ischemia is one of the most dangerous complications that can occur in a patient with subarachnoid haemorrhage (Scherschinski et al., 2022, Baang et al., 2021). This study was carried out as a part of a project aimed at automatically detecting early signs of delayed cerebral ischemia in patients undergoing treatment in the intensive care unit. In the framework of the project, continuous (over several days) collection and analysis of electroencephalographic (EEG) data of patients is carried out; video recording of patients is carried out simultaneously with the EEG recording. The main object of the analysis is EEG data. However, in some cases, the correct interpretation of the EEG data is impossible without taking into account the patient's video image (Tatum et al., 2022). The problem is that some movements of the patient, such as chewing and head trembling, can lead to artefacts in the patient's EEG records that are similar to the epileptic activity, which is one of the indirect signs of delayed cerebral ischemia. Thus, the doctor has to inspect the patient's video records when analyzing the EEG signals to prevent mistakes and this is a very time-consuming job. Methods for automatic EEG analysis should also be supplemented by methods for automatic analysis of the patient's video image.

The purpose of this study is to verify some design solutions that form the basis of the developed system for the logical analysis of patient video in the intensive care unit.

In the framework of the study, the following problems were investigated:

1. What quality of object recognition can be achieved using neural network methods for video data analysis? Is this quality sufficient for the analysis of video-EEG data of the patient undergoing treatment in the intensive care unit?
2. Do modern neural network methods provide data analysis performance sufficient for real-time video-EEG data analysis?
3. What approaches to the integration of the logical and neural methods for the data analysis should be applied within the framework of the problem being solved?

In the first section, the architecture of the designed system for the logical analysis of video images is considered. The second section discusses neural network methods for image analysis, which are supposed to be used as a part of the system. In the third section, the proposed architecture of the neuro-logical system is tested on examples of clinical data analysis.

2. THE ARCHITECTURE OF THE SYSTEM FOR THE LOGICAL ANALYSIS OF VIDEO IMAGES

The developed system for the logical analysis of video images is an example of so-called neural-symbolic systems (Garcez et al., 2019). In a broad context, we consider this study as an experimental testing of a new approach to the convergence of symbolic and connectionist methods of artificial intelligent (Toosi et al., 2021). Following the classification (Yu et al., 2022) of neural-symbolic systems, the developed system could be considered as a kind of learning for reasoning systems, that is, the neural networks are to be used as auxiliary elements in the system that supply information for logical reasoning. It is supposed that the logical system for video analysis will include two main levels of video data analysis:

1. The upper level of the analysis is based on logical rules. This level of the system is supposed to be implemented in the Actor Prolog object-oriented logic language (Morozov, 1999, Morozov and Sushkova, 2017, Morozov et al., 2019).

* Corresponding author

2. The lower level of the analysis is based on neural networks. This level of the system is supposed to be implemented by the means of the Open Source Computer Vision Library (OpenCV, Version 4.6.0, n.d.).

The distinctive feature of the developed architecture for neural-symbolic computing is the application of the object-oriented (OO) logic language. In the framework of logic programming (LP), the application of the ideas of object-oriented programming (OOP) is not only engineering but also a mathematical problem because one has to elaborate logical semantics for all OO constructs of the programming language. The Actor Prolog language instantiates so-called clauses view in LP-based OOP languages (Davison, 1993). In the framework of this view, images and videos are to be considered as instances of some built-in classes of the OO logic language. This enables one to manipulate complex/large data structures as atomic symbols and opens a way to create high-performance software for logical video analysis that is applicable to solving real-life problems. As far as we know, our system is the first neural-symbolic system based on object-oriented logic programming.

The second important feature of the developed architecture is that the Actor Prolog language is a translator to Java (Morozov et al., 2015) that enables one to obtain a high-performance code and simplifies the connection of the software system with the OpenCV library. Note that only a few neural-symbolic systems in the world are based on professional programming languages/libraries like Python (Mojarad et al., 2020, Tarau, 2021, Li et al., 2022, Winters et al., 2022, Ciatto et al., 2022), Kotlin (Ciatto et al., 2021), PyTorch (Manhaeve et al., 2018, Yang et al., 2020), TensorFlow (Li et al., 2022, Ciatto et al., 2022), and Keras (Mojarad et al., 2020). Most neural-symbolic systems are applied only for solving toy problems and only a few neural-symbolic systems applied modern neural architectures like YOLO (Khan et al., 2019) and VGG16 (Padalkar et al., 2023). In contrast to other neural-symbolic systems, the application of the Actor Prolog language let us conduct experiments with any neural architectures and neural network formats supported by the OpenCV library. The OpenCV library provides an interface between the Actor Prolog language and the CUDA architecture. All examples of video images considered in this paper are created using this software interface.

The modern implementation of the Actor Prolog language (Morozov and Sushkova, 2018a, Morozov et al., 2017) includes the *VideoProcessingMachine* built-in class that implements image processing methods including the background subtraction, extraction/tracing blobs, etc. These low-level methods of image processing have no logical semantics; thus, they were implemented in Actor Prolog in a form of a virtual machine for low-level video processing (Morozov and Sushkova, 2018b). This virtual machine is considered as an external process that analyzes the images and sends the results (the co-ordinates of blobs, trajectories of objects, etc.) to the logic program. We have extended the architecture of the virtual machine (Morozov and Sushkova, 2018b) by neural network methods for the image processing and analysis:

1. Neural network methods for blob extraction. Note that the methods for recognition of faces, hands, and skeletons listed below also extract the blobs in the image.
2. Neural network methods for blob classification.
3. Neural network methods for detecting facial landmarks inside given blobs. These methods are applicable to the blobs extracted by the face recognition methods.
4. Neural network methods for detection of faces, hands, and skeletons. In contrast to other blob extraction methods, these methods detect landmarks and skeleton nodes in the images.

The extended architecture for low-level video processing is described in Table 1. New commands of the virtual machine are indicated in bold.

All neural network methods were adopted for the architecture of the virtual machine; for instance, the virtual machine can apply existing methods for tracing blobs that were extracted using a neural network, select a blob with the longest track, and send the trajectory of the corresponding face to the logic program that implements further analysis of the data.

3. SELECTION OF NEURAL NETWORK ALGORITHMS FOR LOW-LEVEL VIDEO ANALYSIS

Two use-cases of applications of the neural network video analysis methods and algorithms were studied:

1. Analysis of the image of the face of the patient undergoing treatment in the intensive care unit. It was taken into account that EEG electrodes are attached to the patient's head and the face can be partially covered by the mask of the artificial respiration device. Patient movements (such as chewing and head trembling) have to be recognized because they can lead to the appearance of artefacts in the EEG records.
2. Image analysis of the video scene "the patient lies in a bed in the intensive care unit". The approach of doctors and nurses on duty to the patient's bed has to be recognized (for instance, they can correct the electrodes on the patient's head) because it can lead to the appearance of artefacts in the EEG records.

The methods were tested on a computer with the Intel i9-12900HX processor (2.30 GHz) and the NVIDIA GeForce RTX 3080 Ti Laptop graphics card. Compute Unified Device Architecture (CUDA) 11.8 and CUDA Deep Neural Network (cuDNN) 8.6.0 drivers were installed on the computer. A set of object-oriented logic programs was developed for the investigation and testing of video image analysis algorithms. Testing was carried out on the data of video monitoring of patients of the Department of Neurosurgery of the Sklifosovsky Research Institute for Emergency Medicine of Moscow Healthcare Department.

The following methods for face detection have been investigated:

1. Neural network face detection and recognition of five facial landmarks using the YuNet architecture (Feng et al., 2021). See an example in Figure 1.
2. Neural network face detection using the Single-Shot-Multibox detector architecture based on ResNet-10 (Liu et al., 2016).
3. Detection of 68 facial landmarks using the Practical Facial Landmark Detector (PFLD) (Guo et al., 2019).

Note that face detection methods based on local binary patterns (Ahonen et al., 2004) and cascade classifiers based on

Sub-stage of the processing	Commands of the virtual machine	Internal arrays of the machine
Pre-processing of the frame	Extract a sub-image; Crop image; Resize the image; Gaussian filtering; etc.	An image (the *BufferedImage* class of the Java language)
Processing of the frame in the pixel representation	Select a channel in HSB/RGB space or the greyscale channel; Smooth the image; Compute gradients; Normalize pixels; etc.	A matrix of pixels (the *int[]* data type of the Java language)
Selection and processing of the foreground pixels in the frame	Push a new mask of foreground pixels into the stack; Pop the top mask from the stack; Add mask pixels corresponding to given conditions; Eliminate mask pixels corresponding to given conditions; Subtract background using the given algorithm; Implement rank filtering of the foreground mask; Erode the mask of the foreground pixels; Dilate the mask; etc.	A mask of the foreground pixels (the *boolean[]* data type of the Java language)
Extraction of the blobs in the frames	Extract blobs using the given algorithm; **Extract blobs using given neural network**; Fill in the blobs; Select blobs corresponding to given conditions; Compute colour histograms of the blobs; **Classify blobs using given neural network**; **Detect facial landmarks in blobs using given neural network**; etc.	Co-ordinates and other attributes of the blobs
Detection of faces, hands, and skeletons	**Detect faces using given neural network**; **Detect hands using given neural network**; **Detect skeletons using given neural network**;	Lists of co-ordinates and other attributes of the landmarks and skeleton nodes
Tracing the blobs in the frames	Trace the movements of the blobs;	Lists of co-ordinates and other attributes of the blobs
Additional processing of the tracks of the blobs	Select tracks corresponding to given conditions; etc.	A list of tracks

Table 1. The extended architecture of the low-level video processing virtual machine.

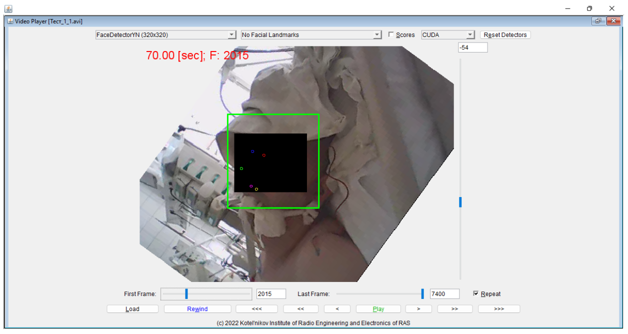

Figure 1. A neural network detection of faces and recognition of five facial landmarks using the YuNet architecture. The image is rotated by 54 degrees to help the neural network to detect the face. A black rectangle is used to anonymize the image.

Haar features (Viola and Jones, 2004) do not provide recognition quality acceptable for video analysis of patients undergoing treatment in the intensive care unit; there are a large number of misses and false positives. The method for detecting 68 facial landmarks using the regression of local binary features (Ren et al., 2014) also turned out to be inoperable under video recording conditions. Neural network methods operate much more reliably; however, we have identified some features of these methods that complicate their application. In particular, none of the neural network methods can provide face detection at an arbitrary angle of rotation of the video image. To ensure reliable detection of a person's face in a lying position, it is first necessary to determine the angle of rotation of the image and make appropriate corrections to the transmitted video image. The performance of neural network methods is satisfactory. In particular, the FaceDetectorYN 640x640 architecture provides a video analysis performance of about 25.3 frames per second (fps) even without the use of a GPU. When using the GPU, the FaceDetectorYN 640x640 architecture provides a performance of 28.6 fps. FaceDetectorYN 320x320 architecture provides 27.1 fps performance when using the GPU. The Single-Shot-Multibox 8bit Quantized TensorFlow architecture delivers 26.6 fps when using the GPU. The Single-Shot-Multibox Caffe FP16 architecture, when using the GPU, showed the worst performance of 24.4 fps. All studied neural network architectures produce unstable face co-ordinates, which can cause difficulties in recognizing EEG artefacts associated with chewing and head movements.

The following methods for pose estimation have been investigated (Cao et al., 2017):

1. Neural network estimation of people's poses using the affinity fields of body parts and the Common Objects in Context (COCO) body model 18 (see an example in Figure 2).
2. Neural network estimation of human poses using body parts affinity fields and Max-Planck-Institut für Informatik (MPI) 15 body model.
3. Neural network estimation of people's poses using the affinity fields of body parts and the MPI 15 body model (a simplified architecture).

The study of these methods showed that all the considered neural network methods cope with the recognition of life events and medical care events, which can lead to the appearance of artefacts in the patient's EEG records. The quality of pose recognition is satisfactory. The use of a graphics processor is required for the operation of all considered neural network methods. The architecture based on the COCO 18 human body model showed the best performance of about 27.1 fps. The MPI 15 model (the simplified architecture) provided a performance of about 27.5 fps, but the recognition quality is worse. The MPI 15 model (non-simplified) showed a performance of about 22.6 fps. The COCO 18 model showed the best quality of human pose recognition.

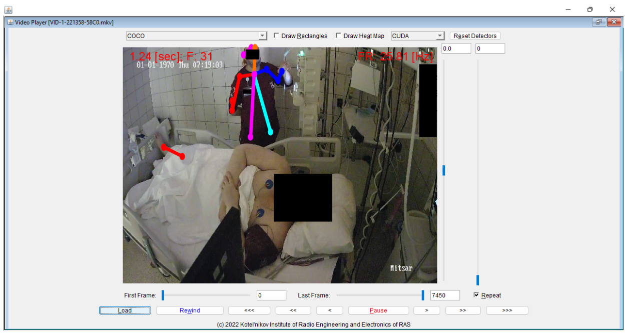

Figure 2. An example of recognizing the posture of a nurse on duty in the intensive care unit using the affinity fields of body parts and COCO 18 body model.

Notably, the neural network method for people pose estimation based on the affinity fields of body parts can estimate the pose of the upper part of the body of the lying patient that is not covered by the blanket (see an example in Figure 3). This can be explained by the fact that indicated neural network method is based on the detection of separate parts of the body and a step-by-step refinement of the results (Wei et al., 2016). Thus, the method can detect the parts of the body that are not covered by the blanket although the neural network was not trained on the images of people lying under the blanket. Note that the detection is possible only if the image of the patient was rotated by about 90 degrees.

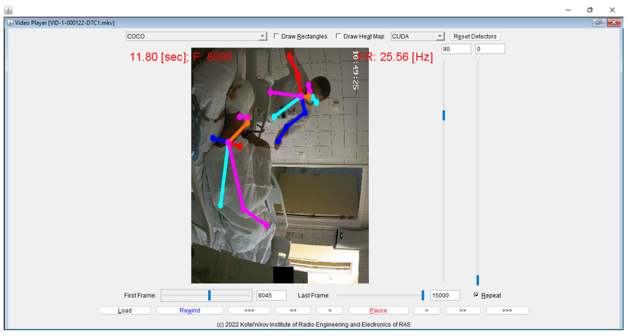

Figure 3. An example of recognizing the posture of a patient lying under a blanket. The image is rotated by 90 degrees to help the neural network to detect the patient.

4. EXPERIMENTS WITH A PROTOTYPE OF THE SYSTEM FOR LOGICAL ANALYSIS OF VIDEO

Let us consider two examples of practical tasks linked with the estimation of the patient's condition in the intensive care unit.

1. Detection and estimation of the patient's head trembling. The trembling can occur during an epileptic seizure.
2. Detection of the chewing movements of the patient. Such movements can be observed during the epileptic seizure but not only when the patient is eating something.

Two different methods for trembling detection based on the video analysis were tested:

1. Analysis of the trajectory of the facial landmark on the nose bridge of the patient (see an example in Figure 4).

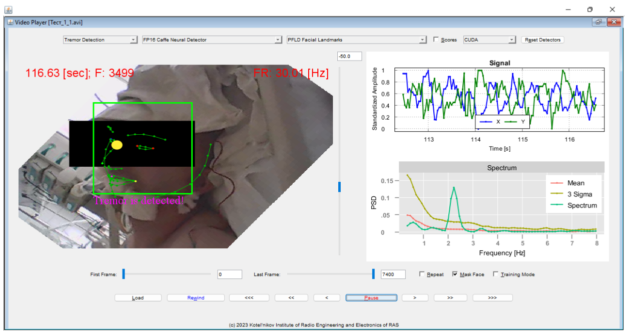

Figure 4. An example of recognizing the tremor of the patient's head. The small yellow circle indicates the facial landmark in the nose bridge. The logic program detects and analyses co-ordinates of this point.

2. Analysis of the trajectory of the central point of the blob corresponding to the patient's face.

The detection of chewing also was implemented using two different methods:

1. Analysis of the distance between the facial landmarks on the nose bridge and chin of the patient (see an example in Figure 5).
2. Analysis of the height of the blob corresponding to the patient's face.

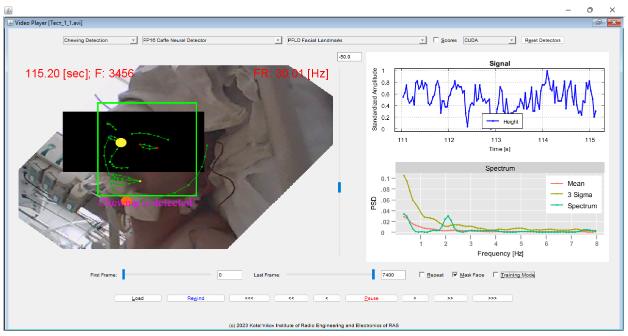

Figure 5. An example of recognizing the chewing in the patient's video. The small yellow circle indicates the facial landmark in the nose bridge. The small orange circle indicates the facial landmark in the chin. The logic program detects and analyses the vertical distance between these two points.

These experiments addressed an estimation of the signal-to-noise ratio of investigated co-ordinates and distances. These estimations are necessary to compare the neural network methods and select the most precise method. The following two face detection neural architectures were checked: the YuNet architecture (Feng et al., 2021) and the Single-Shot-Multibox architecture based on ResNet-10 (Liu et al., 2016). The facial landmarks detection was implemented using the PFLD architecture (Guo et al., 2019).

The experimental logic program was implemented in the Actor Prolog language and translated to Java. The logic program implements the following stages of video analysis:

1. An instance of the *Video Processing Machine* built-in class is created. The program uploads image processing commands in this instance in accordance with the neural network methods chosen by the user in a dialog box.

2. The program opens and reads a video file chosen by the user. Each frame of the video file is transferred to the virtual machine for low-level image processing. The virtual machine implements given image processing commands automatically and returns the results to the logic program. The results include trajectories and co-ordinates of target objects in a symbolic form, that is, in a form of lists, structures, sets, atomic symbols, and numbers.
3. The logic program extracts signals to be analyzed from the collected data structures in a form of lists of co-ordinates and distances.
4. Spectral analysis of the signals is performed and spectra of the signals are computed. The spectrum is computed in the window of the 125-point width (that is, of about 5 seconds). The Welch method is used with the following attributes: FFT window width is 256 points; the Hanning window is applied; the window overlap is 7/8. Spectral analysis is implemented using the *DSP* built-in class of Actor Prolog.
5. The logic program computes the mean and standard deviation of spectra in each frequency point in the interval from 0.25 to 8 Hz.
6. The logic program compares the current spectrum of the signal with the mean and standard deviation of spectra computed before. If the power spectral density (PSD) of the signal is more than the mean PSD for at least 3 standard deviations in some frequency point, the program reports that the target event is detected.
7. The recent points of the target signal, the current spectrum, the mean spectrum, and 3σ confidence interval of the spectrum are demonstrated in the dialog box of the logic program using the *Chart* built-in class of Actor Prolog.

The event detection rule described above has the following form in the notation of the logic language:

```
check_spectrum(Spectrum,Mean,Sigma,Result):-
     UB== Mean + 3 * Sigma,
     check_threshold(Spectrum,UB,Result).
check_threshold([A|_],[B|_],'yes'):-
     A > B,!.
check_threshold([_|R1],[_|R2],Mode):-!,
     check_threshold(R1,R2,Mode).
check_threshold(_,_,'no').
```

We have investigated two videos of a patient in the intensive care unit. In the first video, the patient demonstrates no movements (the duration of the video is about 4 minutes); another video demonstrates a head tremor and chewing movements of the patient (the duration of the video is about 43 seconds). Each video was processed using the logic program; the means and standard deviations of target signal spectra were computed. The ratio between the mean PSD in the second and first videos was considered as an estimation of the signal-to-noise ratio of the signals.

Figure 6 demonstrated the results of the patient’s head trembling detection. The diagrams indicate that the Single-Shot-Multibox detector supplies a better signal-to-noise ratio than the YuNet architecture. Notably, the co-ordinates of the blob center supply a better signal-to-noise ratio than the nose bridge landmark, if the Single-Shot-Multibox architecture is used. However, this regularity is not observed for the YuNet architecture.

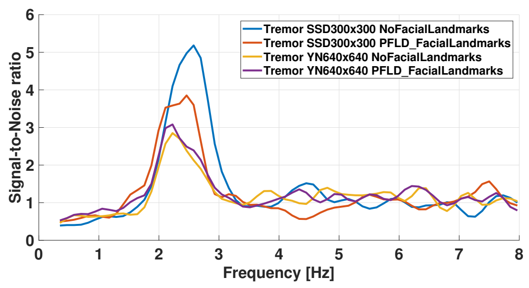

Figure 6. The signal-to-noise ratio that characterized the tremor detectors.

The experiment with chewing detection yields different results. Figure 7 demonstrates that the Single-Shot-Multibox architecture supplies a better signal-to-noise ratio than the YuNet architecture if the height of the face blob is analysed. However, if the distance between the facial landmarks on the nose bridge and chin is analysed, both neural network methods yield bad results.

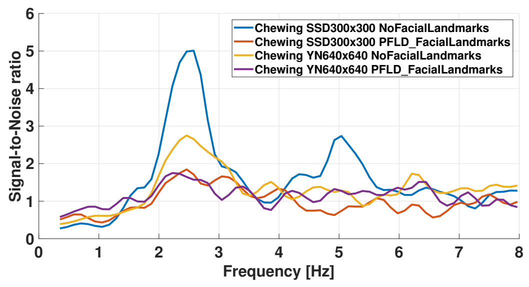

Figure 7. The signal-to-noise ratio that characterized the chewing detectors.

The results of the experiments indicate that the logical analysis of the patient’s video can be used for detecting the patient’s head tremor and chewing movements. Both Single-Shot-Multibox and YuNet neural network architectures can be applied; however, the Single-Shot-Multibox architecture is preferable because it supplies a better signal-to-noise ratio for the co-ordinates and distances in the patient’s video.

5. CONCLUSIONS

Experiments were carried out on the implementation of the selected neural network methods for image analysis in an object-oriented logic programming system and the implementation of a logical inference based on the results of the image analysis. An extended architecture of the Actor Prolog virtual machine for low-level video processing is proposed. The conducted experiments demonstrated the possibility of analyzing videos of patients undergoing treatment in the intensive care unit. The required image recognition quality and processing performance can be achieved by translating Actor Prolog programs to Java with CUDA support. The programming interface between the Actor Prolog language and the CUDA architecture can be provided using the OpenCV library.

An approach to video monitoring of the patient’s head movements, as well as actions of medical personnel, which can cause the appearance of artefacts in the patient’s EEG records, was proposed and tested on real video monitoring data using object-oriented logic programming methods in combination with neural network methods for image analysis.

The research was supported by the Russian Science Foundation, grant no. 22-69-00102, https://rscf.ru/en/project/22-69-00102/.

REFERENCES

Ahonen, T., Hadid, A., Pietikäinen, M., 2004. Face recognition with local binary patterns. *European conference on computer vision*, Springer, 469–481.

Baang, H. Y., Chen, H. Y., Herman, A. L., Gilmore, E. J., Hirsch, L. J., Sheth, K. N., Petersen, N. H., Zafar, S. F., Rosenthal, E. S., Westover, M. B., Kim, J. A., 2021. The Utility of Quantitative EEG in Detecting Delayed Cerebral Ischemia After Aneurysmal Subarachnoid Hemorrhage. *Journal of Clinical Neurophysiology*.

Cao, Z., Simon, T., Wei, S.-E., Sheikh, Y., 2017. Realtime multi-person 2D pose estimation using part affinity fields. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7291–7299.

Ciatto, G., Calegari, R., Omicini, A., 2021. 2P-Kt: A logic-based ecosystem for symbolic AI. *SoftwareX*, 16, 100817.

Ciatto, G., Castigliò, M., Calegari, R., 2022. Logic Programming library for Machine Learning: API design and prototype. *CEUR-WS.org/Vol-3204/paper_12.pdf.*

Davison, A., 1993. A survey of logic programming-based object-oriented languages. P. Wegner, A. Yonezawa, G. Agha (eds), *Research Directions in Concurrent Object Oriented Programming*, MIT Press, 42–106.

Feng, Y., Yu, S., Peng, H., Li, Y.-R., Zhang, J., 2021. Detect Faces Efficiently: A Survey and Evaluations. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 4(1), 1–18.

Garcez, A. d., Gori, M., Lamb, L. C., Serafini, L., Spranger, M., Tran, S. N., 2019. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *arXiv:1905.06088v1 [cs.AI] 15 May 1019.*

Guo, X., Li, S., Yu, J., Zhang, J., Ma, J., Ma, L., Liu, W., Ling, H., 2019. PFLD: A Practical Facial Landmark Detector. *ArXiv:1902.10859v2 [cs.CV] 3 March 2019*.

Khan, A., Bozzato, L., Serafini, L., Lazzerini, B., 2019. Visual Reasoning on Complex Events in Soccer Videos Using Answer Set Programming. *EPiC Series in Computing*, 65, 42–53.

Li, Y., Wei, H., Han, Z., Jiang, N., Wang, W., Huang, J., 2022. Computer Vision-Based Hazard Identification of Construction Site Using Visual Relationship Detection and Ontology. *Buildings*, 12, 857.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A. C., 2016. SSD: Single shot multibox detector. *European conference on computer vision*, Springer, 21–37.

Manhaeve, R., Dumančić, S., Kimmig, A., Demeester, T., Raedt, L. D., 2018. DeepProbLog: Neural probabilistic logic programming. *arXiv:1805.10872v2 [cs.AI] 12 December 2018.*

Mojarad, R., Attal, F., Chibani, A., Amirat, Y., 2020. A context-aware hybrid framework for human behavior analysis. *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, IEEE, 460–465.

Morozov, A. A., 1999. Actor Prolog: an object-oriented language with the classical declarative semantics. K. Sagonas, P. Tarau (eds), *IDL 1999*, Paris, France, 39–53.

Morozov, A. A., Sushkova, O. S., 2017. Real-time analysis of video by means of the Actor Prolog language. *Computer Optics*, 97–105.

Morozov, A. A., Sushkova, O. S., 2018a. Development of agent logic programming means for heterogeneous multichannel intelligent visual surveillance. G. R. Simari, E. Fermé, F. Gutiérrez Segura, J. A. Rodríguez Melquiades (eds), *Advances in Artificial Intelligence – IBERAMIA 2018*, Springer International Publishing, Cham, 29–41.

Morozov, A. A., Sushkova, O. S., 2018b. A Virtual Machine for Low-Level Video Processing in Actor Prolog. *Journal of Physics: Conference Series*, 1096(1), 012044.

Morozov, A. A., Sushkova, O. S., Polupanov, A. F., 2015. A translator of Actor Prolog to Java. N. Bassiliades, P. Fodor, A. Giurca, G. Gottlob, T. Kliegr, G. Nalepa, M. Palmirani, A. Paschke, M. Proctor, D. Roman, F. Sadri, N. Stojanovic (eds), *RuleML 2015 DC and Challenge*, CEUR, Berlin.

Morozov, A. A., Sushkova, O. S., Polupanov, A. F., 2017. Towards the distributed logic programming of intelligent visual surveillance applications. O. Pichardo-Lagunas, S. Miranda-Jimenez (eds), *Advances in Soft Computing, Part II*, Springer International Publishing, Cham, 42–53.

Morozov, A. A., Sushkova, O. S., Polupanov, A. F., 2019. Object-oriented logic programming of intelligent visual surveillance for human anomalous behavior detection. M. Rivas-Lopez, O. Sergiyenko, W. Flores-Fuentes, J. C. Rodriguez-Quinonez (eds), *Optoelectronics in Machine Vision-Based Theories and Applications*, IGI Global Publications, 134–187.

OpenCV, Version 4.6.0, n.d. https://github.com/opencv (13 October 2022).

Padalkar, P., Wang, H., Gupta, G., 2023. NeSyFOLD: A System for Generating Logic-based Explanations from Convolutional Neural Networks. *arXiv:2301.12667v1 [cs.LG] 30 January 2023.*

Ren, S., Cao, X., Wei, Y., Sun, J., 2014. Face alignment at 3000 fps via regressing local binary features. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1685–1692.

Scherschinski, L., Catapano, J. S., Karahalios, K., Koester, S. W., Benner, D., Winkler, E. A., Graffeo, C. S., Srinivasan, V. M., Jha, R. M., Jadhav, A. P., Ducruet, A. F., Albuquerque, F. C., Lawton, M. T., 2022. Electroencephalography for detection of vasospasm and delayed cerebral ischemia in aneurysmal subarachnoid hemorrhage: a retrospective analysis and systematic review. *Neurosurg Focus*, 52(3), E3.

Tarau, P., 2021. Natlog: A lightweight logic programming language with a neuro-symbolic touch. *International Conference on Logic Programming (Technical Communications) 2021 (ICLP 2021)*, 141–154.

Tatum, W. O., Mani, J., Jin, K., Halford, J. J., Gloss, D., Fahoum, F., Maillard, L., Mothersill, I., Beniczky, S., 2022. Minimum standards for inpatient long-term video-EEG monitoring: A clinical practice guideline of the international league against epilepsy and international federation of clinical neurophysiology. *Clinical Neurophysiology*, 134, 111–128.

Toosi, A., Bottino, A., Saboury, B., Siegel, E., Rahmim, A., 2021. A brief history of AI: how to prevent another winter (a critical review). *PET clinics*, 16(4), 449–469.

Viola, P., Jones, M. J., 2004. Robust real-time face detection. *International journal of computer vision*, 57(2), 137–154.

Wei, S.-E., Ramakrishna, V., Kanade, T., Sheikh, Y., 2016. Convolutional pose machines. *CVPR*.

Winters, T., Marra, G., Manhaeve, R., De Raedt, L., 2022. DeepStochLog: Neural stochastic logic programming. *Proceedings of the AAAI Conference on Artificial Intelligence*, 10090–10100.

Yang, Z., Ishay, A., Lee, J., 2020. NeurASP: Embracing neural networks into answer set programming. *29th International Joint Conference on Artificial Intelligence (IJCAI 2020).*

Yu, D., Yang, B., Liu, D., Wang, H., Pan, S., 2022. Recent Advances in Neural-symbolic Systems: A Survey. *arXiv:2111.08164v2 [cs.LG] 20 November 2022*.