Vinnytsia National Technical University
Ministry of Education and Science of Ukraine

Qualification Scientific Work
as a Manuscript

XIA GUANXIANG

UDC 004.93.1

DISSERTATION

MODELS AND METHODS FOR IMPROVING THE EFFICIENCY OF TEXT AREA RECOGNITION IN REAL-TIME VIDEO STREAMS

122 – Computer Science

Submitted for the award of the academic degree of Philosophy Doctor

The dissertation contains the results of the author's own research. The use of ideas, results, and texts from other authors includes references to the respective sources.
Xia Guanxiang Xia Guanxiang

Scientific Advisor: Viacheslav Kovtun, Doctor of Technical Sciences, Professor

Vinnytsia – 2025

ABSTRACT

Xia Guanxiang. Models and methods for improving the efficiency of text area recognition in real-time video streams. – Qualification scientific work as a manuscript.

Dissertation for the degree of Philosophy Doctor in the specialty 122 "Computer Science". – Vinnytsia National Technical University, Vinnytsia, 2025.

In today's conditions of digital transformation, both society and business require automated processing of vast amounts of information, including data from visual sources. One such task is the recognition of document contents in videos, a dynamic and complex field situated at the intersection of computer vision, classical pattern recognition theory, natural language processing (NLP), and deep learning technologies. This task is focused on extracting textual and structural information from documents presented in video streams or recordings. This task manifests itself in the following areas of human activity:

1. Digitization of archival data. Many historical or legal documents are stored in the form of video recordings or film materials. Automating their analysis enables the extraction of text and structural data, accelerating the processing. For example, scanned handwritten documents from archives may be presented in motion or from unconventional angles in video format.

2. Real-time document control systems. In airports and checkpoints, video cameras capture documents such as passports and visas for automatic authenticity verification. This requires text recognition and data comparison with databases under time constraints.

3. Education and conferences. Recognizing text on boards, presentations, or screens during lectures allows for automatic transcription creation, facilitating access to knowledge. This is especially relevant for online education, where video streams may be the only source of information.

4. Automation in manufacturing. In logistics and industry, video surveillance systems analyze text on packaging, labels, or instructions. For example, video cameras can automatically capture serial numbers on moving goods.

At the same time, the task of recognizing text in videos differs from traditional optical character recognition (OCR) due to additional challenges. Videos may contain dynamic camera movement, lighting changes, object occlusion, or noise. This requires more complex algorithms capable of handling factors such as:

1. Low video quality. Poor lighting, low resolution, or camera movement complicate text extraction. It should also be noted that many documents contain security features that make optical scanning of the content more difficult (watermarks, specific backgrounds, reflective (glare) fragments, etc.).

2. Multilingualism and font diversity. Texts in different languages need to be recognized, including writing systems with non-standard fonts or diacritical marks, requiring universal models.

3. Complex document structures. Often, it is necessary to recognize not only the text but also complex elements such as tables, graphs, and charts.

4. Real-time processing requirements. In security and automation systems, fast video processing is required without sacrificing quality.

Thus, document content recognition in videos is a task with enormous potential for application in business, security, science, and education. The development of this field requires the integration of computer vision, machine learning, NLP, and deep learning methods. Solving existing challenges, such as difficult shooting conditions and multilingualism, will ensure an even broader range of applications for this technology in the future.

The introduction justifies the relevance of the chosen research direction, analyzes various approaches to the organization of communicative software, their weaknesses, etc. The goal, objectives, object, subject, and methods of the research are defined, the scientific novelty and practical significance are presented, along

with information about the testing and implementation of the research results, as well as the structure and volume of the dissertation.

The first chapter is devoted to the analysis of the principles of constructing modern document recognition systems. It discusses automatic document input as one of the main tasks arising within electronic and mobile document workflows. The key components of such systems and their properties are described. It is shown that current works related to automatic document input and recognition on mobile devices treat document photographs as their electronic representation and highlight the challenges related to preparing the document image for recognition and the recognition process itself.

The second chapter proposes a new mathematical model for the optical object recognition system in a video stream, with a block for combining recognition results of the object from individual images and a block for stopping the recognition process. A problem formulation for recognition within such a system is proposed.

The third chapter is dedicated to the development of an algorithm for combining (integrating) recognition results of a string object in a video stream within the result model, taking into account alternative classification options for individual characters. The problem formulation, formal description of the algorithm, and results of a comparative experimental study of the proposed algorithm and the ROVER algorithm are presented. A data synthesis method for training and adjusting image recognition algorithms and a methodology for creating open identity document data sets are also discussed.

The fourth chapter introduces a new method for stopping the recognition process of an object in a video stream based on threshold truncation of the estimated expected distance between the current and next integrated results, and presents a new algorithm for stopping the recognition of string objects. The formal formulation of the recognition stop problem is considered, and a method obtained by treating the problem as a monotonic stop problem is proposed. The results of a comparative experimental study of the proposed algorithm and other stop rules

previously suggested for similar tasks are presented. The advantages of the proposed algorithm over other methods are demonstrated.

Scientific Novelty:

1. A dynamic mathematical model of a real-time object recognition system in video streams has been proposed for the first time. In contrast to static analogues, the model enables formalisation of the frame-to-frame information accumulation process and ensures consistency in result integration over temporal sequences. This advancement allows the system's behaviour to be described as an anytime process and provides a rationale for selecting the optimal stopping point based on computational load. Experimental results demonstrate that the application of this model increases the accuracy of symbolic object recognition by up to 9% compared to single-frame approaches without integration, confirming the effectiveness of the proposed solution..

2. A method for integrating the recognition results of symbolic objects in video streams has been developed for the first time, taking into account alternative classifications for each character. Unlike existing approaches that rely on a rigid selection of a single option at each step, the proposed method operates with a set of probabilistic hypotheses per character and enables iterative refinement of the final result under uncertainty. This made it possible to align data integration using a weighted correction mechanism based on Bayesian estimation and to adapt the scheme to low-quality frames. Experimental results demonstrated an improvement in recognition accuracy by 7% compared to the benchmark analogue (ROVER)..

3. A method for stopping the recognition process of symbolic objects in video streams has been developed for the first time, based on threshold reduction of the expected distance between the current and the subsequent integrated result, formally framed as a monotonic stopping problem. In contrast to classical stopping rules that rely on a fixed number of frames or heuristic change detection criteria, the proposed approach evaluates the expected gain from the next step and makes termination decisions based on the predicted improvement in the result. This enabled a significant reduction in the average number of processed frames (by 18%)

without compromising accuracy, which is particularly critical for real-time systems operating on mobile devices with limited computational resources.

Practical Significance of the Results:

1. The developed methods for integrating frame-by-frame recognition results significantly improve the accuracy of final analysis results of video sequences, which is crucial for tasks such as document digitization and real-time data processing.

2. The possibility of adapting the ROVER method, originally designed to combine results from different recognition algorithms, to the task of combining results from different observations of the same object is demonstrated. This expands the scope of application of this method.

3. The developed integration algorithm shows higher accuracy than traditional approaches in the task of recognizing text fields in passport videos and other identification documents. This is particularly important for automating border control and identity verification processes.

4. The proposed method for stopping the recognition process allows the optimal moment for completing the processing of a video sequence to be determined. This reduces the excessive use of computational resources and increases the efficiency of systems, especially on mobile devices with limited resources.

5. The developed methods improve the performance of optical character recognition systems operating in real-time, such as those used for automating the processing of video recordings from surveillance cameras.

6. Experimental validation on real data and the use of the open-source Tesseract library demonstrate the applicability of the developed methods for integration into mobile and cloud applications.

7. The conducted research shows how the properties of input data affect the choice of optimal combination strategy. This allows for system customization for specific conditions, such as the presence of noise or defects in preprocessing.

8. The formal formulation of the stopping task in video sequence recognition and the proposed algorithm are novel and can be used to optimize the performance of systems focused on video stream analysis.

9. Experimental results obtained on the MIDV-500 dataset confirm the effectiveness of the developed methods for text recognition in documents used in verification systems, which opens up prospects for implementation in governmental and commercial applications.

10. The developed methods and algorithms for integration and stopping can be adapted to a wide range of computer vision tasks beyond text recognition, such as dynamic scene analysis and other object-related tasks.

Keywords: Real-time video processing; text area recognition; deep learning; convolutional neural networks (CNN); anomaly detection; machine learning; pattern recognition; semantic segmentation; data augmentation; object detection; ERP integration; computer vision; intelligent systems; image watermarking; image encryption; digital security; OCR accuracy improvement.

LIST OF PUBLICATIONS OF THE APPLICANT

1. X. Guanxiang and V. Kovtun, “The model of the system for objects recognition in the real-time video stream”, Computer systems and information technologies, no. 4, pp. 157–162, Dec. 26, 2024. dooi: 10.31891/csit-2024-4-19.

2. X. Guanxiang and V. Kovtun, “Methods for implementation of string object recognition results in real-time video streams,” Measuring And Computing Devices In Technological Processes, no. 4. Khmelnytskyi National University, pp. 338–347, Nov. 28, 2024. doi: 10.31891/2219-9365-2024-80-41.

3. X. Guanxiang and V. Kovtun, “The method for stopping the string object recognition process in a real-time video stream,” Measuring And Computing Devices In Technological Processes, no. 3. Khmelnytskyi National University, pp. 243–250, Aug. 29, 2024. doi: 10.31891/2219-9365-2024-79-32.

4. X. Guanxiang, “A Study of the Implementation Techniques and Processes of Live Video Courses in Online Teaching and Learning,” Journal of Online Learning, no. 2, pp. 3224–3225, Jun. 29, 2022. doi: 10.12255/j.issn.1672-6677.2021.11.1603.

5. X. Guanxiang, “Post-Processing Software Application Techniques in the Audio-visual Recording of Hunan Chinese Dialect Survey for the Language Protection Project - Taking Adobe Premiere Pro as an Example,” Journal of Audio-visual Studies, no. 5, pp. 162–166, Apr. 15, 2022. [Online]. Available: https://d.wanfangdata.com.cn/periodical/Ch9QZXJpb2RpY2FsQ0hJTmV3UzIwMjUwMTA0MTcwMjI2EhNzbXNqLXNoYW5nMjAyMjA3MDQ3GghkcGk0Zm45NA%3D%3D.

6. X. Guanxiang, “Preliminary quality control in audio-visual recording of Hunan Chinese dialect investigation of China language resources protection project,” Journal of Language Protection, no. 3, pp. 241–243, Mar. 15, 2022. [Online]. Available: https://d.wanfangdata.com.cn/periodical/Ch9QZXJpb2RpY2FsQ0hJTmV3UzIwMjUwMTA0MTcwMjI2EhpRS0JKQkQyMDIyMjAyMjAzMTgwMDAwNjk0NBoIZHBpNGZuOTQ%3D.

7. X. Guanxiang, “Video and audio recording technology in Hunan Chinese dialect investigation of China language resource protection project,” Journal of Media Technology, no. 8, pp. 39–45, Aug. 02, 2019. doi: 10.3969/j.issn.2096-0751.2019.02.009.

8. X. Guanxiang, “Analysis of digital film and television special effects production technology,” Journal of Modern Information, no. 3, pp. 48–50, Jun. 16, 2017. doi: 10.3969/j.issn.1674-6708.2017.06.040.

CONTENTS

THE LIST OF CONDITIONAL SYMBOLS

ANN – Artificial Neural Network

API – Application Programming Interface

BCR – Barcode Recognition

CNN – Convolutional Neural Network

CNN-LSTM – Convolutional Neural Network-Long Short Term Memory

CSV – Comma-Separated Values

DL – Deep Learning

HMM – Hidden Markov Model

KYC – Know Your Customer

MCHSR – Multi-Channel Hidden State Recognition

MIDV – Multimodal Information Document Video

ML – Machine Learning

MRZ – Machine Readable Zone

MSE – Mean Squared Error

NLP – Natural Language Processing

OCR – Optical Character Recognition

OCR – Optical Character Recognition

PDF – Portable Document Format

RANSAC – Random Sample Consensus

ROVER – Recognition Output Voting Error Reduction

Tesseract – An open-source OCR engine

WFST – Weighted Finite-State Transducer

XML – Extensible Markup Language

INTRODUCTION

Justification for the Research Topic

In today's conditions of digital transformation, both society and business require automated processing of vast amounts of information, including data from visual sources. One such task is the recognition of document contents in videos, a dynamic and complex field situated at the intersection of computer vision, classical pattern recognition theory, natural language processing (NLP), and deep learning technologies. This task is focused on extracting textual and structural information from documents presented in video streams or recordings. This task manifests itself in the following areas of human activity:

1. Digitization of archival data. Many historical or legal documents are stored in the form of video recordings or film materials. Automating their analysis enables the extraction of text and structural data, accelerating the processing. For example, scanned handwritten documents from archives may be presented in motion or from unconventional angles in video format.

2. Real-time document control systems. In airports and checkpoints, video cameras capture documents such as passports and visas for automatic authenticity verification. This requires text recognition and data comparison with databases under time constraints.

3. Education and conferences. Recognizing text on boards, presentations, or screens during lectures allows for automatic transcription creation, facilitating access to knowledge. This is especially relevant for online education, where video streams may be the only source of information.

4. Automation in manufacturing. In logistics and industry, video surveillance systems analyze text on packaging, labels, or instructions. For example, video cameras can automatically capture serial numbers on moving goods.

At the same time, the task of recognizing text in videos differs from traditional optical character recognition (OCR) due to additional challenges.

Videos may contain dynamic camera movement, lighting changes, object occlusion, or noise. This requires more complex algorithms capable of handling factors such as:

1. Low video quality. Poor lighting, low resolution, or camera movement complicate text extraction. It should also be noted that many documents contain security features that make optical scanning of the content more difficult (watermarks, specific backgrounds, reflective (glare) fragments, etc.).

2. Multilingualism and font diversity. Texts in different languages need to be recognized, including writing systems with non-standard fonts or diacritical marks, requiring universal models.

3. Complex document structures. Often, it is necessary to recognize not only the text but also complex elements such as tables, graphs, and charts.

4. Real-time processing requirements. In security and automation systems, fast video processing is required without sacrificing quality.

All these factors make the task of document content recognition in videos not so much an engineering problem as a scientific challenge. In particular, leading global research centers are working on this issue, such as Google Research (USA, where Vision API and DocAI have been developed for text and document analysis), Stanford University (USA, where transformers and their application for multimodal data analysis are being studied), ETH Zurich (Switzerland, focusing on video and text analysis under complex conditions), the V.M. Glushkov Institute of Cybernetics of the NAS of Ukraine (Ukraine, developing methods for multimedia data analysis), and Kharkiv National University of Radioelectronics (Ukraine, researching text recognition for educational and industrial tasks).

Also, several prominent researchers are contributing to this field, including: Christopher Manning (USA), Andrew Ng (USA), Pradeep Ravikumar (USA), Simone Teufel (UK), Thomas Hofmann (Switzerland), Rada Mihalcea (USA/Romania), Hinrich Schütze (Germany), Anatoliy Teslyuk (Ukraine), Bohdan Rusyn (Ukraine), Vasyl Romaniuk (Ukraine), and Olga Ovcharenko (Ukraine).

Thus, document content recognition in videos is a task with enormous potential for application in business, security, science, and education. The development of this field requires the integration of computer vision, machine learning, NLP, and deep learning methods. Solving existing challenges, such as difficult shooting conditions and multilingualism, will ensure an even broader range of applications for this technology in the future.

Goal and Tasks of the Research

The **goal** of this work is to improve the eficiensy of object recognition systems in video streams by combining the results of processing multiple input observations.

To achieve this goal, the following **research tasks** need to be solved:

1. Analyze the principles of building modern document recognition systems;
2. Develop a mathematical model of an object recognition system in a video stream that allows the investigation of qualitative characteristics of the result and the time required to obtain it;
3. Investigate the impact of input data characteristics on the choice of optimal strategy for combining results of single image recognition;
4. Develop a method and algorithm for combining results of optical character recognition of string objects and conduct experimental analysis of their characteristics;
5. Develop a method and algorithm for stopping the recognition process of string objects in a video stream within the framework of the constructed mathematical model of the system.

Object of the Research: The object of the research is the process of recognizing string objects in real-time video streams.

Subject of the Research: The subject of the research is methods for combining the results of processing multiple input observations to enhance the quality and efficiency of object recognition in video streams.

Research Methods: To solve the tasks set, the dissertation research applies the principles and methods of systems analysis, mathematical modeling, mathematical statistics, pattern recognition theory, and decision-making theory.

Scientific Novelty:

1. A dynamic mathematical model of a real-time object recognition system in video streams has been proposed for the first time. In contrast to static analogues, the model enables formalisation of the frame-to-frame information accumulation process and ensures consistency in result integration over temporal sequences. This advancement allows the system's behaviour to be described as an anytime process and provides a rationale for selecting the optimal stopping point based on computational load. Experimental results demonstrate that the application of this model increases the accuracy of symbolic object recognition by up to 9% compared to single-frame approaches without integration, confirming the effectiveness of the proposed solution..

2. A method for integrating the recognition results of symbolic objects in video streams has been developed for the first time, taking into account alternative classifications for each character. Unlike existing approaches that rely on a rigid selection of a single option at each step, the proposed method operates with a set of probabilistic hypotheses per character and enables iterative refinement of the final result under uncertainty. This made it possible to align data integration using a weighted correction mechanism based on Bayesian estimation and to adapt the scheme to low-quality frames. Experimental results demonstrated an improvement in recognition accuracy by 7% compared to the benchmark analogue (ROVER)..

3. A method for stopping the recognition process of symbolic objects in video streams has been developed for the first time, based on threshold reduction of the expected distance between the current and the subsequent integrated result, formally framed as a monotonic stopping problem. In contrast to classical stopping rules that rely on a fixed number of frames or heuristic change detection criteria, the proposed approach evaluates the expected gain from the next step and makes termination decisions based on the predicted improvement in the result. This

enabled a significant reduction in the average number of processed frames (by 18%) without compromising accuracy, which is particularly critical for real-time systems operating on mobile devices with limited computational resources.

Practical Significance of the Results:

1. The developed methods for integrating frame-by-frame recognition results significantly improve the accuracy of final analysis results of video sequences, which is crucial for tasks such as document digitization and real-time data processing.

2. The possibility of adapting the ROVER method, originally designed to combine results from different recognition algorithms, to the task of combining results from different observations of the same object is demonstrated. This expands the scope of application of this method.

3. The developed integration algorithm shows higher accuracy than traditional approaches in the task of recognizing text fields in passport videos and other identification documents. This is particularly important for automating border control and identity verification processes.

4. The proposed method for stopping the recognition process allows the optimal moment for completing the processing of a video sequence to be determined. This reduces the excessive use of computational resources and increases the efficiency of systems, especially on mobile devices with limited resources.

5. The developed methods improve the performance of optical character recognition systems operating in real-time, such as those used for automating the processing of video recordings from surveillance cameras.

6. Experimental validation on real data and the use of the open-source Tesseract library demonstrate the applicability of the developed methods for integration into mobile and cloud applications.

7. The conducted research shows how the properties of input data affect the choice of optimal combination strategy. This allows for system customization for specific conditions, such as the presence of noise or defects in preprocessing.

8. The formal formulation of the stopping task in video sequence recognition and the proposed algorithm are novel and can be used to optimize the performance of systems focused on video stream analysis.

9. Experimental results obtained on the MIDV-500 dataset confirm the effectiveness of the developed methods for text recognition in documents used in verification systems, which opens up prospects for implementation in governmental and commercial applications.

10. The developed methods and algorithms for integration and stopping can be adapted to a wide range of computer vision tasks beyond text recognition, such as dynamic scene analysis and other object-related tasks.

Personal Contribution of the Applicant

All the main results of the dissertation work were obtained independently. In the co-authored publications, the applicant contributed the following:

- The mathematical model of the object recognition system in the video stream;

- The method and algorithm for combining the results of optical character recognition of string objects and conducting experimental analysis of their characteristics;

- The method and algorithm for stopping the recognition process of string objects in a video stream within the framework of the constructed mathematical model of the system.

Approval of the Results

The key results of the work were presented at the following seminars and conferences:

- 2020 China New Media Conference Main Forum (2020, China),

- Academic Seminar on Language and Character Application Research (2021, China),

- 5th China Computer Education Conference (2023, China),

- LIII Ukrainian Scientific and Technical Conference of the Subdivisions of Vinnytsia National Technical University (2024, Ukraine),

- XVII International Conference on Control and Management in Complex Systems (2024, Ukraine),

- LIV Ukrainian Scientific and Technical Conference of the Subdivisions of Vinnytsia National Technical University (2025, Ukraine),

Connection of the Work with Scientific Programs, Plans, and Topics

The results of the dissertation work were used in the execution of the departmental scientific research project №46K7 "Development of Methods and Technologies for Automating Control and Management Processes."

Publications

Based on the dissertation, 8 printed works have been published, including 3 articles in profile Ukrainian journals of category B and 5 journal articles.

Structure and Volume of the Dissertation

The structure of the dissertation is determined by its goals, objectives, and the aforementioned methods. The dissertation includes an introduction, four chapters, main conclusions, a list of references, and two appendices. The total volume of the dissertation is 169 pages of printed text, containing 24 figures, 8 tables, and 63 formulas. The list of references includes 134 items.

1 ANALYSIS OF THEORETICAL AND APPLIED SOLUTIONS FOR DOCUMENT CONTENT RECOGNITION ON REAL-TIME VIDEO

1.1 Automatic Document Input

Documentation support for production management, or office work [1], is an integral part of any enterprise, involving the creation, accounting, storage, and organization of document movement. The set of tasks related to the organization of document flow within an organization is called document circulation, which includes input, reception, registration, distribution, execution control, case formation, storage, reuse, etc. To automate office work in enterprises, electronic document management is introduced–a unified mechanism for handling documents in electronic form.

According to the official definition in Ukrainian legislation, a document is defined as a material medium with information fixed on it in any form, such as text, sound recording, image, or a combination thereof, containing attributes that allow its identification and intended for transmission over time and space for public use and storage [2]. A term closely related to the concept of a document, especially in the context of electronic document management, is the term "form," which refers to a set of informational fields (attributes) with a defined logical structure, as well as a logical and visual representation.

One of the aspects of electronic document management is the automatic input of documents – a method of automated data entry using predefined templates and document configurations. Automatic document input emerged as an alternative to manual input to minimize typographical errors and time costs. A typical technological process for automatic mass document input at an enterprise can be described by the following stages:

1. Distribution of the document flow into batches for separate processing.
2. Digitization of documents in the processed batch, i.e., converting the document from paper or another physical medium into electronic form. For paper

documents, this stage often involves scanning the batch of documents using a high-speed industrial scanner.

3. Preparation of digitized documents for recognition, i.e., applying primary electronic information processing methods. For scanned images of paper documents, this stage includes using image processing methods that improve recognition accuracy.

4. Applying recognition methods to convert the information contained in the document into electronic form for further use in the electronic document management system. This stage may sometimes include the extraction of certain fields (attributes) of the document, for which the recognition result is considered questionable or unreliable by the recognition system, followed by verification and correction by an operator.

5. Saving the resulting electronic document in a database and/or exporting it to a format suitable for electronic processing, such as XML, PDF, CSV, etc.

The distribution of the document flow into independently processed batches is the initial stage of the technological process of automatic document input, involving the division of the document flow into parts of limited size and/or grouping documents by type. Here and further, a document type is understood as a named set of its logical structure (title, a set of fields (attributes) with certain semantic and syntactic properties) and its representation structure on paper or another physical medium.

Digitization of a document is the defining stage of the technology for automatic document input, representing the transformation of the document from a physical medium into an electronic form suitable for further processing. For example, when digitizing documents through scanning, a flat (most often paper) document is digitally described as a color (multichannel) or grayscale (single-channel) image with color depth and resolution adjusted based on the technological capabilities of the scanning device and the specifics of subsequent document image processing algorithms. Another example of document digitization is its video or photo capture using a mobile device camera, often applied when efficient

document input is needed in a non-stationary mode. In this case, the document's electronic representation is its digital photograph or video stream, containing an ordered sequence of frames, each of which displays the document or a part of it.

Methods of primary electronic information processing, such as digital image processing, analysis, and establishing relationships between informative parts of frames in the video stream, etc., are used to facilitate the extraction of informative areas of the digital document image and to improve recognition accuracy. After primary processing, methods for determining the document's logical structure and extracting digital representations of informational fields (attributes) with subsequent recognition are applied. Depending on the nature of the input documents, automatic document input systems use methods of optical character recognition (OCR) [3], barcode recognition (BCR) [4], etc. Optical character recognition methods are sometimes classified based on their functional focus, such as methods for recognizing printed characters and text, typewritten characters, handwritten characters, handwritten text, as well as recognition of marks (for example, in multiple-choice questionnaires, ballots, etc.). If the syntactic and/or semantic properties of the document's fields (attributes) are known in advance, automatic correction of the results may be performed after recognition (for example, to correct the recognition results of the "Surname" field, a complete or incomplete frequency dictionary of surnames may be used) [5]. In some automatic document input systems, after the field recognition result is obtained, the reliability of the result is analyzed, followed by operator verification and correction [6; 7].

1.2 Mobile Document Management

Since the 2000s, there has been growing interest in methods for automatic document input using mobile devices. This is driven by the rapidly increasing computational capabilities of widely used mobile devices, such as smartphones and portable tablet computers, as well as the improved technical features of digital cameras installed on these devices. Interest in electronic document management systems, particularly in methods for automatic document input tailored for mobile

devices, is also fueled by the development of mobile application distribution systems, both corporate and aimed at the general public. According to a survey of mobile device users conducted in the U.S. in 2014 by Radium One [8], 88.2% of respondents used their smartphones more than 10 times a day, with 35.5% using them more than 40 times a day.

In the corporate sector, interest is growing in implementing document management (or parts of it) based on mobile document management–a type of electronic document management where users can perform operations with electronic documents via various mobile devices. According to a survey conducted by Litera Corp. in 2013, 97% of the surveyed business professionals used personal or corporate mobile devices for storing and processing documents [9]. Naturally, this raises the need for systems that implement automatic document input, using mobile device cameras as a "scanning" tool–digitizing the document through video or photo capture of the original.

Among regular users of mobile devices, such as smartphones or tablets, interest in purchasing goods and services via online services available from personal mobile devices is growing. According to the previously mentioned survey [8], 61% of smartphone users had made at least one mobile purchase in the last six months. According to a 2014 survey conducted in 18 European countries, 77% of respondents had made at least one mobile purchase in their lifetime (up from 72% in 2013) [10]. In most cases, such transactions require entering data from certain documents (for example, an identity document, bank card details, etc.), and this data often needs to be entered repeatedly, as storing it on a mobile device may lead to data leaks and misuse by malicious actors. The storage of sensitive personal data on internet servers is strictly regulated by law [11] and is also, though to a lesser extent, vulnerable to fraud attacks. This leads to an increasing relevance of automatic document input methods aimed at mobile devices, not only in the corporate sector but also in the field of mass e-commerce.

Another factor driving the relevance of mobile document management systems and mobile document recognition is the role of "Know Your Customer"

(KYC) procedures. According to these procedures, stock exchanges, banking organizations, and other financial institutions are required to accurately identify clients or counterparties for conducting financial transactions. To comply with requirements collectively referred to as the "Know Your Customer" principle ([12; 13]), customer-focused financial organizations are forced to verify the identity of users and counterparties for each transaction. As the share of transactions conducted remotely via mobile devices increases, the need for remote user identification necessitates remote document analysis, including identity verification documents.

As the implementation of technological, social, and commercial processes based on mobile devices and technologies has become commonplace in the modern world, systems for automatic document input and analysis on mobile devices continue to replace traditional stationary systems. The development of document analysis technologies using mobile devices and addressing the hardware limitations associated with them remains a relevant challenge.

1.3 Document Recognition Systems

The aim of this review section is to highlight the main stages of document image processing typical for automatic input systems and to describe their characteristics.

1.3.1 Digital Document Image

Classical document recognition and automatic input systems involve the use of a scanned document image as its digital representation. The image is generated during digitization using a flatbed or sheet-feed scanner and is characterized by several features: such an image typically has high resolution, as modern scanners are capable of producing images with several thousand dots per inch. The lighting of the document in such scanners is generally uniform due to the homogeneous artificial lighting, and the geometric representation of the document closely matches the original, with minimal distortions within the extended motion group.

The majority of works related to automatic document input and recognition on mobile devices consider a photograph of the document as its electronic representation and note the difficulties associated with preparing the document image for recognition and the recognition process itself [14].

Document images captured by a mobile device's camera have significantly lower quality compared to those obtained with a traditional digital scanner. When using mobile devices, the image preparation stage for recognition faces challenges such as uneven scene lighting, projective distortions of the document, nonlinear distortions (caused, for example, by the curvature of the paper), motion-induced distortions, noise, and defocusing [15]. All these conditions result in traditional image preprocessing methods, applied in automatic document input systems using digital scanners, not yielding the necessary effect, thus requiring special methods to enhance recognition accuracy and reliability.

1.3.2 Document Search and Localization

The primary task of document image processing in a recognition system is the accurate identification of the document within the image. This stage is often linked with the task of document type identification. In some specific cases, this stage may be omitted (for instance, when the document type is fully known and the image has no spatial distortions due to the digitization process specifics); however, in most cases, this stage is essential for further content analysis. The main challenges encountered during the document search in a digitized image include image distortions–both geometric (tilts, rotations, projective distortions, nonlinear distortions) and pixel-related (digitization noise, brightness distortions due to uneven lighting, etc.).

The main approaches to determining document tilt in an image can be divided into two groups: global and local [16]. Global approaches analyze features calculated across the entire image, such as projection histograms of image objects on various axes, lines at the borders of image regions, and so on. Local approaches use features significant only in limited areas of the image, such as common axes of

neighboring text connectivity components. Based on the locally calculated tilt estimates, a decision is then made about the global tilt value. In [17], the local tilt estimate is computed by identifying the direction with the highest number of transitions from black to white and back in a local window of a binarized image containing text. Papers [18; 19] describe methods for assessing local tilt, also using the geometric properties of the binarized text image. In [18], the search for chains of connectivity components corresponding to words or parts of words or text lines is performed using the region-growing method [20], and the direction within the window is selected based on the directions of these chains. In [19], a method based on analyzing the directions of individual letter strokes is proposed. In [21-23], the document tilt angle is determined using the Hough transform [24] to detect lines in the document image. In [25], the tilt detection method is based on the assumption that the main axis of the minimum-area bounding rectangle for document components containing text and other typical elements (such as barcodes, tables, etc.) coincides with the component's direction. The proposed algorithm for finding the minimum-area rectangle for a given component is fairly efficient and relies on the boundary-following method described in [26].

Considerable attention is also given to the task of projective correction for document images captured with mobile device cameras [27-31]. Most methods for correcting perspective (projective) distortions in an image are based on detecting the vanishing point of the perspective. These methods include searching for lines and intersections of these lines in the image or performing a texture and frequency analysis of the image components [32]. To find the vanishing point or document lines, one may either search for the document's edges (if the entire document or a full page is present in the image) or search for text line boundaries. Line detection in images is a well-documented problem [33], and numerous methods have been proposed to solve it, such as using the RANSAC algorithm [34], least squares line fitting [35], and the fast Hough transform [24].

Some methods are based on analyzing text line directions, which are necessary when document edges are not in the frame. In [36], a method based on

analyzing connectivity components (characters) in the image is proposed. Rectification (correction of projective distortion) in the proposed method is performed separately for each text line based on approximating its baseline.

In addition to methods that analyze individual document elements for image rectification (i.e., partially reducing the document localization task to determining the parameters of specific geometric distortions), there is another class of methods directly related to the localization of the entire document page image. One approach to page localization based on its visual representation is the generalized Viola-Jones algorithm [37] as a branching model of strong classifiers [38; 39] for detecting the entire page image under limited perspective distortions. However, the most widely used approach here is the search for keypoints (keypoints or feature points). The motivation for applying this approach to solve this problem is the resilience of these points to various image noise and distortions, which can significantly reduce the accuracy of structural and geometric analysis algorithms [40].

The described approaches, though successfully used for analyzing document images on individual images, are generally not extended to the case of multiple images of the same document and thus belong to static object recognition systems. Considering a video stream as a digital representation of the recognizable object, these approaches require appropriate adaptation.

1.3.3 Image Segmentation of a Document

After locating and correcting the document in the image, the next step is segmenting the (now rectified) image of the document into its constituent parts. This task depends on the document's structure and the further processing methods. However, two primary segmentation tasks can be identified, which rarely overlap and generally represent two independent stages in most recognition systems:

1. Segmentation of the document image into separate fragments – text blocks, text fields, lines, paragraphs, figures, formulas, stamps, etc.;

2. Segmentation of a text line or text field into individual characters or graphemes.

Since the segmentation of a document image into informational fragments largely depends on the document's structure and the specific recognition system used, a wide range of methods and approaches are offered in the literature to solve this problem. For segmenting an image into large homogeneous informational blocks, methods such as combinatorial analysis of straight-line elements detected via morphological image analysis [41] , document image decomposition through optimal layering of multiple Gaussian kernels [42] , or using fully convolutional artificial neural networks (ANNs) to identify pixels corresponding to the segmented fragment [43] are proposed. Additionally, structural methods for detecting text lines based on pathfinding in brightness graphs [44] and techniques partially utilizing machine learning to detect text line boundaries [45] are evolving.

A particular focus has been on detecting text lines and individual words, widely discussed in recent years under the "word spotting" or "text in the wild" approach. This approach arose from the broader task of searching for text in arbitrary natural images. Methods for solving this more general task include structural analysis of image regions followed by combinatorial selection of regions likely containing text [46-48] , applying machine learning techniques to local text features [48–50] , and the global use of deep convolutional ANNs to highlight text pixels in arbitrary images [49, 51–53] .

One of the most challenging tasks in text recognition, particularly in document recognition systems, remains segmenting a text line into individual characters [54, 55] . The segmentation of individual characters is complicated not only by general image distortions (which lead to "merging" of adjacent characters and other problems) but also by the wide variety of fonts and typefaces used in document text fields. Traditional document recognition systems often addressed the segmentation of a text line into individual characters by binarizing the text line image (or globally binarizing the entire document image) and then analyzing the "black" connected components. However, this method has become outdated in the

face of distortions typical of images taken with mobile device cameras, as selecting universal binarization parameters that minimize the number of "merges" and "splits" is exceedingly difficult. More widely used approaches to this task without binarization include analyzing the projection of the text line image onto the horizontal axis, pre-setting potential cuts of the line, and applying dynamic programming with cut significance and character recognition confidence scores between cuts as part of the final "quality" metric for segmentation [56, 57] . Other methods use machine learning techniques for trainable segmentation cuts between characters/graphemes: convolutional ANNs and recurrent LSTM networks (long short-term memory networks) help reduce heuristic parameters of the segmentation algorithm and increase resistance to distortions [56] .

It is also worth noting that methods excluding explicit segmentation into individual characters are gaining popularity among researchers. In these approaches, the assumption is made to combine deep convolutional ANNs with LSTM networks for training algorithms to recognize entire words without intermediate stages [58, 59] . Such methods significantly improve the algorithm's resistance to input image distortions and expand the domain of application (e.g., for recognizing handwritten lines, text printed in languages difficult to segment, such as Persian or Arabic, or lines of historical documents with hard-to-segment calligraphic fonts). However, the complexity of training neural network models that solve such a general task, combined with the high computational cost of recognizing each line, currently prevents the widespread use of these approaches in practical recognition systems, particularly considering the hardware limitations of image recognition systems on mobile devices.

1.3.4 Single Character Recognition

Optical recognition of objects in general, and particularly printed or handwritten characters, is one of the most important tasks in computer vision. Over the past few decades, a significant range of methods has emerged to solve the task of recognizing patterns of various objects. In classical terms, recognition systems

are classified by the type of information used in the recognition process into unsupervised systems, learning-based systems, deterministic, probabilistic, logical, structural, and combined systems [60].

Artificial Neural Networks (ANNs), originally aimed at modeling biological systems, have now become one of the most effective mechanisms for pattern recognition, particularly for character recognition. In several tasks, this method has demonstrated its ability to compete with human perception systems [61]. The inspiration for ANNs was a simplified model of the brain [62]. Current research and developments in the field of machine learning continue to advance ANN architectures and training methodologies for solving various tasks. A fundamental breakthrough in image analysis using ANNs was the introduction of convolutional neural networks (CNNs).

CNNs were first proposed by Yann LeCun [63], and their architecture initially involved two convolutional layers, two subsampling layers, and several fully connected layers to form the output. This architecture made the ANN's response invariant to spatial shifts of the input signal and allowed feature processing to be uniform across different local regions of the input image. Based on the original idea of CNNs, numerous neural network architectures have been developed, tailored to specific tasks. For example, AlexNet [64], which introduced distributed training across multiple GPUs; ZFNet [65], which improved hyperparameter tuning by increasing the size of intermediate convolutional layers and introduced a method for network visualization; and VGGNet [66], which demonstrated the importance of network depth in large-scale image classification tasks. Other architectures include [67-69].

The use of CNNs is currently the most accurate method for image recognition, and in some tasks, this method achieves results that can compete with human performance [61]. However, CNNs can show unstable results with minimal changes to the input image [70; 71], even when these changes involve just one pixel [72]. Another problem related to CNNs is their excessive "overconfidence"–

the confidence scores provided by CNNs for incorrectly recognized images can be indistinguishable from those for correct results [73].

The use of CNNs in mobile document recognition systems imposes additional constraints due to their higher computational complexity compared to some other methods, which increases significantly with the input size. To effectively use CNNs in document recognition systems that involve multiple images (e.g., recognizing objects in a video stream), highly efficient methods are needed to combine recognition results from individual images.

1.3.5 Post-Processing and Language Models

In document text field recognition tasks, it is a mistake to simply concatenate the independent classification results of characters, as many fields have a known syntactic and semantic structure. The set of rules regarding permissible characters in certain positions of the text field, their interdependencies within the field, and the dependencies between the values of different text fields in the same document is referred to as the context of the text field. The construction of context-dependent algorithms for refining recognition results generally depends on the specific task, and new approaches to solving applied problems in this area are constantly being proposed [74; 75]. Thus, statistical correction ("post-processing") of recognition results is one of the key components of modern optical document recognition systems.

The context of a document field typically comprises the following elements:

1. Syntax: Principles regulating the textual format of the field;
2. Semantics of the field: Rules based on the meaning of the field or its parts;
3. Semantics of relationships: Rules based on the structural and semantic connections of the field with other fields in the document.

There are many algorithms for statistical post-processing of recognition results, which differ in the language models used for the recognized object, in the algorithms for directly correcting recognition results, and in their areas of applicability. Among the most well-known and widely used methods are those

based on Hidden Markov Models (HMM) [76; 77], finite automata, N-gram and dictionary methods [5; 78], as well as mechanisms using Weighted Finite-State Transducers (WFST) [79].

With information about the semantic and syntactic structure of the document and the recognized field, it is possible to build a specialized post-processing algorithm for each specific field. However, given the need to support and develop recognition systems and the complexity of their development, methods and tools that allow the construction of reasonably good post-processing algorithms with minimal effort (on the part of the recognition system developers) are of particular interest. Such methods would work with a broad class of documents and fields. The methodology for configuring and supporting such an algorithm would be standardized, and the variable component of the algorithm structure would only be the semantics and syntax of the processed field. A sufficiently general model allowing for the construction of a universal post-processing algorithm for recognition results is described in [79]. This model relies on the data structure of Weighted Finite-State Transducers (WFST).

The advantages of this approach are its generality and flexibility. For example, the error model can be easily extended to account for deletions and insertions of characters (this is done by adding transitions with an empty output or input symbol, respectively, to the error model). However, this model also has significant drawbacks. First, the language model must be represented as a finite weighted transducer. For complex languages, such an automaton can become quite cumbersome, and in the event of changes or refinements to the language model, rebuilding the automaton may be required. Additionally, the composition of three transducers usually results in an even more cumbersome transducer, and this composition is calculated every time post-processing of a recognition result is performed. Due to the complexity of the composition, finding the optimal path is often done using heuristic methods [79], such as A^*-search [80].

1.3.6 Evaluation of Recognition Accuracy

When assessing the quality of recognition systems and algorithms, concepts such as accuracy and confidence are used. The accuracy of a recognition system reflects the probability (probability estimate) of correct object recognition. Since the true value of the probability of correct recognition cannot generally be determined, the accuracy of recognition systems is usually assessed by calculating the posterior probability of correct recognition based on a predefined input data set. This "reference" data set often forms the basis for formulating the recognition task: given the reference data set, the recognition problem can be formulated as the task of maximizing the posterior probability of a correct answer for objects in this data set.

Another important quality indicator of an object recognition system is its confidence (some sources also use the similar term "reliability of recognition" or "accuracy of recognition"; to avoid possible term collisions, the term "evaluation of reliability" will be used hereafter). In object recognition systems of varying complexity, tasks often arise related to modeling the interaction of the system with the user, as well as the interaction of subsystems with each other. One of the most important aspects of such interaction models is the system’s response to recognition errors. The evaluation of the reliability of a recognition system reflects its ability to a priori assess the accuracy of its own results.

The task of evaluating the reliability of recognition results within a unified model of document automatic processing system interaction was addressed in the work [6]. In this paper, the author described a scheme for constructing a function to evaluate the effectiveness of the rejection rule, based on the computed posterior probabilities of classifier errors and the costs of these errors set externally. The simplest rejection rules for recognition results (the first alternative rule, the two-alternative rule) were also considered, and general principles for constructing complex rejection rules were formulated. This section will attempt to implement a general method for constructing a reliability function and the corresponding decision rule, based on predefined predictors (reliability features) computed from

the classifier's alternative vector or the original image, while minimizing the penalty functional.

Research on optimal rejection methods for recognition results continues in both directions: developing rejection methods for individual classifiers and modeling optimal rejection schemes in systems with multiple classifiers or human-machine interaction systems [81].

1.3.7 Using Multiple Input Images

Within the framework of object recognition in real-time video stream, where the objective is to identify a single object in individual frames, there are two primary strategies:

1. Align the object images initially and then recognize the object in the aligned frames.

2. Perform recognition on the object in each frame separately and then integrate the results from these recognitions.

Systems that leverage mobile devices for automatic document input offer distinct advantages, both in terms of user experience and the effectiveness of object recognition. Specifically, in a system where mobile device cameras are used for document scanning, the digitization process involves capturing video or photos of the document.

In video recording, the document is represented not by a single image but by a series of frames, each containing an image of the same document or its segment. This allows for repeated recognition of the same object throughout the video stream, enhancing both the accuracy and reliability of the recognition process in real-time.

Nevertheless, incorporating mobile devices in tasks such as object recognition and automatic document input presents challenges. In addition to the limited processing capabilities of mobile devices, there are various distortions introduced by the optical characteristics of mobile device cameras [82], as well as challenges related to the document capture process using these devices [15; 83].

Some of the recognition errors arising from these challenges can be mitigated by performing multiple recognitions of the same object across different frames, which in turn presents the task of determining the most effective strategy for integrating the frame-by-frame recognition results.

Depending on the model used for classification and how the classifier's estimates are interpreted, various strategies for combining the results can be considered. When the classification result is represented as a pair $\langle k, \phi \rangle$, where k is the class label and ϕ is the classifier's confidence score (without alternatives), a voting-based approach is often employed to select the optimal outcome by maximizing a specific quantitative criterion. A generalized form of this criterion is presented in [84]:

$$SCR\left(\langle k, \phi \rangle\right) = \sigma FRQ\left(k\right) + \phi\left(1 - \sigma\right), \tag{1.1}$$

where $FRQ\left(k\right)$ denotes the frequency of class k among the combined classifier outputs, and σ is a learnable parameter of the algorithm. One widely adopted approach, often used in speech recognition systems to integrate the results of multiple algorithms, as well as for merging text string recognition outputs [85; 86], is the Recognizer Output Voting Error Reduction (ROVER) method [87].

Figure 1.1 illustrates examples of text areas in a real-time video stream, while Table 1 displays the corresponding recognition results for each text area. It also shows the combined results obtained using the ROVER method [87] and their changes over time.

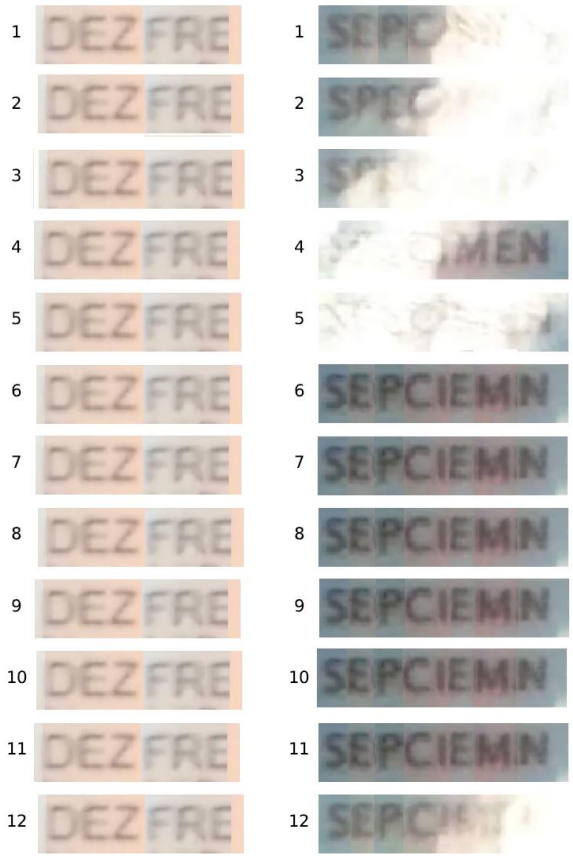


Figure 1.1 – Samples of text areas images in the real-time video stream from the MIDV-500 dataset [88] (video clips TS06, field 1, and NA11, field 1).

Table 1 – Samples of frame-by-frame and integrated text field recognition results, with correct results highlighted.

No.	Frame-by-frame	Шntegrated	Frame-by-frame	Шntegrated
1	0FZFRE	0EZFPE	-	-
2	‘ERE	0EZFPE	SE	
3	OEZ	0EZEPE	SE’	
4	DEZERE	0EZEPE	A	
5	‘EZIP	OEZEPE	B ‘ I ‘ .1	
6	ERE	OEZEPE	‘(EMN	
7	FERDEZ	OEZEPE	SEPCIEMN	E
8	FERDEZ	DEZERE	SEPCIEMN	IFMN
9	FERDEZ	DEZERE	SEPCIEMN	CPEMN
10	FERDEZ	DEZERE	SEPCIEMN	SEPC1FMN
11	FERDEZ	DEZERE	SEPCIEMN	SEPCIEMN
12	RERDEZ	DEZERE		SEPCIEMN

For a more general classification result model, where membership estimates ϕ_k are interpreted within the framework of the Bayesian model (i.e., the membership estimate ϕ_k represents the posterior probability of the image ρ_i belonging to class k_k), several combination rules are discussed in the literature [89; 90]. The foundational rules are outlined in [90], which is considered a key reference for combining the outputs of multiple classifiers within the Bayesian framework. These rules are presented here, but applied specifically to combining the classification results of multiple images processed by a single classifier:

1. Product rule:

$$MUL(\mathrm{P})(\phi) = P(\phi|\mathrm{P}) = \frac{1}{P(\phi)^{N-1}} \prod_{i=1}^{N} C_{\Sigma}(\mathrm{P}_i)(\phi); \tag{1.2}$$

2. Sum rule:

$$SUM(\mathrm{P})(\phi) = \frac{1}{N} \sum_{i=1}^{N} C_{\Sigma}(\mathrm{P}_i)(\phi); \tag{1.3}$$

3. Minimum rule:

$$MIN(\mathrm{P})(\phi) = \left(\min_{i=1}^{N} C_{\Sigma}(\rho_i)(\phi) \right) \left(\sum_{k=1}^{K} \min_{i=1}^{N} C_{\Sigma}(\rho_i)(\phi_k) \right); \tag{1.4}$$

4. Maximum rule:

$$MAX(\mathrm{P})(\phi) = \left(\max_{i=1}^{N} C_{\Sigma}(\rho_i)(\phi) \right) \left(\sum_{k=1}^{K} \max_{i=1}^{N} C_{\Sigma}(\rho_i)(\phi_k) \right); \tag{1.5}$$

5. Averaging rule:

$$AVG(\mathrm{P})(\phi) = \left(\operatorname{avg}_{i=1}^{N} C_{\Sigma}(\rho_i)(\phi) \right) \left(\sum_{k=1}^{K} \operatorname{avg}_{i=1}^{N} C_{\Sigma}(\rho_i)(\phi_k) \right). \tag{1.6}$$

In the case where the estimates ϕ_k are interpreted as fuzzy evidence of class membership or abstract indicators of confidence, combination methods based on Dempster-Shafer theory are used [91; 92]. Additionally, studies addressing heterogeneous classifier combination methods discuss strategies for weighting the significance levels of classifiers [93], learning-based methods for combination rules that account for the statistical characteristics of the classifiers being combined [94–96], and methods that, while not relying on the statistical properties of the classifiers, use a multi-set framework to build a model for group classification of objects [97; 98].

While numerous methods for combining recognition results from different techniques have been extensively discussed, less attention has been given to the challenge of integrating recognition outcomes from multiple images of the same object. This problem is particularly significant when developing object recognition systems for video streams, as it remains a key hurdle in ensuring accurate and efficient object detection across multiple frames.

In addition to combining recognition results from individual frames, video stream recognition introduces a new challenge: determining when to halt the recognition process. This "stopping" task is particularly critical for Instantaneous visual recognition and document recognition systems deployed on portable devices, where the time taken to reach a result is just as crucial as the precision of that result [99–101].

From a system design perspective, object recognition in video streams – where individual frames are processed independently and merged into a cohesive outcome can be conceptualized as an "anytime" algorithm [102]. An anytime algorithm is a time-constrained computational method that can provide the best available solution at any given moment, without needing to complete the entire computation to produce a final result. Among the characteristics of anytime algorithms, video stream recognition exhibits two essential properties: interruptibility (the ability to terminate after processing any frame and accept the integrated result as final) and monotonicity (the property that the quality of results

does not deteriorate as processing continues). Furthermore, in the context of video stream object recognition, there may be a lack of definable quality at any given time, meaning that the quality of the ongoing result may be unknown or unmeasurable when the algorithm is executed. Notably, other examples of anytime algorithms in recognition and computer vision involve systems that can provide incremental results, initially focusing on easily recognizable objects and progressively identifying more complex ones [103].

The stopping problem, in its broader context, has been extensively analyzed in the fields of mathematical statistics and decision theory [104–107]. Several variations of this problem have been examined, including the "selective bride problem" [108], the "house selling problem" [109], and the "average optimization problem" (also referred to as the S_n-by-n^1 problem) [110], among others.

One of the most extensively studied stopping problems, the proofreading problem [106; 111], is particularly relevant to the stopping problem of recognition of objects in a real-time video stream. The problem can be outlined as follows: a paper, digitized with M errors, undergoes a series of proofreading passes, each of which corrects P_i errors at a fixed cost T. Any remaining errors in the final version of the manuscript incur a penalty. After each ith proofreading pass, the decision must be made whether to send the manuscript for publication or to continue the proofreading process, aiming to reduce the overall expected cost. Several solutions to this problem have been proposed [111; 112], each founded on distinct assumptions regarding the distributions of M and P_i. Additionally, there are other variations of the problem, such as the automatic software testing problem [113].

The task of proofreading shares several similarities with the task of stopping the object recognition process in a video stream. Both involve a sequence of steps in which decisions must be made at each stage, balancing the cost of further processing with the potential improvement in results. Specifically, object recognition is carried out across a series of frames, and in order to obtain the next recognition result, certain costs must be incurred, such as the time required for

preprocessing the next frame and executing an additional iteration of the recognition algorithm. Assuming that recognition results accumulate over time, after processing each frame, a decision must be made: either continue by investing in further observation, hoping for a more accurate result, or stop the process and accept the current recognition outcome. In this scenario, the expected loss can be formulated as a linear combination of two factors: the expected number of frames processed and the discrepancy between the expected recognition result and the true value, evaluated within a predefined metric.

Nevertheless, there are key differences between the two tasks that should be emphasized:

1. Nature of Results: In many formulations of the proofreading task, it is assumed that the variable P_i only takes non-negative values. This means that proofreading typically either eliminates errors or, at the very least, does not introduce new ones. In contrast, object recognition in a video stream lacks such a guarantee. The recognition result from the next frame—or the cumulative result after processing the current frame—does not always improve in proximity to the true value. However, it is generally assumed that the algorithms for integrating recognition results (as discussed in Chapter 3) are designed so that, on average, recognition results for multiple versions of the same object tend to be more accurate than results derived from a single frame.

2. Cost Assessment and Confidence Evaluation: Solutions to the proofreading task, or its variants, often rely on the ability to evaluate the cost of stopping the process or assessing the cost difference at different stages, given that the observed value P_i directly affects the penalty function. In contrast, the object recognition process in a video stream involves estimating the distance between the new recognition result and the true value using confidence evaluation methods. These evaluations may be overly optimistic [114; 115] or, in some cases, unavailable to the decision-maker.

A comprehensive discussion on the proofreading task can be found in [111], where the optimal stopping rule is formulated using the concept of monotonic

stopping tasks. The theory surrounding monotonic stopping tasks is extensively covered in [116] and [106], and this theory will be applied in this work to develop a stopping method tailored to the object recognition process in a video stream.

1.4 Conclusions on the Analytical Part

The advancement of automatic document input systems and optical object recognition technologies is closely tied to progress across multiple disciplines, including image processing, computer vision, machine learning, software system design, and computing technology, especially in mobile devices. Each of these fields contributes to the efficiency of various components and subsystems, and the development trends within each field influence the algorithms used to address specific challenges in automatic document analysis and recognition.

A considerable amount of research in the literature focuses on systems for processing document images, algorithms for searching documents and fragments, image segmentation techniques for documents, and methods for detecting and classifying objects and document attributes. The ongoing interest in developing highly accurate algorithms for these tasks, coupled with the frequent publication of relevant research in major journals and specialized conferences, underscores the importance of this topic. Both structural image analysis methods and machine learning-based approaches are explored. Notably, recent trends (over the past 3-4 years) highlight the increasing application of deep learning techniques for addressing complex tasks, such as searching for entire text fragments in natural images and performing end-to-end text string recognition without the need for separate character segmentation.

However, while these deep learning-based methods show promising recognition accuracy and robustness against noisy input data, their practical use in industrial recognition systems and automatic document input is limited by their high computational demands. These high computational costs make such methods particularly unsuitable for use in mobile computing devices, where processing power and efficiency are critical.

Object recognition in video streams is an emerging area of study within the broader field of document recognition and automatic input systems. Existing literature primarily focuses on theoretical approaches to group classification of multi-feature objects and attempts to apply multiple recognition methods, often combining the results of various classifier outputs. However, a comprehensive exploration of methods aimed at improving recognition accuracy through multiple recognitions is still a relevant and under-researched topic. When dealing with video streams, as opposed to a single image, the challenge of integrating recognition results from different observations of the same object becomes particularly important.

A key issue that remains under discussion is the task of determining when to stop the object recognition process in a video stream. This problem is particularly critical for real-time computer vision and document recognition systems running on mobile devices, where processing efficiency is paramount. Although a variety of stop criteria have been proposed and theoretically developed in the literature, further work is needed to effectively apply these theories to object recognition systems in video streams.

1.5 Detailed Dissertation Tasks

Based on the analysis of the core principles underlying modern automatic document input systems and optical document recognition technologies, the following key tasks have been identified for this dissertation:

1. Develop a mathematical model for an object recognition system in a video stream, enabling the investigation of the qualitative characteristics of the recognition results and the time required to achieve them.

2. Analyze the influence of input data characteristics on the selection of the optimal strategy for combining recognition results.

3. Design an algorithm for combining optical character recognition results of a string object and evaluate its performance and characteristics.

4. Propose a method for determining when to stop the object recognition process in a video stream, based on the constructed mathematical model of the system.

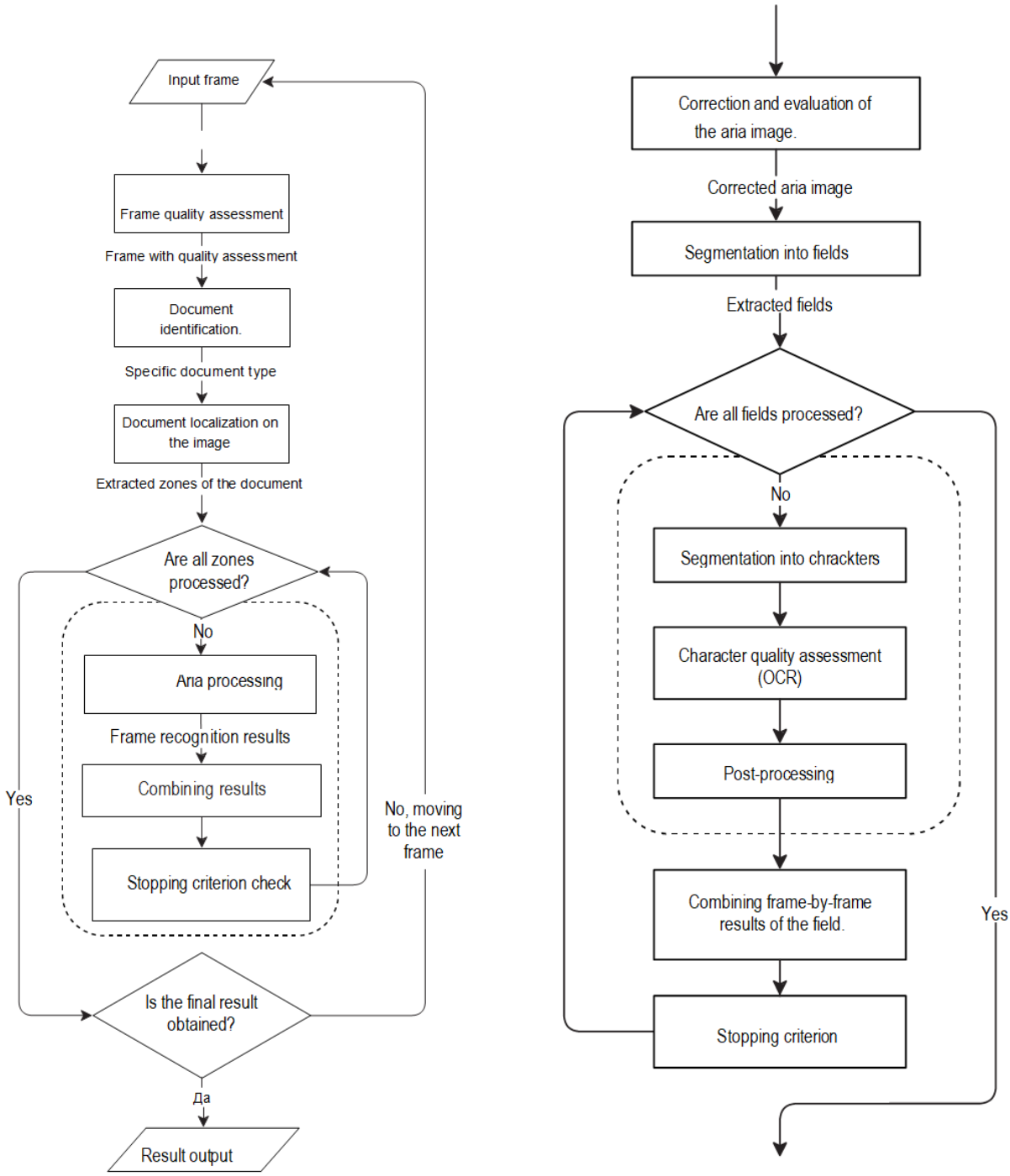
5. Create an algorithm for stopping the recognition process of a string object and perform an experimental analysis of its performance.

2 THE MODEL OF THE SYSTEM FOR OBJECT RECOGNITION IN THE REAL-TIME VIDEO STREAM

2.1 Introduction

The integration of technological, social, and commercial processes based on mobile devices and technologies has become a routine part of modern life. Technical vision systems using mobile technologies, such as automated systems for document input and analysis on mobile devices, are progressively replacing traditional stationary systems. Consequently, the development of technical vision technologies that can function under the hardware constraints of mobile devices is becoming increasingly relevant.

Conventional recognition and automated input systems rely on a scanned image or photograph of an object as its digital representation. When using mobile devices to digitize images of recognizable objects, an additional opportunity arises to utilize the video stream from a digital camera, rather than only individual photographs or frames. Capturing an image of an object with modern mobile devices involves an operator "aiming" the camera at the object, with real-time display of video stream frames on the device screen for operator control. If image processing is performed on a single frame, the information contained in the preceding captured frames is only indirectly utilized (by the operator). Viewing the entire video stream as the digital representation of the object allows for the use of significantly more visual information [99]. The diagram of the considered systems for automated document input from a video stream is shown in Figure 2.1.



a) General scheme b) Zone processing scheme

Figure 2.1 – Frame processing scheme in a document recognition system within a video stream. On the left (a) is the general scheme, and on the right (b) is the scheme of the document zone processing block (outlined with a dashed line in the general scheme).

The use of video streams enables tasks that are not feasible when analyzing a single photograph. External shooting conditions may result in significant distortions of the object in a single image [15]. For instance, a glare from an extended light source appearing on the glossy surface of a flat object is illustrated in Figure 2.2.

Since the geometric position of the captured object typically changes between frames in a video stream, the glare also "shifts," allowing hidden details of the object to be captured in another frame. Another important class of objects involves those whose detection and recognition are impossible in single frames– such as holographic security elements, which may appear indistinguishable from glare or patterns in individual images [15].

Figure 2.2 – Process of capturing an identification document using a mobile device (a mock-up of a German ID card is used as the document).

In such conditions, the task arises of selecting an optimal strategy for combining the results of frame-by-frame recognition. This topic is scarcely described in the literature, with the closest related field concerning the combination of recognition results of the same object by different classifiers [85; 89; 90].

Besides basic evaluation merging strategies, works addressing heterogeneous classifier result combination discuss strategies for weighting classifier significance levels [93], training combination rules that account for statistical features of the classifiers being combined [94; 96], and methods independent of classifier-specific statistical features but leveraging multiset apparatus to construct a group classification model [97; 98].

The main distinction of the video stream as a digital representation of a recognizable object lies in the sequence of observations of the same object, which differ from each other. Let us analyze the reasons why object recognition results might be erroneous, assuming the system operates deterministically–that is, it always produces the same recognition results for identical input data under the same external conditions. Consequently, any error arises from the system's inability to distinguish an object of a specific class. Recognition errors can be broadly divided into three categories:

1. Errors due to imperfections in the recognition algorithm: These "internal" errors stem from the object recognition system itself and may occur even with ideal functioning of other subsystems. This error class is an inherent attribute of any recognition system, regardless of the input model.

2. Errors due to preprocessing defects: A single-image recognition system is typically part of a larger complex, with input images being the result of other subsystems (see Figure 2.3). Consequently, errors may arise from imperfections in preceding subsystems. For instance, suppose an error occurs during segmentation of a text line image into individual character images, leading to incorrect determination of the right boundary of the image of the letter "P," resulting in the loss of the bridge between two horizontal strokes. From the perspective of the single-character recognition system, the resulting image may become indistinguishable from the letter "F."

3. Errors due to environmental noise: Such errors occur when external conditions render the image of the recognizable object indistinguishable from that of an object of another class. For example, consider capturing an identity document

containing a field labeled "Name" with the actual value "HANNA." This field is inscribed on a white background, and the document is covered with a protective glossy layer. During the capture, a glare from an external light source completely obscures the letter "H," leaving the other letters unchanged. Consequently, the image of this field becomes indistinguishable from that of the field "ANNA" on a similar document.

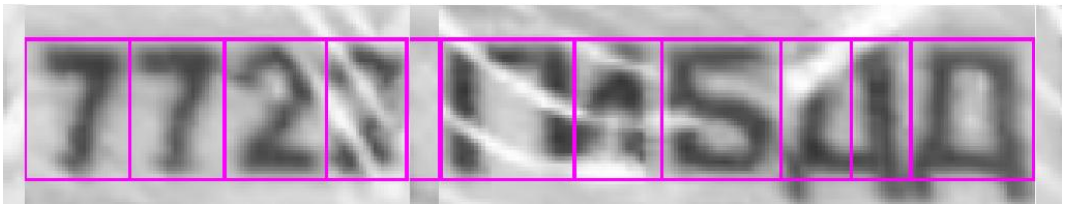


Figure 2.3 – Example of incorrect segmentation of a text string into individual characters under conditions of image blur and defects associated with the protective holographic layer of the document.

For a single-image recognition system, errors caused by environmental noise or preprocessing defects result from distortions in the input image. When multiple observations of the object are available, it can be expected that the impact of environmental noise and preprocessing defects will vary among these observations. However, even when the single-object recognition system is fixed, regardless of preprocessing, errors caused by imperfections in the classification model persist. Modern research demonstrates that the most effective method of image recognition–convolutional neural networks [63; 64], which in certain tasks achieve results comparable to those of humans [61]–may still produce unstable results with minimal changes to the input image [70; 71], even if these changes involve just a single pixel [72]. Thus, even with the most accurate recognition method, relying on a single input image of the object, it is impossible to separate the useful signal from noise, whose influence can significantly alter the outcome.

Consequently, by considering a video stream as the digital representation of an object instead of a single image, it becomes possible to reduce errors due to the

variability of noise across individual frames of the video stream–something that classical object recognition systems lack.

One method for analyzing multiple images of the same scene to reduce the impact of optical system noise and defects caused by uncontrolled shooting conditions is the "super-resolution" technique. This involves obtaining a high-resolution image from multiple lower-resolution images of the same object. This task has received considerable attention in the literature, with numerous approaches proposed that account for the specifics of the final task of image processing and object or scene recognition [117; 118]. However, as noted earlier, further processing of the resulting single image of the object remains susceptible to recognition algorithm errors, particularly the instability of convolutional neural networks.

2.2 The Model of the System for Object Recognition in the Real-Time Video Stream

Consider the model of a system for recognizing a single object ρ. Let there be a set containing N classes $K = \{k_1, k_2, \ldots, k_N\}$. For example, when considering the task of recognizing individual characters in the "Surname" field of a Ukraine citizen's passport, the set of classes consists of the Ukrainian alphabet with the addition of space and hyphen characters. In the task of document page typization in an image after localizing its boundaries and projective correction, the set of classes can represent a collection of document page types available for further processing. It is worth mentioning separately that sometimes in object and phenomenon recognition tasks, the presence of an "empty class" is allowed, which should be the system's response to an input image of an object unknown to the system or to an image that does not contain an object.

Let an image of an object $\mathrm{P}(\rho)$ from a certain set of all possible images P be given, and within the framework of the interaction model between the recognition system and the user/operator (or with other system components), there

exists a class $k^*(\rho) \in K$ to which the object ρ belongs. The task of recognizing a single object image consists of determining this class. The result of the recognition system's work can generally be represented as a well-defined mapping from the set of classes K to the set of membership scores $\hat{f}: K \to \square$. Given that the set of classes K contains exactly N elements

$$\hat{f}\left(\mathrm{P}(\rho)\right) = \left\{(k_1, \varphi_1), (k_2, \varphi_2), \ldots, (k_N, \varphi_N)\right\}, \tag{2.1}$$

where $\varphi_i \in \square$, $i \in \{1, \ldots, N\}$, represents the real-valued membership scores of object ρ to class $k_i \in K$ under the condition that the image of object $\mathrm{P}(\rho)$ is observed. The final classification decision is made for class $k^*\left(\mathrm{P}(\rho)\right) = \arg\max \hat{f}\left(\mathrm{P}(\rho)\right)$. The trivial scheme of an object recognition system within the described model is illustrated in Figure 2.4.

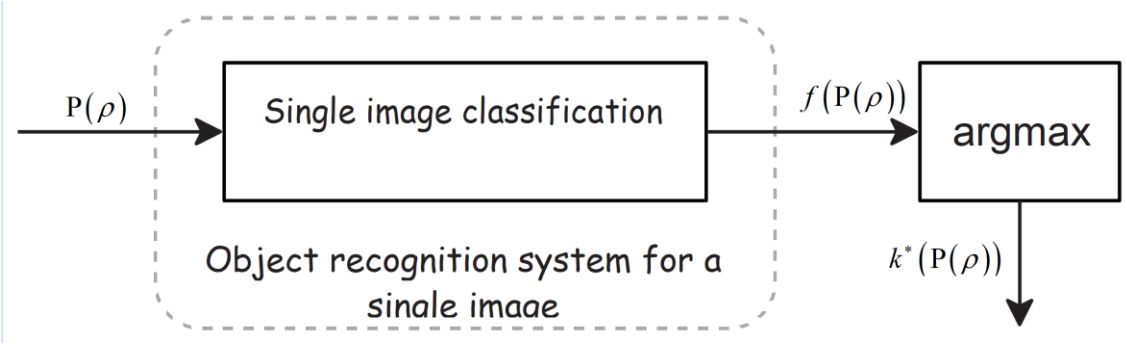


Figure 2.4 – Trivial Scheme of a Single Object Recognition System.

If we exclude the validation process of recognition results and the process of training the parameters of the recognition system (in cases where machine learning methods, such as artificial neural networks, are used for classification), and focus directly on the recognition process, such a recognition system is static and does not involve feedback.

Now, let's consider the task of recognizing object ρ in a video stream. The video stream is generated by a capturing device that provides a sequence of frames, each of which is an independent image of object ρ. Under the condition of a fixed

number of frames, the task of recognizing an object in a video stream can be treated as a static system, similar to the one presented in Fig. 2.4, but with a more complex input model. Thus, the sequence of M frames can be considered as a set of images of object ρ as $\mathrm{P}(\rho)=\{\mathrm{P}_1(\rho),\mathrm{P}_2(\rho),\ldots,\mathrm{P}_M(\rho)\}\subset\mathrm{P}$. Meanwhile, the output model of the system remains unchanged.

Implementations of such a system may differ in their approaches to data integration. A trivial approach might consider the classification process as a "black box" that processes multiple images simultaneously (scheme in Fig. 2.5a. Other options partially or fully use methods for recognizing individual object images and perform integration either at the input image level (Fig. 2.5b) or at the recognition results level of each individual image (Fig. 2.5c).

However, the presented static models of the object recognition system in a video stream do not fully reflect the recognition scenario using a mobile device, as these models assume a set of frames as input, without ordering, and do not account for changes in the system's state during the capture process. Additionally, given the hardware limitations of mobile devices, storing and processing multiple images may be impractical or impossible. To more accurately correspond to the object recognition process in a video stream on a mobile device, it is proposed to consider a dynamic model with discrete time.

For the purposes of formalization, let us represent the video stream as a sequence of images of the object generated over time. Thus, discrete time is given as $t=0,1,2,\ldots$, and the video stream contains images of the observed object $\mathrm{P}_t(\rho)\in\mathrm{P}$. This discrete model of the video stream corresponds to the principles of representing a coded video stream in software systems [119].

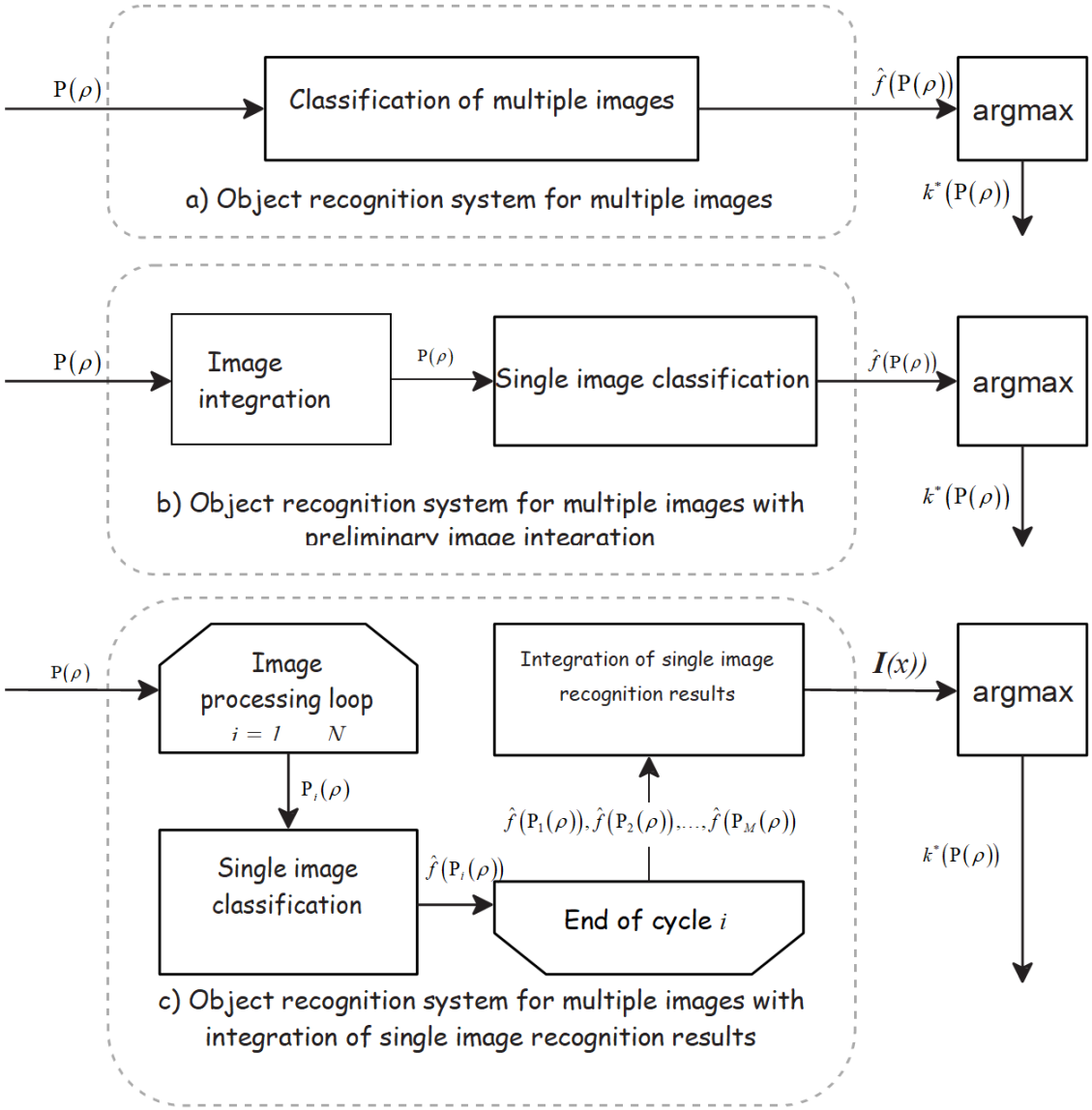


Figure 2.5 – Options for static recognition systems of multiple object images.

To define the object recognition system in a video stream, which is generated independently, it is necessary to define a service model that acts as an intermediate layer between the video stream and the direct flow of images processed by the recognition system. The most trivial service model is one where images generated during the processing of the previous image by the recognition system are discarded. If storing a collection of images is possible, an alternative service model is one with a buffer that allows incoming images to be accumulated and delivered to the system on request at any arbitrary time, without limitations associated with the discretization of image generation by the source. From the perspective of the recognition system of the image sequence, the set of recognition

methods and algorithms, as well as result integration, do not depend on the service model. Therefore, in this work, it will be assumed that at any time t, the "current" image $\mathrm{P}_t(\rho)$ can be captured, and during system loading periods, images may be discarded.

The recognition system maintains some internal state $v_t \in V$, which changes over time. The time Δ required to obtain an updated result after inputting the next image $\mathrm{P}_t(\rho)$ is, in general, a function of the image and the system's internal state: $\Delta = f\left(\mathrm{P}_t(\rho), v_t\right)$, which may be computationally infeasible at time t. The recognition result, which takes into account the information contained in the image captured at time t, may only be available at time $D(t) = t + \Delta$.

At the initial moment in time: $t = 0$, the system's internal state is initialized to v_0. Let image $\mathrm{P}_t(\rho)$ be captured at time t, which is then fed into the recognition module $\hat{f}$. The recognition result $\hat{f}\left(\mathrm{P}_t(\rho)\right)$ becomes available at time $t' \geq t$ and is stored in the system's memory module (i.e., it becomes part of the state $v_{t'}$). After that, the results of object image recognition accumulated up to that point are combined, and at time $D(t) \geq t'$, the recognition result $Y_{D(t)}$ is output. After the result is output, the next image $\mathrm{P}_{D(t)}(\rho)$ is captured, and the process continues. Thus, the result $Y_{D(t)}$ takes into account the information contained in the images with indices $0, D^1(0), D^2(0), \ldots, t$ (the superscript sign in the function $D(t)$ indicates multiple composition of the function, not exponentiation). The quality of the result is characterized by how close result $Y_{D(t)}$ is to the true value $w(\rho)$ of object ρ, according to a certain metric. The diagram of the described recognition system is presented in Fig. 2.6.

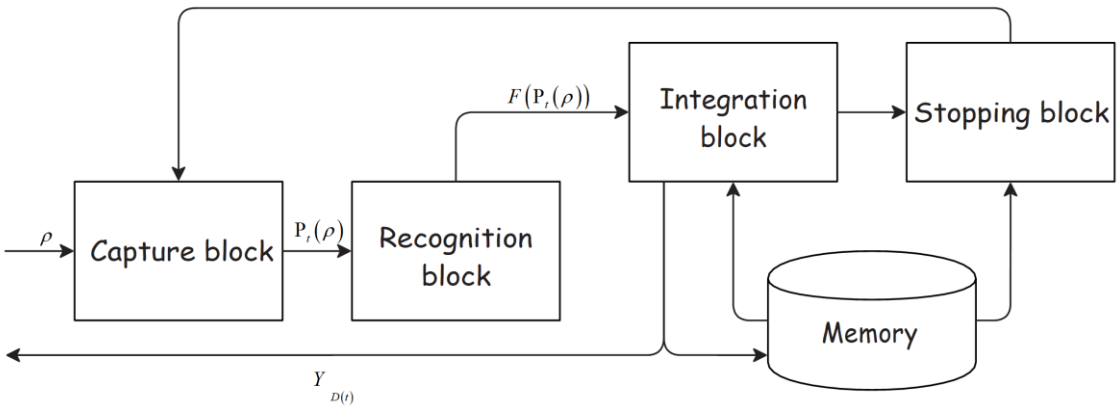


Figure 2.6 – Diagram of the object recognition system in a video stream with pause.

The feature extraction and object classification methods applicable in static systems (see Fig. 2.5) are also applicable in the dynamic model. However, the dynamic model of the object recognition system in a video stream has a number of specific properties. First and foremost, it is important to highlight the increased influence of the performance of single-image recognition algorithms on the system's output. Indeed, reducing the time Δ required to recognize a single image $\mathrm{P}_t(\rho)$ allows more information about the object ρ to be processed in the same absolute time (i.e., within the same time frame from the perspective of the user/operator). In addition, within such a system, tasks arise that are atypical for traditional object recognition systems on images. The first such task is obtaining the result $Y_{D(t)}$ – the task of combining (integrating) the recognition results of the same object in different images into a single result. The second task is stopping the recognition process – since image capture may not be naturally limited, at time $D(t)$, the task arises of deciding whether the image capture process should be terminated and the accumulated result up to that point should be accepted as final.

As the performance measure of the system at the stop moment $t = t_{fin}$, it is proposed to consider a linear combination:

$$\alpha l\left(Y_{t_{fin}}, w(\rho)\right) + \beta \Pi\left(t_{fin}\right), \tag{2.2}$$

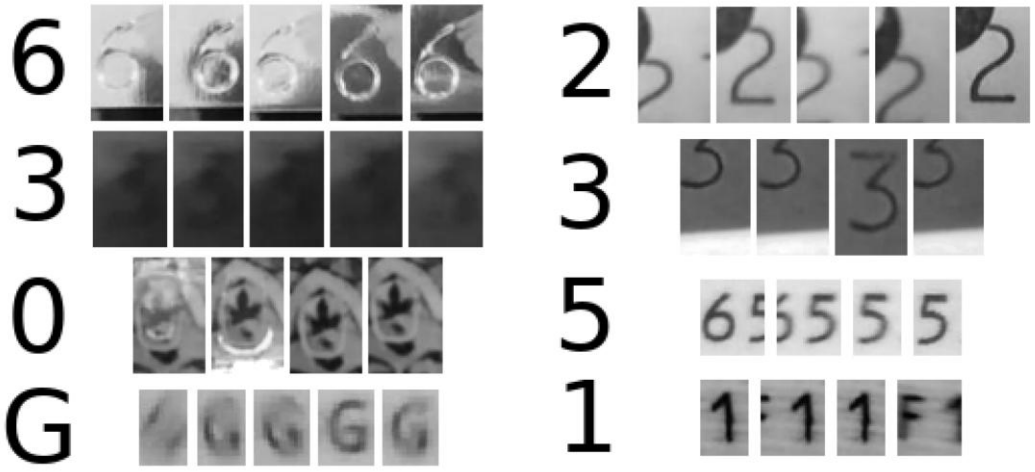
where α, β are constants, $l\left(Y_{t_{fin}}, w(\rho)\right)$ is the distance from the integrated result Y_t to the true value $w(\rho)$, characterizing the quality of the result, and $\Pi(t)$ is a penalty function based on time. A special case of the penalty function $\Pi(t)$ is the number of processed images:

$$\Pi(t) = \max\left\{ i \middle| D^i(0) \le t \right\}. \quad (2.3)$$

2.3 The Task of Integrating Object Recognition Results

The primary task of traditional object recognition systems is to maximize recognition accuracy (i.e., to maximize the proportion of "correct" object classifications). The task of integrating object recognition results is to maximize the accuracy of recognizing a set of various images of the same object, given the recognition results of individual images.

Figure 2.7b shows examples of image sequences of the same object subjected to characteristic distortions that can be classified as environmental noise: distortions associated with the optical system of compact digital cameras, aberrations, glare and reflections within the optical system, digital noise, uneven or insufficient scene illumination, image defocus and blurring due to the movement of the optical sensor relative to the medium, glare from external light sources, geometric distortions such as projective image distortions or nonlinear distortions caused by the bending of the paper medium, interference from holographic protective layers, and others [15; 83]. Figure 2.7a also presents examples of image sequences of an object affected by preprocessing defects, in this case, errors in detecting and localizing the object in the input frame, errors in analyzing the structure and localizing text lines, and errors in segmenting text lines into individual characters [56].



a) b)

Figure 2.7 – Examples of image sequences of objects without preprocessing defects but affected by environmental noise (a) and with preprocessing defects that generate the image (b).

To formalize the problem of integration from the perspective of the object recognition system model in a video stream, let us assume that a set of objects $\mathrm{P} = \{\rho_1, \rho_2, \ldots, \rho_J\}$ of cardinality I and a set of video sequences

$$X = \left\{\mathrm{P}_1\left(\rho_{x_1}\right), \mathrm{P}_2\left(\rho_{x_2}\right), \ldots, \mathrm{P}_J\left(\rho_{x_J}\right)\right\} \tag{2.4}$$

of cardinality J are given, where x_j is the index of an object from the set X for each $j \in \{1, 2, \ldots, J\}$, and each video sequence $\mathrm{P}_j\left(\rho_{x_j}\right) = \left\{\mathrm{P}_{j1}\left(\rho_{x_j}\right), \mathrm{P}_{j2}\left(\rho_{x_j}\right), \ldots, \mathrm{P}_{jMj}\left(\rho_{x_j}\right)\right\}$ is a sequence of images of object $\rho_{x_j} \in \mathrm{P}$, which may be affected by environmental noise and preprocessing defects (see Section 2.1). Additionally, a set of classes $K = \{k_1, k_2, \ldots, k_N\}$ and information about the ideal belonging of each object to the corresponding class w are provided: $\mathrm{P} \to K$.

The task of object recognition in a video stream can be formulated as finding a classification function $F: \mathrm{P}^* \to K$, that maximizes recognition accuracy [120]:

$$W_F(X) = \frac{1}{J}\sum_{j=1}\left[F\left(\mathrm{P}_j\left(\rho_{x_j}\right)\right) = w\left(\rho_{x_j}\right)\right] \to \max_F . \quad (2.5)$$

A more specific task of integrating the results of single-object recognition involves an integration function $Y: \left(\square^K\right)^* \to \square^K$ that transforms a sequence of recognition results for individual images into a unified recognition result for video sequences (here, $\square^K$ is the set of all possible mappings from the set of classes K to the set of scores $\square$, i.e., the set of all possible classification outcomes). Since the final recognition result for a video sequence is the class corresponding to the highest score in the recognition output $F(\mathrm{P}) = \arg\max Y\left(\hat{f}\left(\mathrm{P}\right)\right)$ (see Section 2.2), the formulation of the integration task is based on (2.5) and takes the form of

$$W_F(X) = \frac{1}{J}\sum_{j=1}\left[\arg\max Y\left(\hat{f}\left(\mathrm{P}_j\left(\rho_{x_j}\right)\right)\right) = w\left(\rho_{x_j}\right)\right] \to \max_Y . \quad (2.6)$$

In the ideal case, the classification function F or the result integration function Y should have the ability to filter out outliers that appear in the input data stream due to environmental noise or preprocessing defects, and should also be capable of filtering classifier noise, mitigating random internal errors.

It is easy to notice that the approach to integration as the task of constructing a classification function F can be reduced to the task of building a result integration function Y by applying the existing method of classifying individual object images. Alternative approaches include, for example, super-resolution techniques [118], which perform pixel matching of a set of input images $\mathrm{P}(\rho)$ and create a unified (ideal) image of the object, which is then classified. However, it should be noted that due to the specific characteristics of the most accurate existing method for image classification (convolutional neural networks), namely, their instability to random pixel distortions, this work will focus on methods for constructing the integration function Y for the results of single-image recognition.

In the literature, the task of combining single-object classification results is usually discussed in the context of methods for obtaining more accurate classification by combining the results of several different classifiers [89; 121; 122]. Depending on the model used for the classification result of an object and the interpretation of the classifier's scores, various combination methods are considered.

The task of combining object classification results can be considered as a collective decision-making problem. Let us introduce the concept of a predictor of classifier result reliability as a real-valued function $r\left(\mathrm{P}(\rho),\hat{f}\right)$, reflecting the degree of confidence that the classification result of image $\mathrm{P}(\rho)$ by the function $\hat{f}$ will be correct. It makes sense to use computable image characteristics as predictors, which are known to influence classification accuracy [123], such as blur and focus level assessments [115], noise level estimation, digitization artifacts [124], and so on. (Such predictors can be considered a priori, as they directly rely on the characteristics of the input images). Another class of predictors is conditioned by the values of classification scores (posterior predictors), which are related to the concept of the recognition result reliability score [7; 125]. An example of a widely used posterior predictor of reliability is the value of the first (maximum) alternative score [6]:

$$r\left(\mathrm{P}(\rho),\text{资}\right)=\max f\left(\mathrm{P}(\rho)\right). \tag{2.7}$$

Let a certain reliability predictor be given. Then, the task of integrating the recognition results of a sequence of images $\mathrm{P}(\rho)=\left\{\mathrm{P}_1(\rho),\mathrm{P}_2(\rho),\ldots,\mathrm{P}_M(\rho)\right\}$ of cardinality M can be considered as a collective decision-making problem with experts, whose competence levels are functions of the values of the reliability predictor. It is worth noting that the experts' competence levels in this model reflect the input data, as the characteristics of individual observations (i.e.,

individual images $P_1(\rho),\ldots,P_M(\rho)$) are necessary for evaluating the significance of the experts.

An important issue within this task is the question of whether it is more appropriate to use voting from multiple experts instead of relying on the opinion of the most competent expert [126; 127]. When addressing the specific problem, this question is formulated as follows: under which input data models in the task of combining recognition results should one choose a particular strategy for combination?

To answer this question, an experimental study is proposed. Four datasets were prepared, the characteristics of which are provided in Table 2. The MRZ-MSEGM and MRZ-CLEAN datasets contain video sequences of recognition results for characters in the machine-readable zone of international documents [83]. The ICN-MSEGM and ICN-CLEAN datasets contain video sequences of recognition results for the "Number" field characters of payment bank cards, captured using indent printing. The images of characters in the considered test datasets are subject to a wide range of distortions: uneven or insufficient lighting, digital noise, defocusing and "blurring" due to the optical sensor's movement relative to the medium, glare from an external light source, and interference created by the holographic security layer of the document, among others. The recognition result for each individual character image was obtained using convolutional neural networks, separately trained for the machine-readable zone characters and for the "Number" field characters of payment bank cards, on separate training image sets using data augmentation techniques [128]. The MRZ-MSEGM and ICN-MSEGM datasets contain errors caused by the incorrect or insufficiently accurate operation of document localization algorithms and text line segmentation algorithms. The MRZ-CLEAN and ICN-CLEAN datasets are subsets of the corresponding MRZ-MSEGM and ICN-MSEGM datasets, which do not contain such errors. Thus, in the MRZ-CLEAN and ICN-CLEAN datasets, each video sequence contains images of the exact same character, without any segmentation defects.

Table 2 – Characteristics of the test datasets MRZ-MSEGM, MRZ-CLEAN, ICN-MSEGM, and ICN-CLEAN.

Dataset description	MRZ-CLEAN	MRZ-MSEGM
Cardinality of the set of classes K	37	
Total number of symbol images	631530	637874
ecognition accuracy of individual images, %	96.8994	96.7357
Number of video sequences	7508	7581
Minimum length $\mathrm{P}(\rho)$	3	
Maximun length $\mathrm{P}(\rho)$	223	
Mean length $\mathrm{P}(\rho)$	21	
Dataset description	ICN-CLEAN	ICN-MSEGM
Cardinality of the set of classes K	10	
Total number of symbol images	29166	31580
ecognition accuracy of individual images, %	96.8936	90.9816
Number of video sequences	1748	1898
Minimum length $\mathrm{P}(\rho)$	3	
Maximun length $\mathrm{P}(\rho)$	25	
Mean length $\mathrm{P}(\rho)$	12	

The comparison of basic classifier combination strategies presented in the overview chapter was conducted on the provided test datasets: the product rule (1.2), sum rule (1.3), minimum rule (1.4), maximum rule (1.5), and median rule (1.6). The recognition accuracy of a video sequence of characters is the relative proportion of video sequences for which the ideal answer matches the class that received the highest score according to a given combination rule. Additionally, a comparison was made between the basic combination rules and the voting method (1.1), generalized as follows:

$$Poll(\sigma)\left(\hat{f}\left(\mathrm{P}(\rho)\right)\right)(k) = \\ = \sigma\frac{1}{M}\sum_{i=1}^{M}1_{\mathrm{P}_k(\rho)}\left(\mathrm{P}_i(\rho)\right)+(1-\sigma)\max_{i=1}^{M}\left(1_{\mathrm{P}_k(\rho)}\left(\mathrm{P}_i(\rho)\right)r\left(\mathrm{P}_i(\rho),\hat{f}\right)\right), \tag{2.8}$$

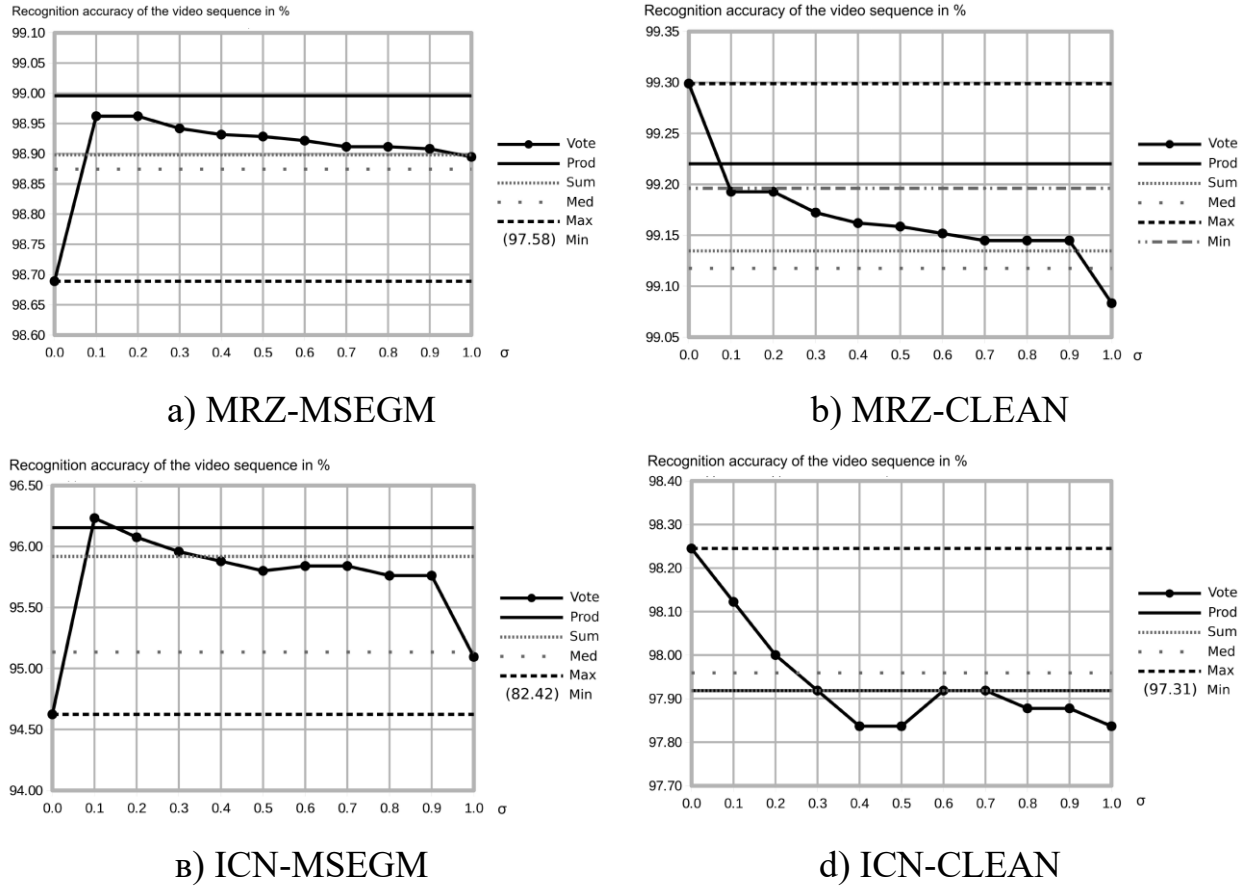
where $\mathrm{P}_k(\rho) = \left\{ \mathrm{P}(\rho) \in \mathrm{P}(\rho) \middle| f(\mathrm{P}(\rho)) = k \right\}$ is a subset of video sequence elements for which the classifier choice corresponds to class k, $1_{\mathrm{P}_k(\rho)}(\mathrm{P}(\rho))$ is the indicator function for the membership of image $\mathrm{P}(\rho)$ in subset $\mathrm{P}_k(\rho)$, and $r(\mathrm{P}_i(\rho), \hat{f})$ is the credibility predictor. The posterior predictor "first alternative rule" (2.7) was used as the credibility predictor.

Figure 2.8 shows the comparative values of recognition accuracy for video sequences using combination rules (1.2), (1.3), (1.4), (1.5), (1.6), and (1.1) on the test datasets MRZ-MSEGM, MRZ-CLEAN, ICN-MSEGM, and ICN-CLEAN. The horizontal axis of the graphs corresponds to the values of parameter a for the combination rule (1.1). The recognition accuracy using the other combination rules is represented by horizontal lines.

Figure 2.8 demonstrates a significant difference in the optimal choice of combination strategy depending on the input data model: on test sets where errors in character localization and segmentation occur, higher recognition accuracy for video sequences is achieved by the product rule (1.2), voting (1.1), and the sum rule (1.3) (Fig. 2.8a, 2.8c). However, on test sets where such errors were excluded (Fig. 2.8b, 2.8d), higher recognition accuracy is provided by the maximum rule (1.5). In other words, when considering this task as a collective decision-making problem, in the case of a stricter input data model (with no errors in character localization and segmentation), it is more advantageous to trust a single competent expert rather than the collective opinion of several experts.

In the presence of errors in character localization and segmentation, the stability of the credibility predictors decreases, which, in turn, increases the difference between the competency ratings of the experts (constructed based on the values of the predictors) and the actual competency values of the experts (which correspond to the posterior probabilities of making the correct decision). In such cases, selecting the expert with the highest competency level is more often incorrect, and thus the difference between the actual competency level of the chosen expert and the competency levels of the other experts decreases. Therefore,

the optimal choice, in terms of the reliable predictor of video sequence credibility, is absent when segmentation and localization errors are present, which aligns with a broader interpretation of collective decision-making theory [126; 129]. According to this theory, a violation of the first part of Condorcet's assertion (that with an increasing number of experts, the probability of collective correct decision-making increases if each expert has a higher probability of making a correct individual decision than a wrong one) occurs when the difference between the competency levels of the most competent expert and the others increases.



a) MRZ-MSEGM

b) MRZ-CLEAN

в) ICN-MSEGM

d) ICN-CLEAN

Figure 2.8 – Comparison of recognition accuracy for video sequences of characters using basic combination strategies.

The results of the experiment allow the conclusion that when building an object recognition system in a video stream, the choice of a combination strategy for the results should be guided not only by the model of the object recognition

results but also by the noise model of the input data. In this case, with a fixed noise model for the input data, the results of studies aimed at combining different classifiers to maximize the recognition accuracy of a single object can be used for integrating classification results of individual objects. However, direct application of the considered combination rules is not possible if the object recognition result model is more complex than a simple classification result (2.1). An example of such an object is a text string, for which classification is performed independently for each character. The task of integrating such objects will be addressed in Chapter 3.

2.4 Stopping Criterion

The model of the object recognition system in a video stream (see Fig. 2.6) does not assume limitations on the number of input images. Since the main goal of the object recognition system is to automate input, an important parameter is the absolute time (i.e., the time from the operator's perspective) required to obtain the final recognition result. Unlike the process of taking a photograph, a video stream is naturally unlimited in time. Therefore, the stopping criterion arises, which involves making a decision about whether the newly obtained result $\hat{f}\left(\mathrm{P}_0(\rho),\mathrm{P}_{D^1(0)}(\rho),\mathrm{P}_{D^2(0)}(\rho),...,\mathrm{P}_t(\rho)\right)$ at time $D(t)$ can be considered final, and whether the image capture cycle can be terminated. In the case of recognizing complex objects that consist of multiple independently recognized components, the decision to stop the recognition of individual components affects the time Δ needed to recognize the composite object, and thus the amount of information processed within the overall system. Hence, the stopping criterion (closely related to the integration task) is an important aspect of object recognition in a video stream, especially in the context of interaction with other subsystems, where the object of recognition is a composite entity, such as a text field or the document as a whole.

In its simplest form, the stopping rule can be represented as a predicate operating on the video sequence $\mathrm{P}\colon \mathrm{P}^{*} \to \{0,1\}$. The truth of the predicate leads to the termination of the image capture and recognition process:

$$\mathrm{P}\left(\left\{\mathrm{P}_1(\rho),\mathrm{P}_2(\rho),\ldots,\mathrm{P}_m(\rho)\right\}\right)=\begin{cases}1\forall\, \textit{decision to stop},\\ 0\forall \textit{if decision tocontinue.}\end{cases} \tag{2.9}$$

Let $\mathrm{P}(\rho)=\left\{\mathrm{P}_1(\rho),\mathrm{P}_2(\rho),\ldots,\mathrm{P}_M(\rho)\right\}$ be the sequence of images of an object $\rho\in\mathrm{P}$, and $\mathrm{P}^{(m)}(\rho)=\left\{\mathrm{P}_1(\rho),\mathrm{P}_2(\rho),\ldots,\mathrm{P}_m(\rho)\right\}\subseteq\mathrm{P}(\rho)$ be the prefix of this sequence with length $m\le M$. Let $A_{\mathrm{P}}\left(\mathrm{P}(\rho)\right)$ denote the number of images that will be processed by the recognition system until the stopping rule (2.9) is triggered:

$$A_{\mathrm{P}}\left(\mathrm{P}(\rho)\right)=\min\left[M,\min\left\{\left|\mathrm{P}^{(m)}(\rho)\right\| m\in\{1,2,\ldots,M\}\wedge\mathrm{P}\left(\mathrm{P}^{(m)}(\rho)\right)\right\}\right]. \tag{2.10}$$

Taking the stopping rule into account, only images from the subsequence $\mathrm{P}(\rho)$ are fed into the recognition system during the processing of the video sequence $\mathrm{P}^{(\mathrm{P})}(\rho)=\mathrm{P}^{\left(A_{\mathrm{P}}(\mathrm{P}(\rho))\right)}(\rho)$, and the original set of video sequences (2.4) takes the form $X^{(\mathrm{P})}=\left\{\mathrm{P}_1^{(\mathrm{P})}\left(\rho_{x_1}\right),\mathrm{P}_2^{(\mathrm{P})}\left(\rho_{x_2}\right),\ldots,\mathrm{P}_J^{(\mathrm{P})}\left(\rho_{x_J}\right)\right\}$.

To formalize the stopping problem, we use a general model of interaction between the recognition system and the user, which is employed in tasks for determining the reliability of object recognition results [6; 81], and for assessing the system's performance using a functional described in economic terms. Let Z_c be the cost of inputting a correct object recognition result, Z_{ic} be the cost of inputting an incorrect result, and Z_ρ be the cost of recognizing a single image of the object. Then, the efficiency function of the stopping rule can be expressed as the average cost of the system's operation:

$$Z_{F,\mathrm{P}}(X) = Z_c W_F\left(X^{(\mathrm{P})}\right) + Z_{ic}\left(1 - W_F\left(X^{(\mathrm{P})}\right)\right) + Z_\rho \frac{1}{J}\left(\sum_{j=1} A_j\left(\mathrm{P}(\rho)\right)\right), \qquad (2.11)$$

where $W_F\left(X^{(\mathrm{P})}\right)$ is the recognition accuracy of video sequences considering the stopping rule (2.9), calculated according to (2.5) (similarly, in the case of integrating the recognition results of individual objects, the accuracy is calculated according to (2.6)).

By simplifying expression (2.11) and considering the constancy of Z_{ic}, we arrive at the general formulation of the stopping problem as the task of finding a stopping rule that optimizes the efficiency functional:

$$Z_{F,\mathrm{P}}(X) = W_F\left(X^{(\mathrm{P})}\right)\left(Z_c - Z_{ic}\right) + Z_\rho \frac{1}{J}\left(\sum_{j=1} A_j\left(\mathrm{P}(\rho)\right)\right). \qquad (2.12)$$

A similar efficiency functional is constructed taking into account the accuracy functional (2.6) within the framework of the task of integrating the recognition results of individual objects.

In the context of object recognition in a video stream, the stopping problem is relatively new and underexplored. This problem, as well as the method for solving it, will be discussed in detail in Chapter 4.

2.5 Conclusions of the Chapter

This chapter demonstrated the properties of the object recognition problem in a video stream. Various approaches to formalizing the recognition system in a video stream were presented, and a model of a dynamic system was constructed, featuring a module for combining frame-by-frame recognition results and a stopping module. The properties of the dynamic object recognition system in a video stream were highlighted, and problem statements for integrating recognition results from multiple observations of the same object and for the stopping problem within such systems were proposed.

The task of integrating (combining) recognition results from multiple observations of the same object was considered as a collective decision-making problem. It was shown that the model of input data can affect the choice of the optimal combination strategy. According to the experimental study, in test datasets where image preprocessing defects occur, combination rules such as voting and the product rule are effective. Meanwhile, in test datasets where such errors were excluded, the maximum rule provided higher recognition accuracy for the video sequence. In terms of collective decision-making, when a stricter model of input data is used (i.e., when recognizing a set of images with minimal preprocessing defects), it is more beneficial to trust the single most competent expert rather than the collective opinion of several experts.

The stopping problem in object recognition processes in video streams was described, arising due to the lack of a natural limitation on the number of observations obtained over time. This problem is novel in the context of optical object recognition systems.

3 IMPLEMENTATION OF STRING OBJECT RECOGNITION RESULTS IN REAL-TIME VIDEO STREAMS

3.1 Introduction

The recognition of objects such as text paragraphs, text lines, document fields, etc., involves a set of challenges, especially when the image source is a mobile device camera. Under such conditions, the images are often characterized by distortions such as defocusing, blurring, glare on reflective surfaces, insufficient resolution for achieving adequate accuracy of character recognition algorithms, and more [15; 130; 131]. Figure 3.1 illustrates an example of glare on a document and its impact on the images of text fields extracted from consecutive frames of a video stream.

Figure 3.1 – A frame fragment with glare on the reflective surface of a document. Images from the MIDV-500 dataset [88] (clip HA44).

One of the advantages of using video streams for object recognition is the ability to process multiple frames in real time, i.e., recognizing the same object repeatedly, thereby increasing the final recognition accuracy. It should also be noted that selecting a single best result may not always be an acceptable strategy, as the video stream may lack a frame with the object fully visible. Therefore, there is a need to explore methods for combining multiple recognition results.

The objectives of this chapter are to construct a model of the recognition result for a string object, accounting for alternative classifications of individual objects, and based on this model, to develop an algorithm for integrating the recognition results of string objects. Section 3.2 describes the model of recognition results for single and string objects, which is used for constructing the algorithm. Section 3.3 presents the problem statement for integrating string object recognition results. Section 3.4 introduces the proposed algorithm, and Section 3.5 provides its experimental evaluation.

3.2 Model of Recognition Result for a String Object

Consider the model for the recognition result of a single object. Let the image P of an object k be classified into one of N classes from a set $K=\{k_1,k_2,\ldots,k_N\}$ using a classification module f. In the classical formulation, the result of classification is one of the classes $f(\mathrm{P})=k_f$, where $k_f \in K$, and the task of recognizing a single object is to maximize the posterior probability that the class k_f matches the true value k. In a more general formulation, the classification module $\hat{f}$ assigns a set of pairs $\hat{f}(I)=\{(k_1,\phi_1),(k_2,\phi_2),\ldots,(k_N,\phi_N)\}$ to the input image, where ϕ_i is the membership score of the object to class k_i. The final recognition result is the class corresponding to the maximum membership score:

$$f(\mathrm{P})=\arg\max\left\{\text{资}(\mathrm{P})\right\}\in\left\{k_f\left|\left((k_f,\phi_f)\in f(\mathrm{P})\right)\wedge\left(\phi_f=\max_{(k,\phi)\in\hat{f}(\mathrm{P})}\phi\right)\right.\right\}. \qquad (3.1)$$

In the case where there are multiple pairs $(k_{f1},\phi_f),(k_{f2},\phi_f),\ldots$ with the same maximum membership score, one of the classes is selected as the answer according to the adopted convention (for example, the class with the smallest index in the set K). The recognition result model for a single object (3.1) is a variant of the Algorithm for Calculating Scores (ACS) [132] and is also the most widely used

model in optical image recognition methods using convolutional neural networks [64].

To define the recognition result of a string object, it is necessary to introduce the concept of an empty class η, representing the absence of a single object. The extended result of classifying a single object is considered to be a mapping $\alpha : K \cup \{\eta\} \to [0,1]$ from the set of classes, augmented with the label of the empty class η, to the set of membership scores. Each membership score is a real number between 0 and 1, and the sum of the membership scores is equal to one. Thus, the set of all possible recognition results for a single object K is defined as:

$$\hat{K} \overset{def}{=} \left\{ \alpha \in [0,1]^{K \cup \{\eta\}} \,\middle|\, \sum_{k \in K \cup \{\eta\}} \alpha(k) = 1 \right\}. \tag{3.2}$$

On the set of all possible recognition results for a single object K, a metric can be defined as follows:

$$l_{\hat{K}}(\alpha, \beta) \overset{def}{=} \frac{1}{2} \sum_{k \in K \cup \{\eta\}} |\alpha(k) - \beta(k)|, \ \forall \alpha, \beta \in \hat{K}. \tag{3.3}$$

It is easy to verify that the function $l_{\hat{K}}(\alpha, \beta)$ possesses the properties of a valid metric:

1. $l_{\hat{K}}(\alpha, \beta) = 0 \Leftrightarrow \forall k \in K \cup \{\eta\} : \ \alpha(k) = \beta(k) \Leftrightarrow \alpha = \beta$, thus the identity axiom is satisfied.

2. $\forall k \in K \cup \{\eta\} : |\alpha(k) - \beta(k)| = |\beta(k) - \alpha(k)| \Rightarrow l_{\hat{K}}(\alpha, \beta) = l_{K}(\beta, \alpha)$, thus the symmetry axiom is satisfied.

3. $\forall i, j \in \square : |i + j| \le |i| + |j| \Rightarrow \forall k \in K \cup \{\eta\} :$ $|\alpha(k) - \gamma(k)| \le |\alpha(k) - \gamma(k)| + |\gamma(k) - \beta(k)| \Rightarrow l_{\hat{K}}(\alpha, \beta) \le l_{K}(\alpha, \gamma) + l_{\hat{K}}(\gamma, \beta)$, thus the triangle inequality is also satisfied.

It is worth noting that the metric $l_{\hat{K}}(\alpha, \beta)$ corresponds to the Manhattan metric in the vector space over the ordered set $K \cup \{\eta\}$. Since for α and β, the

sum of the values across all $k \in K \cup \{\eta\}$ equals one, the set of values for the function $l_{\hat{K}}(\alpha,\beta)$ forms a segment [0, 1]. We denote the "empty result" as:

$$\eta \overset{def}{=} \{(\eta,1),(k_1,0),(k_2,0),\ldots,(k_N,0)\}. \tag{3.4}$$

The result P of recognition of a string object will be called a string over the set $K \setminus \{\hat{\eta}\}$, i.e., an element $\mathrm{P} \in P$, where $P \overset{def}{=} \left(\hat{K} \setminus \{\hat{\eta}\}\right)^*$. The string P represents a sequence of recognition results of individual objects $\mathrm{P} = \rho_1\rho_2\ldots\rho_m$, where $\rho_i \in \hat{K} \setminus \{\hat{\eta}\}$, and the length of the string $|\mathrm{P}| = m$ is called the number of elements in this sequence. The notation $\mathrm{P}_{i\ldots j}$ refers to a substring of the string P, including elements $\rho_i\rho_{i+1}\ldots\rho_{j-1}\rho_j$ for $1 \le i \le j \le m$. When $i > j$, the substring $\mathrm{P}_{i\ldots j}$ corresponds to an empty string $\hat{\eta}$ of zero length.

Let us introduce the concept of an elementary edit operation D as a pair $(\alpha,\beta) \neq (\eta,\eta)$, where $\alpha,\beta \in \hat{K}$. An edit operation $D = (\alpha,\beta)$, applied to the string P, corresponds to:

1. If $\beta \neq \eta$: replacing element $\rho_i = \alpha$ in the string P with element β, if $\beta \neq \eta$;

2. If $\beta = \hat{\eta}$: deleting element $\rho_i = \alpha$ from the string P;

3. If $\alpha = \hat{\eta}$: inserting element β into the string P.

Let us consider two arbitrary strings $\mathrm{P},\Theta \in P$ of finite length. An editorial prescription is defined as a sequence of elementary editorial changes $D_{\mathrm{P},\Theta} = D_1D_2\ldots D_L$ that transforms a string P into a string Θ. The weight of an editorial prescription is considered to be the sum of the distances (in terms of the metric $l_{\hat{K}}$) between pairs of objects involved in the elementary editorial changes $D_i = (\alpha_i,\beta_i)$ of the prescription $D_{\mathrm{P},\Theta}$:

$$\omega\left(D_{\mathrm{P},\Theta}\right)\overset{def}{=}\sum_{i=1}^{L} l_{\hat{K}}\left(\alpha_i,\beta_i\right). \tag{3.5}$$

A metric on the set of recognition results for string objects P is defined as the minimal weight of an editorial prescription that transforms one string into another:

$$l_P\left(\mathrm{P},\Theta\right)=\min\left\{\omega\left(D_{\mathrm{P},\Theta}\right)\right\}. \tag{3.6}$$

The metric (3.6) can be considered as one of the implementations of the Generalized Levenshtein Distance [133] and possesses the properties of a true metric, provided that the metric (3.3) also satisfies these properties [134].

To calculate the distance between two recognition results of string objects $l_P\left(\mathrm{P},\Theta\right)$, the following recursive scheme can be used. Let $a\left(i,j\right)\overset{def}{=}l_P\left(\mathrm{P}_{1\ldots i},\Theta_{1\ldots j}\right)$ represent the distance between the prefixes of strings P and Θ, which have lengths i and j, respectively. Then

$$\begin{gathered}
a\left(0,0\right)=0,\\
a\left(i,0\right)=\sum_{h=1}^{i} l_{\hat{K}}\left(\rho_h,\hat{\eta}\right),\\
a\left(0,j\right)=\sum_{h=1}^{j} l_{\hat{K}}\left(\hat{\eta},\theta_h\right),\\
a\left(i,j\right)=\min\left\{\begin{array}{l} l_{\hat{K}}\left(\rho_i,\hat{\eta}\right)+a\left(i-1,j\right),\\ l_{\hat{K}}\left(\hat{\eta},\theta_j\right)+a\left(i,j-1\right),\\ l_{\hat{K}}\left(\rho_i,\theta_j\right)+a\left(i-1,j-1\right)\end{array}\right\},
\end{gathered} \tag{3.7}$$

and the sought value of the metric $l_P\left(\mathrm{P},\Theta\right)$ corresponds to the value $a\left(\left|\mathrm{P}\right|,\left|\Theta\right|\right)$.

It is worth noting that the maximum possible value of the metric $l_P\left(\mathrm{P},\Theta\right)$ is the maximum length of the strings P and Θ (when using (3.3) as the metric on the

set of recognition results for single objects). Since l_P is a special case of the Generalized Levenshtein Distance, it is possible to construct a normalized version of this metric while preserving the axioms of identity, symmetry, and the triangle inequality [134]:

$$\tilde{l}_P(\mathrm{P},\Theta)\,def\,\frac{2l_P(\mathrm{P},\Theta)}{\sigma(|\mathrm{P}|+|\Theta|)+l_P(\mathrm{P},\Theta)}, \tag{3.8}$$

where σ is the maximum possible weight of an elementary insertion or deletion. For the case of the metric (3.3): $\sigma=\max\left\{l_K(\alpha,\eta),l_K(\eta,\beta);\alpha,\beta\in K\right\}=1$.

In addition to the Generalized Levenshtein Distance, there are other approaches for comparing string objects, such as the Dynamic Time Warping (DTW) algorithm [133; 135]. In its classical formulation, however, the DTW algorithm assumes the alignment of boundary components of string objects, does not account for penalties for insertion and deletion of components, and does not possess metric properties (it does not guarantee the triangle inequality).

3.3 Task of Integrating String Object Recognition Results

Let us consider the task of recognizing a string object in a video sequence. The input to the system is a sequence of images $\mathrm{P}_1,\mathrm{P}_2,\ldots,\mathrm{P}_M$ of the string object $w\in K^*$. Using the module $\hat{F}$ for recognizing the string object in a single image, each image is assigned a recognition result $\hat{F}(\mathrm{P}_i)\in P$. Within the framework of the considered model, we assume that in the initial recognition result of the string object, the membership estimates corresponding to the empty class η are equal to zero:

$$\begin{gathered}\hat{F}(\mathrm{P}_i)=P_i,\ P_i\in P,\\ \mathrm{P}_i=\rho_1^i\rho_2^i\ldots\rho_{m_i}^i,\\ \rho_j^i(\eta)=0\ \ \forall j\in\{1,\ldots,m_i\}.\end{gathered} \tag{3.9}$$

The task consists of combining the results $\mathrm{P}_1,\mathrm{P}_2,\ldots,\mathrm{P}_M$ with certain weights $\omega_1,\omega_2,\ldots,\omega_M$ into a unified result $\mathrm{P}\in P$, minimizing the distance to the true value w according to a given metric. Since $\mathrm{P}\in P$ is a string over the set $\hat{K}\setminus\{\eta\}$, and w is a string over the set of classes K, additional conversion is required to determine the distance between them. The most natural approach is to transform the true value w into a string $\hat{w}\in P$:

$$\begin{gathered} w = w_1 w_2 \ldots w_{m_w},\ w_j \in K, \\ \hat{w} = w_1 \hat{w}_2 \ldots w_{m_w},\ \hat{w}_j \in \hat{K}\setminus\{\hat{\eta}\}, \\ \hat{w}_j \overset{def}{=} \{(\eta,0),(k_1,0),(k_2,0),\ldots,(w_j,1),\ldots,(k_N,0)\}, \end{gathered} \tag{3.10}$$

and use the distance (3.6) or its normalized variant (3.8) as the distance from the integrated result P to the true value w.

However, from a practical application perspective, it is also important to obtain the final recognition result of the string object (analogous to the final result (3.1) for a single object). The following two-step procedure can be used to obtain the final result:

1. At the first step, each component $\rho_j \in \hat{K}\setminus\{\hat{\eta}\}$ of the integrated result $\mathrm{P}=\rho_1\rho_2\ldots\rho_{m_\mathrm{P}}$ is assigned either the class $k_{\rho_j}\in K$ with the maximum membership estimate $\rho_j\left(k_{\rho_j}\right)$, or the empty class η, if its estimate $\rho_j(\eta)$ exceeds a certain threshold λ:

$$\bar{\rho}_j = \begin{cases} \arg\max \rho_j(k) \forall \rho_j(\eta) < \lambda, \\ \eta \forall \rho_j(\eta) \geq \lambda. \end{cases} \tag{3.11}$$

2. At the second step, all components $\bar{\rho}_j = \eta$ are removed from the resulting string $\bar{\rho}_1\bar{\rho}_2\ldots\bar{\rho}_{m_\mathrm{P}}$. The resulting string $\bar{\mathrm{P}}_\lambda \in K^*$ can be used as the final recognition result of the string object.

As the distance from the integrated result P to the true value w, the Levenshtein distance $levenshtein\left(\overline{\mathrm{P}}_{\lambda}, w\right)$ [133] or its normalized variant 134]:

$$l_L\left(\overline{\mathrm{P}}_{\lambda}, w\right) = \frac{2levenshtein\left(\overline{\mathrm{P}}_{\lambda}, w\right)}{\left|\overline{\mathrm{P}}_{\lambda}\right| + |w| + levenshtein\left(\overline{\mathrm{P}}_{\lambda}, w\right)}, \tag{3.12}$$

can now be used.

The task of integrating string objects was examined in [87] in the context of speech recognition. Instead of integrating the recognition results of multiple images $\mathrm{P}_1, \mathrm{P}_2, \ldots, \mathrm{P}_M$ using a single recognition module $\hat{F}$, as in this work, [87] focuses on integrating the recognition results of a single "image" P processed by different recognition systems $F_1, F_2, \ldots, F_M$. These problem formulations can be considered similar, differing mainly in the noise model: the integration of string object recognition results in a video sequence aims to filter out the noise component in the original images $\mathrm{P}_1, \mathrm{P}_2, \ldots, \mathrm{P}_M$ (caused by input data inaccuracies, preprocessing errors, etc.) and its impact on the recognition module $\hat{F}$, whereas the integration of results from different recognition modules is aimed at filtering noise introduced by the recognition modules $F_1, F_2, \ldots, F_M$ themselves.

In addition to speech recognition, the approach presented in [87] has also been applied to combining multiple classifiers in optical recognition tasks for printed [85] and handwritten [136] texts.

The approach described in [87], known as Recognizer Output Voting Error Reduction (ROVER), employs a two-module scheme, as shown in Figure 3.2.

At the first stage, the alignment module converts all input string objects into strings of equal length by optimally inserting the empty class η.

At the second stage, the voting module selects a class for each component of the resulting string based on a linear combination of occurrence frequency and confidence scores generated by the recognition module.

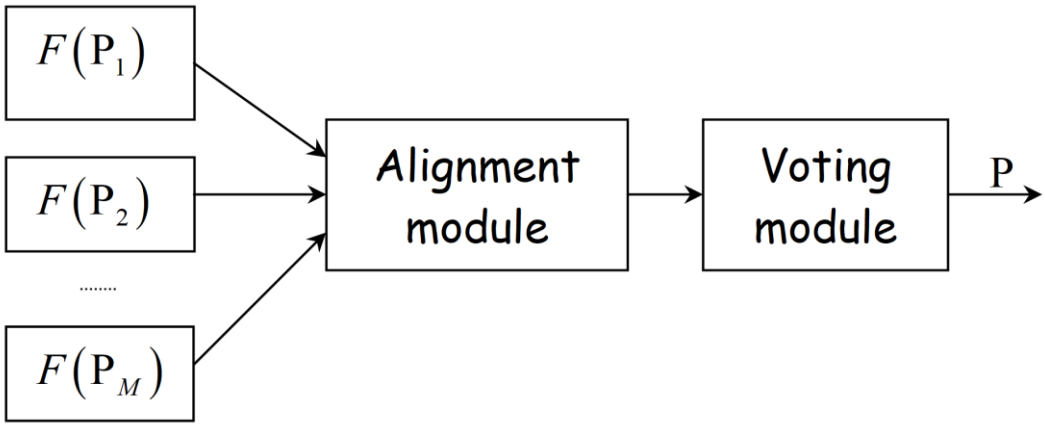


Figure 3.2 – Two-module scheme of the ROVER approach.

The model of a single string recognition result in the ROVER approach [87] is represented as a pair consisting of a string over the set of recognition classes for individual objects and the confidence score of the recognition module, i.e., an object from the set $K^* \times$ □ .

To construct an algorithm for integrating string object recognition results with an extended model of a single recognition result, let us consider the problem formulation of string alignment of the form (3.9).

Let $\mathrm{P}_1, \mathrm{P}_2, \ldots, P_M$ be the set of M strings, where $\mathrm{P}_i \in P$ and $|\mathrm{P}_i| = m_i > 0$ are given:

$$\begin{gathered} \mathrm{P}_1 = \rho_1^1 \rho_2^1 \ldots \rho_{m_1}^1, \\ \mathrm{P}_2 = \rho_1^2 \rho_2^2 \ldots \rho_{m_2}^2, \\ \ldots \\ \mathrm{P}_M = \rho_1^M \rho_2^M \ldots \rho_{m_M}^M. \end{gathered} \tag{3.13}$$

The alignment of a given set of strings will be understood as an *align* function: $\{1,\ldots,M\} \times \left\{1,\ldots,\max_{i=1}^{M} m_i\right\} \to \left\{1,\ldots,\sum_{i=1}^{M} m_i\right\}$. The function $align(i,j)$ specifies the index of the output "integrated" string component, into which the components ρ_j^i contribute. For each input string, the values of the *align* function

for individual string components are different and maintain their order: $\forall i \in \{1,\ldots,M\},\ \forall j \in \{1,\ldots,m_i - 1\}:\ align(i,j) < align(i,j+1)$.

We also introduce a *match* function: $\{1,\ldots,M\} \times \left\{1,\ldots,\sum_{i=1}^{M} m_i\right\} \to K$, defined as follows:

$$match(i,c) \overset{def}{=} \begin{cases} \rho_j^i \forall align(i,j) = c, \\ \eta \forall \exists j : align(i,j) = c. \end{cases} \tag{3.14}$$

The alignment problem consists of finding the *align* function such that the penalty functional

$$\sum_{c} \sum_{i_1 < i_2} l_{\hat{K}} \left(match(i_1,c), \left(match(i_2,c), \right) \right) \to \min \tag{3.15}$$

is minimized, reflecting the total pairwise distance between the recognition results of individual objects that contribute to the same components of the integrated result.

To generalize the voting module (see Fig. 3.2), which selects a class for each component of the resulting string, we introduce a family of functions for combining the recognition results of individual objects $y^{(M)}$:

$$y^{(M)} : K^{M} \times \left(\square_{0}^{+} \right)^{M} \to K \setminus \{\hat{\eta}\}. \tag{3.16}$$

The function $y^{(M)}$ takes as input M recognition results of individual objects $\alpha_1, \alpha_2, \ldots, \alpha_M$ such that $\exists i : \alpha_i \neq \hat{\eta}$, and a set of associated non-negative weights $\omega_1, \omega_2, \ldots, \omega_M$, reflecting the significance of the result, such that $\sum_{i=1}^{M} \omega_i > 0$.

Then the function $Y^{(M)}$ for integrating the recognition results of string objects takes the form

$$Y^{(M)}\left(\mathrm{P}_1,\mathrm{P}_2,\ldots,\mathrm{P}_m,\omega_1,\omega_2,\ldots,\omega_M\right)=y_1^{(M)},y_2^{(M)},\ldots,y_{m_Y}^{(M)}. \quad (3.17)$$

where $m_Y=\max align(i,j)$, and each component of the resulting string is calculated using the combination function (3.16) and in accordance with the alignment result (3.14):

$$y_j^{(M)}=y^{(M)}\left(match(1,j),match(2,j),\ldots,match(M,j),\omega_1,\omega_2,\ldots,\omega_M\right). \quad (3.18)$$

In the general case, the exact solution to the problem (3.15) involves calculating a dynamic programming scheme (analogous to the scheme for calculating the Generalized Levenshtein Distance (3.7) with a computational cost that exponentially depends on the number of input strings M (since the results of the string alignment subproblems $\mathrm{P}_{11\ldots i_1},\mathrm{P}_{21\ldots i_2},\ldots,\mathrm{P}_{M1\ldots i_M}$ must be used for all tuples $(i_1,i_2,\ldots,i_M)\in\{1,\ldots,m_1\}\times\{1,\ldots,m_2\}\times\ldots\times\{1,\ldots,m_M\}$. Heuristic algorithms for shortest path search, such as A^*-search [137], can also be applied when calculating this scheme. The next section will present an algorithm for integrating the recognition results of string objects, with the alignment functional approximated using the method employed in the ROVER approach [87].

3.4 Algorithm for Integrating the Recognition Results of a String Object

When calculating the integrated result of the recognition of a string object, a set of intermediate integrated results $Y^{(1)}\left(\mathrm{P}_1,\omega_1\right),\ldots,Y^{(i-1)}\left(\mathrm{P}_1,\ldots,\mathrm{P}_{i-1},\omega_1,\ldots,\omega_{i-1}\right)$ is generated, where the result $Y^{(i-1)}$ is used to solve the alignment task at step i. In the first step of the algorithm:

$$Y^{(1)}\left(\mathrm{P}_1,\omega_1\right)=\mathrm{P}_1. \quad (3.19)$$

At each subsequent i-th step of the algorithm, the optimal alignment of the strings P_i and $Y^{(i-1)}\left(\mathrm{P}_1,\ldots,\mathrm{P}_{i-1},\omega_1,\ldots,\omega_{i-1}\right)$ is built using a dynamic programming

scheme, similar to (3.7). Let $\gamma(d,e) \overset{def}{=} l_P\left(\mathrm{P}_{i1\ldots d}, Y^{(i-1)}\left(\mathrm{P}_1,\ldots,\mathrm{P}_{i-1},\omega_1,\ldots,\omega_{i-1}\right)_{1\ldots e}\right)$ and $R_r(d,e)$ are the auxiliary functions for $r \in \{1,2,3\}$. The calculation of $\gamma(d,e)$ and $R_r(d,e)$ is performed according to the following procedure:

$$\begin{gathered}
\gamma(0,))=0,\ \gamma(d,0)=\sum_{c=1}^{d} l_{\hat{K}}\left(\rho_c^i,\hat{\eta}\right),\ \gamma(0,e)=\sum_{c=1}^{e} l_{\hat{K}}\left(\hat{\eta},y_c^{(i-1)}\right),\\
R_1(d,e)=l_{\hat{K}}\left(\rho_d^i,\hat{\eta}\right)+\gamma(d-1,e),\\
R_2(d,e)=l_{\hat{K}}\left(\hat{\eta},y_e^{(i-1)}\right)+\gamma(d,e-1),\\
R_3(d,e)=l_{\hat{K}}\left(\rho_d^i,y_e^{(i-1)}\right)+\gamma(d-1,e-1),\\
\gamma(d,e)=\min\left\{R_1(d,e),R_2(d,e),R_3(d,e)\right\}.
\end{gathered} \tag{3.20}$$

To calculate the integration result $Y^{(i)}\left(\mathrm{P}_1,\ldots,\mathrm{P}_i,\omega_1,\ldots,\omega_i\right)$ at the i-th step we introduce two auxiliary functions $\varphi_{\mathrm{P}}: \left\{0,\ldots,m_i+m_{Y_{i-1}}\right\}\to\left\{1,\ldots,m_i\right\}$ and $\varphi_Y: \left\{0,\ldots,m_i+m_{Y_{i-1}}\right\}\to\left\{1,\ldots,m_{Y_{i-1}}\right\}$, whose calculation is performed according to the following recursive procedure:

$$\begin{gathered}
\varphi_{\mathrm{P}}(0)=m_i,\\
\varphi_Y(0)=m_{Y_{i-1}},\\
\varphi_{\mathrm{P}}(c+1)=\begin{cases}\varphi_{\mathrm{P}}(c)\,\forall R_2\left(\varphi_{\mathrm{P}}(c),\varphi_Y(c)\right)\wedge R_1\left(\varphi_{\mathrm{P}}(c),\varphi_Y(c)\right)\neq\gamma\left(\varphi_{\mathrm{P}}(c),\varphi_Y(c)\right),\\ \varphi_{\mathrm{P}}(c)+1\,\forall else,\end{cases}\\
\varphi_Y(c+1)=\begin{cases}\varphi_Y(c)\,\forall R_1\left(\varphi_{\mathrm{P}}(c),\varphi_Y(c)\right)=\gamma\left(\varphi_{\mathrm{P}}(c),\varphi_Y(c)\right),\\ \varphi_Y(c)+1\,\forall else.\end{cases}
\end{gathered} \tag{3.21}$$

The integrated result at the i-th step is calculated as follows:

$$m_{Y_i}=\min\left\{c:\varphi_{\mathrm{P}}(c)=\varphi_Y(c)=0\right\}, \tag{3.22}$$

$$Y^{(i)}\left(\mathrm{P}_1,\ldots,\mathrm{P}_i,\omega_1,\ldots,\omega_i\right)=y_1^{(i)}y_2^{(i)}\ldots y_{m_{Y_i}}^{(i)},$$

$$y_c^{(i)}=\begin{cases}y^{(2)}\left(y^{(i-1)}_{\varphi_Y(\varphi(c))+1},\hat{\eta},Z_{i-1},\omega_i\right)\forall\,\varphi_{\mathrm{P}}\left(\varphi(c)\right)=\varphi_{\mathrm{P}}\left(\varphi(c)-1\right),\\ y^{(2)}\left(\hat{\eta},\rho^{i}_{\varphi_{\mathrm{P}}(\varphi(c))+1},Z_{i-1},\omega_i\right)\forall\,\varphi_Y\left(\varphi(c)\right)=\varphi_Y\left(\varphi(c)-1\right),\\ y^{(2)}\left(y^{(i-1)}_{\varphi_Y(\varphi(c))+1},\rho^{i}_{\varphi_{\mathrm{P}}(\varphi(c))+1},Z_{i-1},\omega_i\right)\forall\,else,\end{cases}$$

where $Z_i\overset{def}{=}\sum_{c=1}^{i}\omega_c$; $\varphi(c)\overset{def}{=}m_{Y_i}-c+1$ is the auxiliary function, and $y^{(2)}$ is the function for integrating the two recognition results of single objects (3.16).

It should be noted that within the proposed algorithm, the integration function (3.16) requires the following property:

$$\begin{aligned}&y^{(M)}\left(\alpha_1,\ldots,\alpha_M,\omega_1,\ldots,\omega_M\right)=\\&=y^{(2)}\left(y^{(M-1)}\left(\alpha_1,\ldots,\alpha_{M-1},\omega_1,\ldots,\omega_{M-1}\right),\alpha_M,\omega_1+\ldots+\omega_{M-1},\omega_M\right).\end{aligned}\tag{3.23}$$

In the case where the used function y does not possess the property (3.23), the alignment procedure remains unchanged, and the integrated result at step i must be calculated for each component of the resulting string using formula (3.18), after explicitly restoring the functions *align* and *match*.

Within the framework of this dissertation, it is proposed to use the weighted average as the function y, which possesses the property (3.23):

$$y^{(M)}\left(\alpha_1,\ldots,\alpha_M,\omega_1,\ldots,\omega_M\right)(k)=\frac{1}{Z_M}\sum_{i=1}^{M}\alpha_i(k)\omega_i,\ \forall k\in K\cup\{\eta\}.\tag{3.24}$$

In pseudocode form, the procedure for integrating the recognition results of a string object is presented as Algorithm 1.

Algorithm 1 – Algorithm for Integrating the Recognition Results of a String Object: Calculation of $Y^{(M)}\left(\mathrm{P}_1,\mathrm{P}_2,\ldots,\mathrm{P}_M,\omega_1,\omega_2,\ldots,\omega_M\right)$.

Input. $M > 0$ and $\forall i \in \{1,\ldots,M\} : |\mathrm{P}_i| > 0$

1. $Y \leftarrow \mathrm{P}_1$

2. $Z \leftarrow \omega_1$

3. for $i = 2$ to M do

4. $\gamma(0,0) \leftarrow 0$

5. $r(0,0) \leftarrow 0$ /*Path label*/

6. for $c = 1$ to $|\mathrm{P}_i|$ do

7. $\gamma(c,0) \leftarrow \gamma(c-1,0) + l_{\hat{K}}\left(\rho_c^i, \hat{\eta}\right)$ /* $\mathrm{P}_i = \rho_1^i \rho_2^i \ldots \rho_{|\mathrm{P}_i|}^i$ */

8. $r(c,0) \leftarrow 1$ /*Path 1 – alignment of ρ_c^i and the empty symbol*/

9. end for 6.

10. for $c = 1$ to $|Y|$ do

11. $\gamma(0,c) \leftarrow \gamma(0,c-1) + l_{\hat{K}}\left(\hat{\eta}, y_c\right)$ /* $Y = y_1 y_2 \ldots y_{|Y|}$ */

12, $r(c,0) \leftarrow 2$ /*Path 2 – alignment of y_c and the empty symbol*/

13. end for 10.

14. for $d = 1$ to $|\mathrm{P}_i|$ do

15. for $e = 1$ to $|Y|$ do

16. $R_1 \leftarrow l_{\hat{K}}\left(\rho_d^i, \hat{\eta}\right) + \gamma(d-1,e)$

17. $R_2 \leftarrow l_{\hat{K}}\left(\hat{\eta}, y_e\right) + \gamma(d,e-1)$

18. $R_3 \leftarrow l_{\hat{K}}\left(\rho_d^i, y_e\right) + \gamma(d-1,e-1)$

19. $\gamma(d,e) = \min\{R_1, R_2, R_3\}$

20. if $R_1 = \gamma(d,e)$ then

21. $r(d,e) \leftarrow 1$

22. else if $R_2 = \gamma(d,e)$ then

23. $r(d,e) \leftarrow 3$

24. else $R_2 = \gamma(d,e)$ /*Path 3 – alignment of ρ_d^i and y_e */

25. end if 20.

26. end for 15.

27. end for 14.

28. $Y' \leftarrow \varnothing$ /*Empty string*/

29. $\varphi_{\mathrm{P}} \leftarrow \left|\mathrm{P}_i\right|$

30. $\varphi_Y \leftarrow \left|Y\right|$

31. while $\varphi_{\mathrm{P}} > 0$ or $\varphi_Y > 0$ do

32. if $r\left(\varphi_{\mathrm{P}}, \varphi_Y\right) = 1$ then

33. $Y' \leftarrow y\left(\hat{\eta}, \rho^i_{\varphi_{\mathrm{P}}}, Z, \omega_i\right)$ /*Insertion of a new element at the beginning of Y'*/

34. $\varphi_{\mathrm{P}} \leftarrow \varphi_{\mathrm{P}} - 1$

35. else if $r\left(\varphi_{\mathrm{P}}, \varphi_Y\right) = 2$ then

36. $Y' \leftarrow y\left(y_{\varphi_Y}, \hat{\eta}, Z, \omega_i\right) Y'$

37. $\varphi_Y \leftarrow \varphi_Y - 1$

38. else

39. $Y' \leftarrow y\left(y_{\varphi_Y}, \rho^i_{\varphi_{\mathrm{P}}}, Z, \omega_i\right) Y'$

40. $\varphi_{\mathrm{P}} \leftarrow \varphi_{\mathrm{P}} - 1$

41. $\varphi_Y \leftarrow \varphi_Y - 1$

42. end if 32.

43. end while 31.

44. $Y \leftarrow Y'$

45. $Z \leftarrow Z + \omega_i$

46. end for 3.

47. end

The computational complexity of calculating the functions (3.3) and (3.24) is $O\left(N\right)$, where N is the number of classes into which each individual object is classified. Since the upper bound on the length of the resulting string Y after

performing the i-th iteration of the algorithm is $O\left(\sum_{j=1}^{i}\left|\mathrm{P}_i\right|\right) \le O\left(i \max_{j=1}^{i}\left|\mathrm{P}_i\right|\right)$, the complexity of each iteration of the algorithm can be estimated as $O\left(I^2MN\right)$, where $I = \max_{i=1}^{M}\left|\mathrm{P}_i\right|$. The total complexity of Algorithm 1 is $O\left(I^2M^2N\right)$.

3.5 Experimental Investigation of Integrating the Recognition Results of a String Object

This section will present the results of the experimental study on the performance of the algorithm for integrating the recognition results of string objects, as described in Section 3.4. The object of recognition considered was the text area of an identity document.

The experimental study was conducted on the open MIDV-500 dataset [88], which holds 50 different types of identity documents videos (with 10 video clips for each certificate; 30 frames per video) with explained ideal locations and content of text areas. Four groups of fields were analyzed: dates recorded with numbers and punctuation marks, Machine-Readable Zone (MRZ) strings, certificate number, and elements of the certificate holder's name written in the Latin alphabet.

Only frames in which the entire document is visible were considered (thus, the video sequences in the examined subset of a dataset had varying sizes, from 1 to 30 frames). To decrease the effects of normalization and offer a clearer representation of the outcomes, every clip was extended in duration to 30 frames by repeating the clip from the beginning (thus, all analyzed clips had an identical duration of 30 frames).

Each area was extracted from the source image using a projective transformation, according to the joint annotation of the ideal boundaries of the document and the coordinates of the text area, with margins increased to 30% of the shortest side of the text area. The size of the extracted text area images corresponded to a resolution of 300 dots per inch. Each extracted text area was

recognized using the component of the Smart IDReader system [99] responsible for recognizing a single text string, with an extended result model (3.9).

The normalized Levenshtein distance (3.12) between the true value and the text string obtained using the procedure (3.11), described in Section 3.3, was used as the distance between the integrated recognition result of the text area and its true value. All character comparisons were performed regardless of case, and the Latin letter "O" was considered identical to the digit "0".

In the framework of this experimental study, Algorithm 1, operating within the extended model of the string object recognition result, was compared with an analog operating within the classical model. For each group of text areas and each video sequence, integration was performed using the ROVER method, where simple text strings formed by the procedure (3.11), applied to the frame-by-frame recognition results, were used as input data. The threshold λ for the empty symbol score (3.11) was set to 0.6 for both the control ROVER method and Algorithm 1.

Figure 3.3 presents the results of the compared algorithms for the four groups of text areas in the MIDV-500 dataset. It can be noted that for each group of fields, the integration using Algorithm 1 of the full recognition results (i.e., considering alternative recognition options for each individual character) achieves a lower error value than integration using the ROVER method (which only considers the first alternatives for the recognition of each character), regardless of the length of the sequence of integrated results.

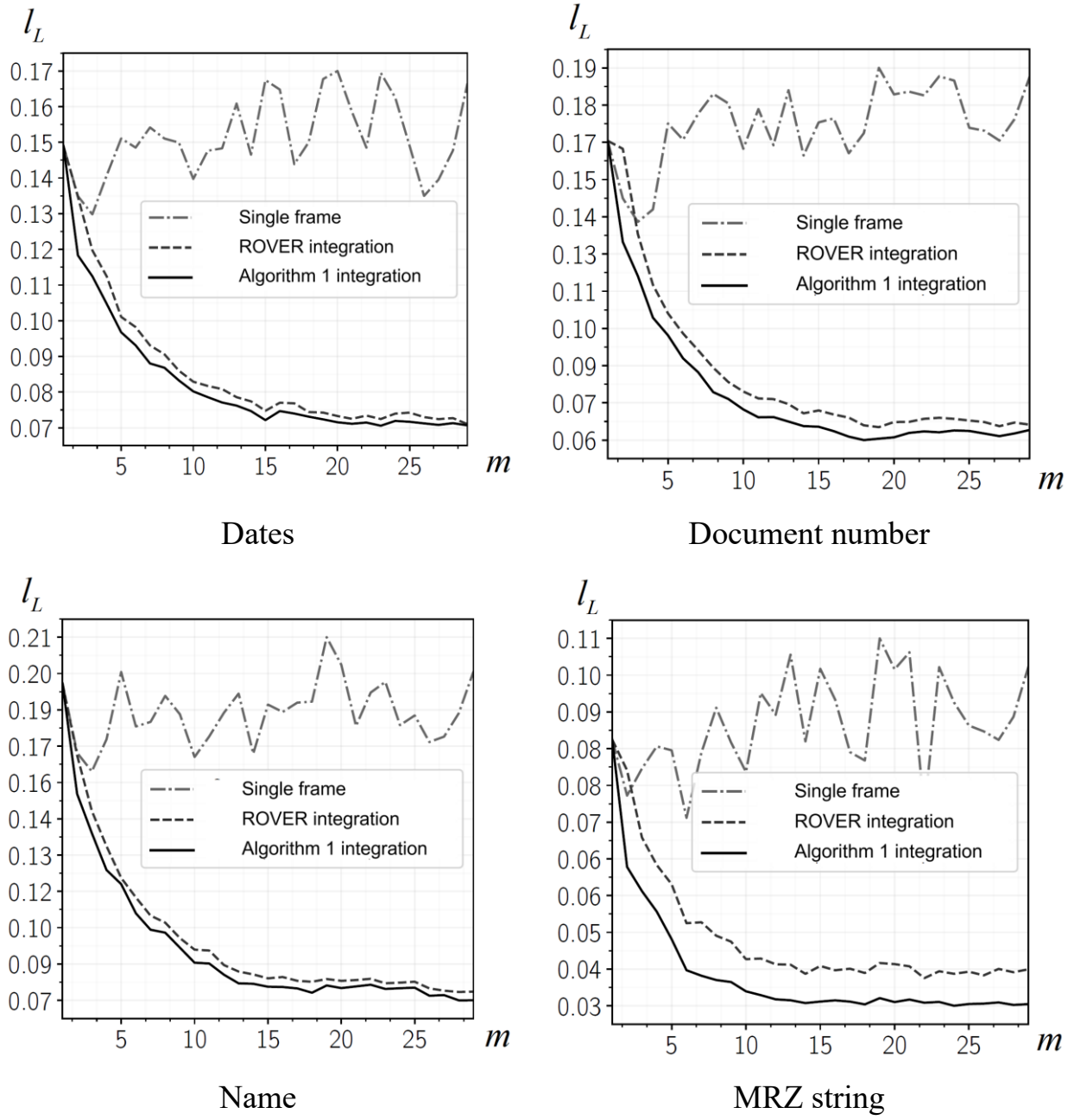


Figure 3.3 – Results of the integration algorithms for four groups of text areas in the MIDV-500 dataset.

Table 3 – Achieved distance between the integrated recognition result and the true value without integration, using the ROVER method, and with Algorithm 1.

Integration method	Frame number (length of the sequence of integrable results)								
	3	6	9	12	15	18	21	24	27
Without integration	0.136	0.154	0.160	0.157	0.168	0.159	0.165	0.166	0.150
Integration using Algorithm 1	0.115	0.089	0.078	0.071	0.066	0.065	0.066	0.066	0.064
Integration using the ROVER method	0.125	0.096	0.083	0.075	0.070	0.069	0.069	0.069	0.067

Figure 3.4 presents the results of the algorithms applied jointly to all four groups of fields. The achieved average values of the distance between the integrated recognition result of the text area and its true value for different lengths of the integrated video sequence prefix are presented in Table 3.

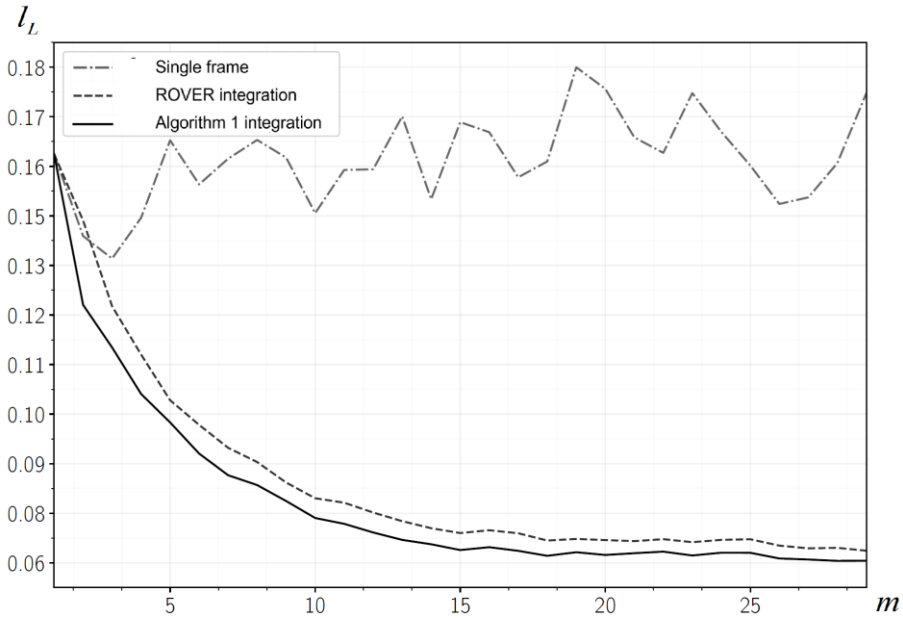


Figure 3.4 – Results of the integration algorithms for text areas in the MIDV-500 dataset.

3.6 Data Synthesis Method for Training and Tuning Image Recognition Algorithms

The quality of document recognition, including the recognition of textual attributes or graphical objects, directly depends on the quality of training of individual classifiers. An example of such a classifier might be a stamp classifier or a classifier for individual text characters. The initial stage of building a classifier involves forming a training dataset. For instance, in the case of a character classifier, the training dataset may consist of images of characters of various types (black-and-white (binary), grayscale, colored) and the attributes corresponding to the characters (character code in a predefined alphabet, font features such as boldness, italicization, and typeface). The combination of attributes defines the training and classification alphabets.

The effectiveness of classifier training, i.e., achieving high recognition accuracy and maintaining monotonicity in recognition reliability assessments, is highly dependent on the size of the training dataset and the precision in attributing the established character attributes.

Generally speaking, all the aforementioned also applies to the test dataset, which is necessary for evaluating the quality of the trained classifier. However, within this section, we will focus exclusively on training datasets. The training process is typically viewed as an iterative procedure, meaning that different training datasets are used for initial training and subsequent retraining sessions.

When constructing a training dataset, special attention must be paid to the represetativeness of the data. In [72], key principles for forming a training dataset are highlighted:

- Sufficiency: The number of training examples must be adequate for reliable training. Naturally, the required number of examples can vary depending on the training models. For instance, in the case of a neural network, the number of training examples should be several times greater than the number of weights in the inter-neuronal connections; otherwise, the model may fail to acquire the ability to generalize [72]. In practice, sufficiency is assessed using the characteristics of the trained method, such as analyzing the dependence of recognition accuracy on the number of training examples, assuming that this dependence is monotonic.

- Diversity: A large number of diverse combinations of features in the training examples. This principle is closely related to the previous one and intensifies the requirements for the number of training examples in which the classifier's features and their combinations are explicitly evaluated. Diversity can be assessed using clustering based on feature representation, with diversity measured, for example, by the dependence of the number of resulting clusters on the number of training examples.

- Class frequency distribution: Depending on the classification method used, examples of different classes often need to be represented in the training dataset in proportions approximately corresponding to the class proportions in the test dataset.

Dominant classes are identified as more likely for new observations. For instance, when creating a classifier for standard texts in a specific language, it is reasonable to consider the frequency distribution of individual character occurrences [73].

A set of character images for training a classifier can be created in various ways: generated artificially or extracted from images (either those intended for recognition or similar images). Subsequently, the character images must undergo "labeling," which involves assigning attributes to each image, at a minimum the character code (the term "annotation" is also used in the literature). Labeling can be performed either automatically or manually. When creating document recognition classifiers, manual labeling is mandatory if training relies on examples extracted from sequences of random document images.

Manual labeling can be based on presenting either an individual character to the operator (the person performing the labeling) or a portion of a document image with the contextual surroundings of the character. The latter method of labeling is more accurate than the former but requires more operator time to analyze each character.

The task of forming a training dataset of character images involves obtaining as many reliably labeled and diverse images as possible. In other words, the task can be outlined as follows:

- Estimating the dataset size and, if applicable, assessing the number of feature combinations;

- Minimizing mislabeled images and evaluating the proportion of mislabeled images;

- Assessing the alignment of class frequency distribution with a predefined distribution, if necessary.

Even for a single classification task essential in document recognition systems, including identity documents—such as the classification of printed characters—the task of constructing a dataset of training images can be quite labor-intensive and costly [74; 75]. During the formation of a training dataset from real

data, preprocessing tasks must be addressed, such as character detection, noise removal, and the elimination of extraneous objects.

The complexity of building a dataset of individual character images is highlighted in [76], which notes that the quality of the training dataset directly depends on the accuracy of the character boundary detection algorithm: each character must be strictly centered, and identical characters must have consistent dimensions.

Studies [74; 75] emphasize the creation of specialized forms that must be completed according to specific rules. This approach to forming a training dataset requires significant human resources. During the form completion stage, as previously noted, it is important to obtain the widest possible range of variations for writing each class of images. This feature necessitates involving a larger number of respondents, making this approach to creating a training dataset expensive and inefficient.

In [77], various methods for obtaining a database of graphical symbols are discussed. A clear advantage is noted in saving character images directly from the document recognition software. For instance, in such a dataset creation scheme, the boundaries of each character are determined with high accuracy. Naturally, the question of the reliability of recognizing individual characters arises.

When using real-world data (i.e., not artificially synthesized images of recognizable objects but those extracted from real-world objects) to create large, representative training datasets, significant resources are required for labeling.

In the case of manual labeling, a distinct challenge arises: how to create the largest possible training dataset within a limited time given the available operator resources. This subsection will consider a model for the process of forming a large-scale training dataset, using the example of creating a dataset for training a single-character text classifier from real-world data.

Assume there is a source of images supplying both character images and "non-character" images (i.e., images of objects that should be classified as not

belonging to any predefined character class). Two mechanisms for labeling character samples from a given source are considered:

- An automatic image classifier based on the principles of optical character recognition (OCR),

- Operator-based labeling, where images can be either pre-classified or unclassified.

Automatic labeling is fast but prone to errors. Manual labeling is more accurate but limited by the speed of the operators.

The labeling process for a set of character images may include the following operations:

- Modifying the character image;

- Adjusting the character boundaries;

- Verifying the character boundaries with possible rejection (removal of the image from the training dataset);

- Entering the character code;

- Verifying the character code with possible rejection.

These operations are listed in descending order of the time required to complete them. The time spent on verifying the character code with possible rejection can be reduced if operators are presented with previously classified characters that share the same code.

It is worth noting that the rejection function reduces operator workload but may also decrease the diversity of features in the training dataset.

As mentioned earlier, operators may be presented with either individual images or images in the context of neighboring characters. In the latter case, the accuracy of labeling, defined as the ratio of mislabeled images to the total number of images, improves due to the increased time spent analyzing groups of characters.

The time required to perform operations ranges from 0.3 to 0.5 seconds for verifying the character code with possible rejection (when homogeneous images sorted by character code and other attributes are presented) to 30 seconds or more for modifying a character image.

Let's consider the partition of the set I, extracted from a certain source of images, into three subsets $I_a \cup I_v \cup I_e$, where I_a is the set of confidently classified images of a symbol (these images do not require manual verification); I_v is the set of images that require manual verification, first, of the membership to symbols, and second, of the correctness of classification; and I_e is the set of images that cannot be automatically classified and that the operator must reclassify manually during the manual verification process.

Let us define the time estimates t_v and t_e for processing one image by the operator from the sets I_v and I_e, respectively. Then, the total processing time by the operator for the set t is determined as

$$t = |I_v| t_v + |I_e| t_e . \tag{3.25}$$

The method of partitioning the set I will define the processing time as t. For large volumes of sets, the processing time is almost always limited. For example, for $|I| = 10^6$ images, with $t_e = 0.5$ seconds per image, the total processing time for each symbol will be approximately 18 days. Therefore, it follows that it will not be possible to rely on a single operator for the described task, and automation tools are necessary for the process of forming the training set.

Difficulties often arise when classifying similar symbols, such as the need to distinguish between the digit "0" and the letter "O". To address this issue, it is necessary to refer to the image of the text field from which the symbol was extracted. The operator should have access to the original text string with the indication of the current symbol in the text field. Thus, the context of the field significantly improves the quality of the training set.

From the above, it follows that the method of partitioning the set I into subsets I_a, I_v, I_e and the method of representing the elements of these subsets allow minimizing the time spent by the operator on processing the subsets I_v, I_e and minimizing the number of classification errors for the images in the set I. It should be noted that the method of partitioning the set I into subsets may also

include manual operations, which need to be taken into account when estimating the total time costs.

In the document recognition task, for example, when archiving a stream of document images, incorrectly recognized images must be corrected or, at the very least, marked as unreliable. The verification and editing stages of the recognition results are determined by the business logic of the document recognition system [78]. These stages are carried out by operators from the organization operating the OCR system.

Quite often, the results of document recognition, that is, the documents in digital form, cannot be transmitted to the developers of the OCR system from the organization operating the OCR system, primarily due to information security requirements (especially when it comes to identity documents). However, the recognition results, consisting of the set $I_a \cup I_v \cup I_e$, do not allow the restoration of the original documents and can be transmitted to the OCR system developers for retraining the classifiers. That is, the process of partitioning into $I_a \cup I_v \cup I_e$ is carried out on the technical resources of the organization operating the OCR system. Despite the use of results from verification and editing for partitioning, the operators do not need to perform any new special actions.

The method proposed in this section for forming the set of symbol images is based on the use of recognition results of text fields and test strings, confirmed by the operator.

Let's consider the task of character-by-character matching of the recognition result of a string (a set of alternatives with weights for each image) and the corresponding sequence of symbols confirmed by the operator during the editing and verification stages. In other words, each symbol of the text string needs to be matched with the symbol image from the recognized string.

The solution to the problem, that is, matching the recognition result with the set of symbols, will be carried out using the dynamic programming method with the Levenshtein distance metric [79]. We will base our approach on the fundamental principles of the MCHSR algorithm [80]. MCHSR is one of the

methods for context-based processing of recognition results. This algorithm was developed for finding the occurrence of a text fragment within the recognition result string. However, we are considering the task of complete optimal matching of the verified string with the recognition result.

In the first step of the algorithm, we will construct a table where the cells will indicate the quality of the symbol classifier if the symbol matches one of the alternatives for the current position. If the current symbol is not present in the list of alternatives, the cell in the table will remain empty.

In the second step of the algorithm, we will find the best path (the path with the highest weight) from the bottom-left point of the table to the top-right point. The following transitions are allowed (see Fig. 3.5):

- Moving up along the cell's edge: this occurs when there is no alternative in the recognition results corresponding to the symbol entered by the operator (the symbol image was not recognized). For simplicity, we will assume the transition cost is 0.

- Moving right along the cell's edge: this occurs when none of the symbols in the string corresponds to the recognition result (a "garbage" fragment is recognized as a symbol). Similarly, we will assume the transition cost is 0.

- Diagonal transition: matching the symbol with one of the alternatives for the current position. The transition cost will be equal to the value in the cell.

As a result of the above-described algorithm, a set of correspondences will be formed: test string symbol – symbol image. There may be cases where a raster image is not found for a symbol, or conversely, a symbol is not found for a raster image.

After matching, each raster image can be classified into one of three types:

- Confidently recognized symbol image: for this image, the best alternative (the one with the highest weight) corresponds to the symbol from the string, and the alternative itself has high recognition quality.

- Image requiring confirmation: for this symbol image, one of the secondary alternatives (not the best alternative) corresponds to the symbol from the string.

- "Incorrectly" recognized symbol image: a symbol image for which no corresponding symbol from the string is found.

T				*254*						*14*
O			*147*						*245*	
Y	2	*84*						*254*		
M						*243*		*3*		
K	*1*				*251*					
T				*254*						*174*
P			*188*							
B		*150*						*213*		
L	*242*				*2*					
	L	B	P	T	K	M	″	Y	J	C
	242	150	168	254	251	243	254	254	249	223
	K	Q	J	,	Z	D		B	.	T
	1	133	147	2	5	7		213	67	174
	Y	Y	D	8	1	R		D	9	**2**
	1	84	55	2	2	3		3	2	2
		1		T	L	S		M		**7**
	83		2	2	3		3		2	

Figure 3.5 – Model for matching the recognition result and the text string entered by the operator: constructing the table and finding the best match between raster images and symbols (indicated by the bold gray line).

Thus, we obtained three sets of symbol images I_a, I_v, I_e. The stored symbols can be represented as binary, grayscale, or color images. In the latter cases, training may require not only the images themselves but also the parameters for separating grayscale and color letter images from the background. In the simplest case, the background separation threshold can be taken from the binarization results of the group of symbols forming the string and refined using adaptive thresholding algorithms, such as the Niblack method [63].

Let us test the proposed model and method experimentally. For the experiment, more than $2 \cdot 10^6$ symbols were obtained, among which more than 80% of the images were assigned to the set I_a. The proportion of symbols requiring additional verification by the operator was less than 10%, which significantly speeds up the process of creating the training set. The set I_e consisted of about 8% of the symbols.

At the stage of assigning the symbol image to one of the three groups, it is proposed to consider the symbol reliably recognized if the first alternative has high quality. Let us explore the size of the set of images I_v based on the fixed quality of the first alternative symbol Υ_0. In other words, a symbol does not need further confirmation if the quality of the first alternative is $\upsilon > \Upsilon_0$; otherwise, the symbol falls into the set of doubtfully recognized images. Fig. 3.6 shows the results of the experiment for the "Passport" documents.

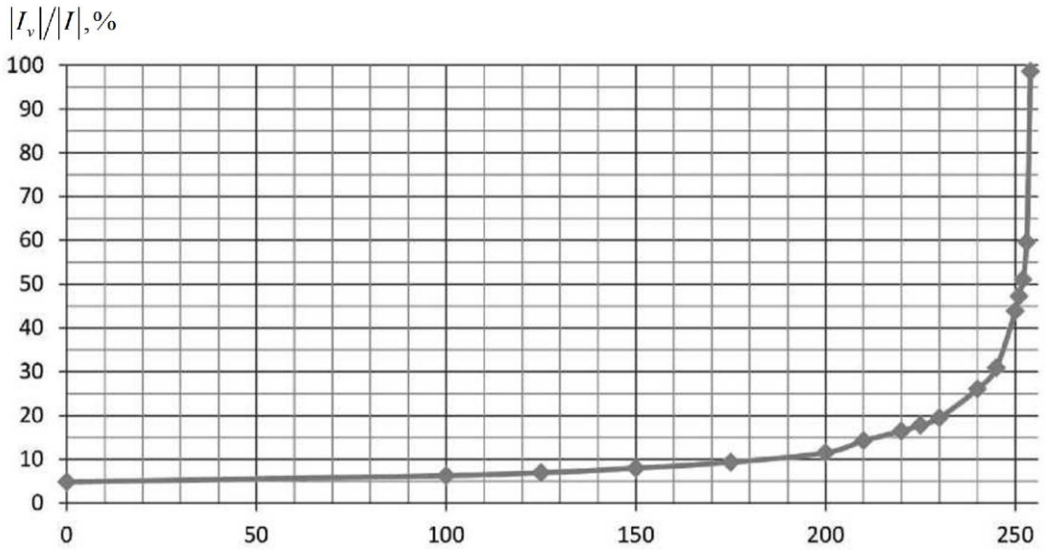


Figure 3.6 – Dependence of the size (in % of the total number of images) of the set I on the minimum acceptable quality value of the image from the set I_a for "Passport" documents.

The central focus in the task of forming a training set is the reliability of image classification. Let's analyze the dependence of the number of errors in the set I_a on the quality of the first alternative symbol. To do this, we will count the number of errors for each range of alternative quality values in a "Passport" document sample. The reduction in the error rate as the quality of the alternative increases is shown in Fig. 3.7.

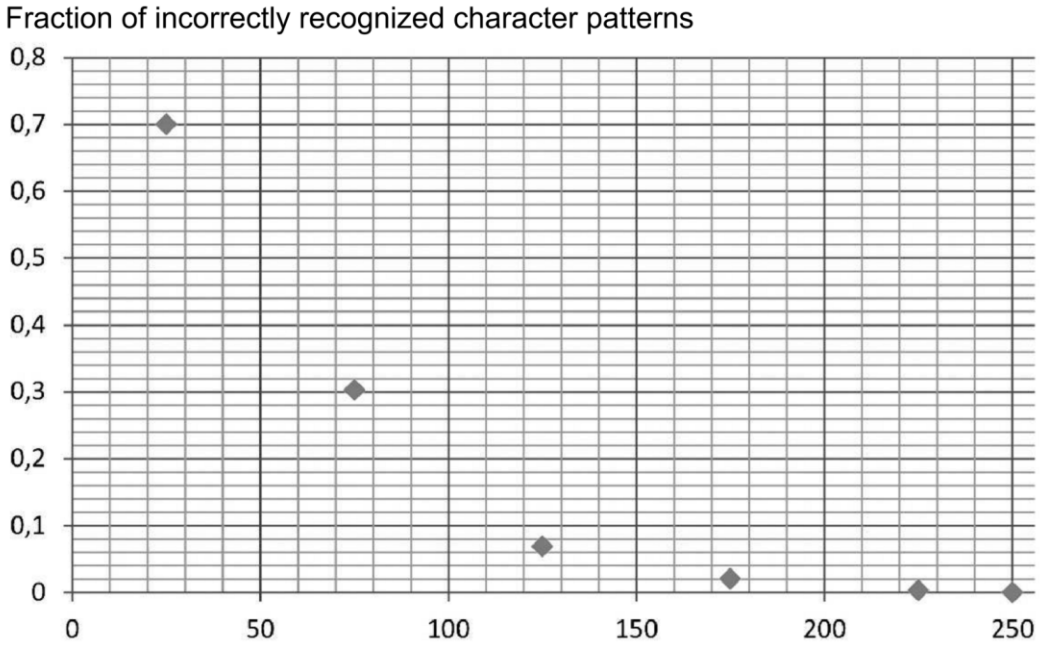


Figure 3.7 – Dependence of the number of incorrectly recognized symbols on the quality of the first alternative.

The question arises whether it is possible to create a set I_a where the proportion of incorrectly classified images does not exceed a pre-defined number τ. Investigation of this problem showed (see Fig. 3.8) that as the quality of the first alternative Υ_0 increases, the proportion of incorrectly classified images in the set I_a monotonically decreases. Thus, for a given number of incorrectly classified images υ, there exists a value Υ_0 such that the set I_a contains fewer than τ incorrectly classified images.

Let's calculate the required time for creating a set of graphical images using the proposed method and compare it with the time required to classify each symbol in the set I. The calculation will be performed for character images obtained from a "Passport" document sample. We assume that it is necessary to label 10^6 character images, with the proportion of incorrectly classified symbols not exceeding 0.1%. According to the calculations presented earlier, labeling all the symbols will take 18 days.

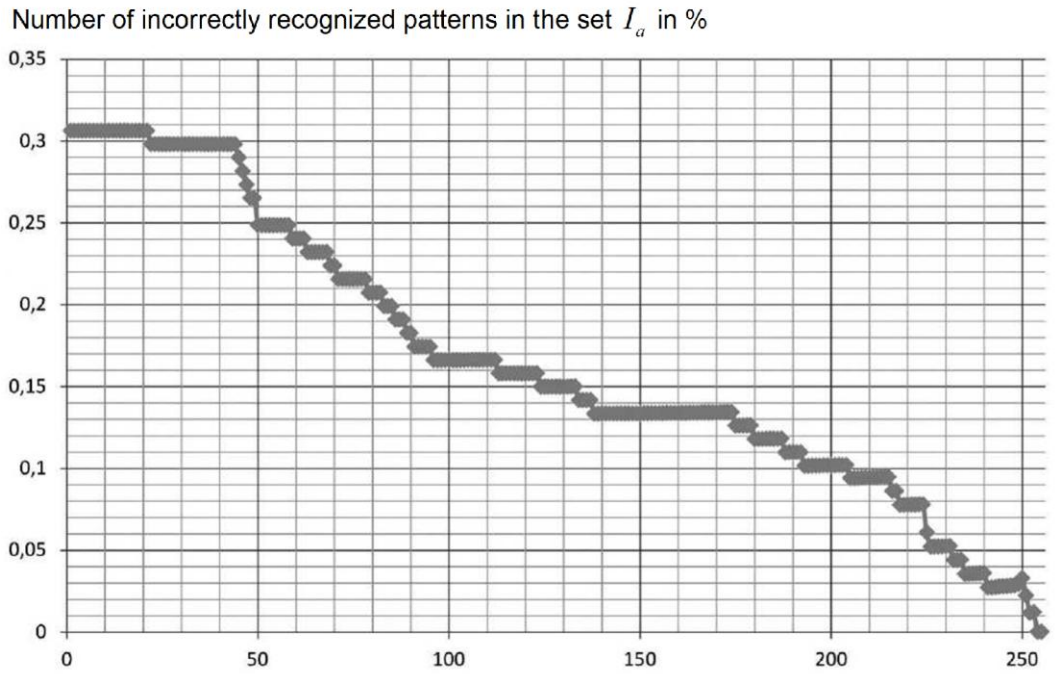


Figure 3.8 – Dependence of the proportion of incorrectly recognized images in the set I_a on the quality of the symbols that make up this set.

The set I_a will contain less than 0.1% incorrectly classified images when $\Upsilon_0 = 200$ (see Fig. 3.7). For the obtained value of Υ_0, the size of set I_v will be 10^6 images (see Fig. 3.7). Therefore, the time required for labeling 10^6 character images will be approximately $t = |I_v| t_v \approx 2$ days.

It is important to note that a real operator cannot work 8 hours a day with constant productivity, which will result in a proportional increase in time costs for labeling using both methods. Nevertheless, the calculation conducted shows that the proposed method significantly accelerates the process of forming the training set (by 9 times in the example provided). The proposed method allows for improving the OCR system classifier used in a given organization, primarily through labeling as prescribed by the operators' workflow regulations within this system.

3.7 Methodology for Creating Open Data Packages of Identity Documents

To keep up with the current pace of development in computer vision methods and approaches, particularly considering the trend of using deep neural network models, it is necessary to create large volumes of data for training and testing. In the context of specific tasks, such as the analysis and recognition of identity documents, this presents challenges – either related to the specifics of the target object (such as compliance with legal and ethical standards regarding the security of sensitive data) or associated with the limited size of the community working on such tasks. However, as the experience with MIDV data packages demonstrates, even if the primary experimental data package is not published, a limited open data package like MIDV-500 can be used as an additional dataset for publishing an objective evaluation of the quality of a given method, which can be compared with others.

The authors of several open data packages preemptively divide the published samples into training, validation, and test sets. However, as can be observed from the practice of using MIDV family data packages, such a division may not always be meaningful – the variety of approaches to solving specific tasks is so vast that researchers are often forced to independently choose data splitting protocols for training and validation. Sometimes, for solving higher-level tasks, a combination of several data packages from different domains is used (as, for example, in the work [118]).

A much more important aspect of building open data packages is structuring the packages in such a way that researchers can extract subsets defined by specific characteristics. For example, the authors of the analyzed works separately analyzed subsets of data packages based on lighting conditions and the degree of projective distortion [128], the number of visible corners of the document in the frame [108; 310], the presence and integrity of certain types of text fields or machine-readable zones [127; 132], and so on. Since one of the goals of publishing open data packages is to provide different research groups with the opportunity to compare developed methods and approaches on a common experimental basis, it is

necessary to explicitly specify the subsets of the data package used in a particular experiment.

The most important issue in the creation and development of open data packages is their expansion, either with new data (to increase the overall volume or to add data with new, previously unrepresented features), or with the addition of specific, problem-oriented annotations (for example, bounding rectangles around facial ovals [124], ideal binarization of document images [71], expert annotation of image quality or image fragments [36], etc.), or with derivative, generated images (such as images with background texture distortions of the document [42]). Often, such expansions are used by authors for specific studies but are not made publicly available for reproducing results or for comparative analysis. Therefore, an important aspect of open data packages is providing tools for users to extend these data packages with new images or new annotations, so that other research groups can select not only the subsets of published data they are interested in but also specific derivative images or pre-prepared annotations. This helps reduce inconsistencies when comparing methods and simplifies the research methodology [45].

Based on the experience of creating open data packages for evaluating document analysis and recognition systems for identity documents, as well as an analysis of their use in the scientific community, the following key stages and methodological principles for creating such data packages can be formulated:

1. Source Images of Identity Documents: The source images of identity documents should be taken from publicly available examples in open repositories, such as Wikimedia Commons. Most images of identification documents in such repositories can be used for any purpose, as, according to local legislation, images that are part of regulatory documents cannot be copyrighted. It should also be noted that many of these images have the word "SAMPLE" printed on the document in some language, which may interfere with document recognition and does not align with the intended use case for document recognition systems. Therefore, such elements should be retouched. In some cases, it may be useful to

create entirely synthetic templates of identification documents, not tied to any specific country or issuing authority, as done for the MIDV-Holo dataset, following international standards and recommendations for the structure of such documents.

2. Generating Random Values for Textual Attributes: Openly available random name generators in various languages can be used to generate random values for textual attributes. For generating random images of the document holder's face, existing services for generating synthetic faces, such as Generated Photos [101], based on Generative Adversarial Networks (GANs), can be used.

3. Publishing Templates with Artificial Data: Prepared templates of documents with artificial data should be published as part of the data package, as such source images allow for the full annotation of attribute values and their geometric placement relative to the document template. Annotations should be made using well-known and validated tools, whose data storage formats are already recognized by the scientific community. An example of such a tool is the VGG Image Annotator [103]. Additionally, when publishing source templates, the scientific community will have the opportunity to complement the data packages with new photographs or video sequences, including those captured in conditions not represented in the original data package, or with simulated attacks of other types.

4. Simulating Glare Effects on Plastic Documents: Since most identity documents are made of smooth plastic or their main pages are protected by polymer films, glare often appears on the document surface when captured by mobile device cameras. To simulate this effect, printed examples of documents should be laminated before filming.

5. Document Shooting and Lighting Conditions: When shooting video or photographing documents, it is important to classify the external shooting conditions and include this information in the data package structure or its description. Key shooting parameters include the device used for shooting or photographing, lighting conditions, features of the shooting scenario (e.g., glare

avoidance, acceptable ranges of projective distortions, etc.), background characteristics, and more.

6. Document Positioning on Scans or Photographs: On prepared scans or photographs, it is usually sufficient to annotate the precise geometric position (coordinates of the corners of the quadrilateral) of the document and the template type identifier. Along with the annotation of the template content, this information is enough to obtain the exact reference location of each document object.

7. Video Sequences for Annotation: Prepared video sequences should be frame-extracted with a known and specified frame rate in the data package description. On each frame, as in the case of photographs, the exact coordinates of the document's quadrilateral corners and the template type identifier should be annotated. It is important that the original video files are also provided as part of the data package, so that researchers who require deep modeling of the document filming process can use not only annotated video frames but also the complete set of raw data.

8. Simulating Attacks for Document Authentication and Validation: When creating data packages intended for evaluating methods of detecting document presentation attacks or verifying the authenticity of documents or their components, possible types of attacks should be described, corresponding simulations should be conducted, and photos or video sequences representing the attacks should be included in the data package. These should be captured under the same shooting conditions as those used for images or sequences of "original" documents.

3.8 Conclusions of the Chapter

The chapter discussed the problem of combining multiple recognition results of a string object in order to increase the accuracy of the final recognition result. A model of the recognition result for a string object was described, which takes into account alternative classification options for individual objects, and an algorithm for integrating recognition results of string objects within the framework of the described model was presented.

Based on the results of the experimental study, the following conclusions can be made:

1. The methods for integrating the recognition results of string objects allow for a significant increase in the accuracy of the final recognition result when analyzing a video sequence.

2. The ROVER method, originally intended for combining recognition results of the same object image by multiple recognition algorithms, can also be applied to combine recognition results of different images of the same object using a single recognition module.

3. Both the ROVER method, which takes recognition results of string objects in the form of strings over a set of classes of individual objects as input, and Algorithm 1, which takes recognition results of string objects in the extended model (3.9) as input, show a significant improvement in the accuracy of the integrated result with an increasing number of frames used. In the task of recognizing text areas on identity documents, Algorithm 1 demonstrates higher accuracy than the direct application of the ROVER algorithm.

4. From the shape of the graphs showing the distance between the integrated result and the true value as a function of the number of frames used (see Fig. 3.3 and 3.4), it can be inferred that the integration exhibits the property of diminishing returns (according to the terminology of anytime algorithms [102]). This property is important for solving the object recognition stop task in a video sequence.

5. The section also proposes a method for forming a set of symbol images involves matching recognition results of text fields with test strings confirmed by the operator during the editing and verification stages, using dynamic programming and the Levenshtein distance metric. This approach, based on the MCHSR algorithm, adapts it for optimal matching of a verified string with the recognition result. The process consists of two steps: first, constructing a table that evaluates symbol matching quality, and second, finding the best matching path through the table, considering up, right, and diagonal transitions. The algorithm

also handles cases where no symbol match is found, either due to unrecognized symbols or extra symbols in the recognition result.

6. In creating open data packages for evaluating identity document recognition systems, key methodological principles include using publicly available source images, generating random data for document attributes, and publishing annotated document templates with artificial data. Essential aspects also involve simulating realistic shooting conditions (e.g., glare on plastic documents), classifying shooting parameters, and annotating precise geometric positions of documents. Video sequences should be annotated frame by frame, with corresponding raw data provided. Additionally, the creation of attack simulations for document authentication and validation is crucial, ensuring that such attacks are presented under the same shooting conditions as original documents. These steps promote reproducibility, consistency, and comparability in research while facilitating future extensions and methodological advancements.

4 THE OBJECT RECOGNITION PROCESS STOPPING PROBLEM IN CONTEXT OF REAL-TIME REAL-TIME VIDEO STREAM

4.1 Introduction

In addition to the task of integrating object recognition results under multiple observations, the processing of real-time video streams involves the challenge of optimal stopping. The stopping problem is particularly relevant for computer vision systems that perform real-time object recognition on mobile devices [99]. In such systems, the time needed to acquire the recognition outcome is frequently as critical as the precision of the outcome.

This chapter examines the object recognition process stopping problem in a real-time video stream. Section 4.2 presents a formal statement of the object recognition stopping problem in a real-time video stream as a problem-solving task to terminate an iterative process, assuming that confidence estimates for recognition results are unavailable. Section 4.3 describes the properties of monotonic stopping problems, and Section 4.4 proposes a method for solving the problem in the given formulation by reducing it to a monotonic problem. Section 4.5 provides the results of an experimental study demonstrating the effectiveness of the proposed method for the task of recognizing a certificate's text aria.

4.2 General Problem Definition

Contemplate the object recognition problem in a real-time video stream. Let P denote the entire set of possible values of the recognizable object (e.g., the set of strings over a fixed alphabet in the case of recognizing text arias), with a determined metric $l: P \times P \to [0,\infty)$, on this set. The real-time video stream is used to recognize an object with a true value $\mathrm{P}^* \in P$. The recognition process assumes that a decision-maker observes a series of random recognition outcomes $\mathrm{P} = (\mathrm{P}_1, \mathrm{P}_2, \ldots)$, one result per step of the process, and every observation $\rho_i \in P$ is an execution of P_i.

We suppose that $P_1, P_2, \ldots$ shares the same joint distribution as P^*. Within this framework, we assume that confidence estimates for the recognition results of the object are unavailable.

We also define a family of integration functions for multiple recognition results, which return a single integrated result $Y^{(m)}: P^m \to P$, as their output. At any moment m, the observation results $P_1 = \rho_1, \ldots, P_m = \rho_m$ are available, and an integrated result $Y_m = Y^{(m)}(\rho_1, \ldots, \rho_m)$ can be acquired. The process can be interrupted at any moment $m > 0$ with the following penalty function:

$$k_d l(Y_m P^*) + k_f m, \tag{4.1}$$

where $k_d > 0$ represents the recognition error cost in relation to the range to the true value, and $k_f > 0$ denotes the every observation cost.

Since k_d and k_f are non-negative constants, a penalty function can be redefined devoid of altering the optimization problem structure as follows:

$$L_m \overset{def}{=} l(Y_m, P^*) + km, \tag{4.2}$$

where $k = k_f / k_d$ represents the functional dependence of the penalty. The penalty value when stopping at step $m = 0$ (i.e., if no observations have been obtained) can be considered infinite.

The task is to select the step at which the observation process should be stopped to minimize the anticipated penalty. This formulation can be formalized using the notation from [106].

A stopping rule can be outlined as the series of functions:

$$\Psi \overset{def}{=} (\psi_0, \psi_1(\rho_1), \psi_2(\rho_1, \rho_2), \psi_3(\rho_1, \rho_2, \rho_3), \ldots), \tag{4.3}$$

where $\forall m: 0 \le \psi_m(\rho_1, \ldots, \rho_m) \le 1$ represents the stopping criterion. The function $\psi_m(\rho_1, \ldots, \rho_m)$ reflects the probability of stopping at moment m, considering that

this step has been attained (i.e., based on the observations $\mathrm{P}_1 = \rho_1, \ldots, \mathrm{P}_m = \rho_m$ obtained up to that point).

Using a sequence of observations P and the stopping rule Ψ, it is possible to define the random stopping time M. Let $R\left(M = m \middle| \mathrm{P} = \left(\rho_1, \rho_2, \ldots\right)\right)$ denote the probability function for stopping at step m, given a specific sequence of observations P. This function is expressed in terms of the stopping rule (4.3) as follows:

$$\begin{gathered} R\left(M = 0 \middle| \mathrm{P} = \left(\rho_1, \rho_2, \ldots\right)\right) = \psi_0; \\ R\left(M = m \middle| \mathrm{P} = \left(\rho_1, \rho_2, \ldots\right)\right) = \psi_m\left(\rho_1, \ldots, \rho_m\right) \prod_{j=1}^{m-1} \left(1 - \psi_j\left(\rho_1, \ldots, \rho_j\right)\right), \\ \forall m \in \left\{1, 2, \ldots\right\}; \\ R\left(M = \infty \middle| \mathrm{P} = \left(\rho_1, \rho_2, \ldots\right)\right) = 1 - \sum_{j=0}^{\infty} R\left(M = j \middle| \mathrm{P} = \left(\rho_1, \rho_2, \ldots\right)\right). \end{gathered} \tag{4.4}$$

Alternatively, considering the random stopping time M, a stopping rule for moment $m = \left\{0, 1, \ldots\right\}$ can likewise be represented as the conditional probability of stopping at the moment m, considering a specific series of observations P, and assuming a process did not terminate at earlier stages:

$$\psi_m\left(\mathrm{P}_1, \ldots, \mathrm{P}_m\right) = R\left(M = m \middle| M \geq m, \mathrm{P} = \left(\rho_1, \rho_2, \ldots\right)\right). \tag{4.5}$$

The task is to determine a stopping rule Ψ that minimizes the anticipated loss functional $W\left(\Psi\right)$. This can be formulated as follows:

$$W\left(\Psi\right) = \mathrm{E}\left(L_m\left(\mathrm{P}_1, \ldots, \mathrm{P}_M\right)\right). \tag{4.6}$$

4.3 Optimal Stopping and Monotonic Stopping Problems

In this section, we explore the concept of optimal stopping and its relationship with monotonic stopping problems. An optimal stopping problem

involves determining the best time to stop a process in order to maximize or minimize some objective function, such as minimizing the anticipated penalty discussed in earlier sections.

A monotonic stopping problem refers to situations where the decision to stop depends on a sequence of observations that either increase or decrease monotonically over time. The key feature of monotonic problems is that the optimal stopping time is often easier to determine, as the value of stopping at each step either consistently improves or worsens based on the accumulated observations.

In the context of our problem, we can model the stopping rule as a sequence of monotonically increasing or decreasing decisions that lead to the optimal time to stop the object recognition process. The structure of the monotonic problem helps simplify the decision-making process, as the stopping decision at each step is based on a clear progression of observed data.

This section discusses how the properties of monotonicity can be leveraged to find an optimal stopping rule for object recognition tasks, minimizing the anticipated penalty associated with stopping too early or too late in the process.

4.3.1 Optimal Stopping Rule

Since the metric l cannot take negative values $L_m \geq km$, and since k is a non-negative constant, the subsequent statements hold true:

$$\mathrm{E}\left(\inf_m L_m\right) > -\infty,\ \lim_{m\to\infty} L_m \geq L_\infty . \tag{4.7}$$

If the statements in (4.7) hold true, it can be demonstrated [105; 106] that there exists an optimal stopping rule that aligns with the principle of optimality. This principle states that the decision to stop at any given step should be optimal given the information available at that point, and the remaining decisions should also follow an optimal strategy. In other words, once the stopping rule is defined for the ongoing step, it ensures that the future decisions, based on the ongoing state,

will also contribute to minimizing the anticipated penalty or loss. This recursive property simplifies the task of determining the optimal stopping time and makes the solution more tractable.

Let W_m^* denote the lowest anticipated loss for any stopping rule M, such that $R(M \geq m) = 1$, that is, for any stopping rule that arrives step m:

$$W_m^*(\rho_1,\ldots,\rho_m) = ES\inf_{M \geq m} \mathrm{E}_m(L_M), \quad (4.8)$$

where $\mathrm{E}_m(\cdot)$ refers to the conditional expectation given the sequence of observations up to step m, denoted as $\mathrm{E}(\cdot|\mathrm{P}_1 = \rho_1,\ldots,\mathrm{P}_m = \rho_m)$. In equation (4.8), the essential infimum is used, as, in the general case, there may be an uncountable stopping rules set $M \geq m$, and the infimum of an uncountable range of random variables might not be measurable [106]. Therefore, (4.8) means that for all $M \geq m$: $R(W_m^*(\rho_1,\ldots,\rho_m) \leq \mathrm{E}_m(L_m)) = 1$, and if Q is any another random variable for which $\forall M \geq n$, then $R(Q \leq \mathrm{E}_m(L_m)) = 1$, то $R(Z \leq W_m^*(\rho_1,\ldots,\rho_m)) = 1$.

An optimality principle assumes that a decision to stop at the moent m is optimal if and only if the loss at that step is equivalent to the lowest anticipated loss value for all stopping rules that reach step m. The relationship among the lowest anticipated loss value for all stopping rules $M \geq m$, and for all stopping rules that do not conclude at moment m (i.e., reaching step $m+1$) can be expressed by the following optimality equation:

$$W_m^* = \min\left\{L_m, \mathrm{E}_m\left(W_{m+1}^*\right)\right\}. \quad (4.9)$$

Using the statements in (4.7), it can be proved [106] that equality (4.9) is satisfied, ensuring the optimality of the following stopping rule:

$$M^* = \min\left\{m \geq 0 : L_m \leq \mathrm{E}_m\left(W_{m+1}^*\right)\right\}. \quad (4.10)$$

That is to say, an optimality principle outlines the rule (4.10), which stops the recognition process at the first step m such that the loss at this step does not

exceed the Anticipated loss value for any stopping rule that reaches step m. This ensures that the process is stopped at the point where the anticipated penalty is minimized, thus optimizing the recognition process.

4.3.2 Monotonic Stopping Problems

A special class of stopping problems, known as monotonic stopping problems [106; 116], is specified as follows. Suppose B_m represent the occurrence $\left\{L_m \le \mathrm{E}_m\left(L_{m+1}\right)\right\}$. A stopping problem is called monotonic if the following condition holds:

$$B_0 \subset B_1 \subset B_2 \subset \ldots \tag{4.11}$$

A condition of (4.11) implies that if at certain step m the loss function value is less than or equal to the anticipated loss at the following step, this will be valid for all following steps.

Consider the following stopping rule, which is called the single-step look-ahead rule:

$$M_B = \min\left\{m \ge 0 : L_m \le \mathrm{E}_m\left(L_{m+1}\right)\right\}. \tag{4.12}$$

The rule (4.12) stops the process at step m if the ongoing value of the loss function does not exceed the loss at the stopping step $m+1$. It can be demonstrated [106; 116] that if a stopping problem has a limited horizon (i.e., for a certain $D < \infty$, all stopping rules are required to stop at step D) and it is monotonic, then the single-step look-ahead rule (4.12) is optimal. In other words, under these conditions, the single-step look-ahead rule (4.12) minimizes the anticipated loss by halting the process at the first step where continuing would result in a higher anticipated penalty, making it the optimal stopping strategy for monotonic problems with a finite horizon.

To create the stopping rule for the process of object recognition in a real-time video stream, the following section will describe the conditions under which

the problem can be examined monotonic, no less than starting from some step, and will present a stopping rule that simulates the behavior of the single-step look-ahead rule (4.12).

4.4 Proposed Method

We formulate the following requirement for the integration functions $Y^{(m)}$: the Anticipated range among two sequential integrated results of recognition remains stable over time:

$$\mathrm{E}\left(l\left(Y_m, Y_{m+1}\right)\right) \geq \mathrm{E}\left(l\left(Y_{m+1}, Y_{m+2}\right)\right),\ \forall m > 0\,. \tag{4.13}$$

Within the scope of "anytime" algorithms [102], the requirement (4.13) implies that the problem shows the characteristic of diminishing returns. Based on this supposition concerning the integration functions $Y^{(m)}$, it can be shown that a stopping problem (4.6) with a loss function (4.2) turns into monotonic from a specific step onward.

Actually, suppose U_m represent occurrence $\left\{\mathrm{E}_m\left(l\left(Y_m, Y_{m+1},\right)\right) \leq k\right\}$, and examine the stopping problem (4.6) beginning at the step m, where occurrence U_m first took place. The occurrence B_m regarded in the monotonicity state (4.11) be presented in the following way:

$$\begin{aligned} B_m : &\left\{l\left(Y_m, \mathrm{P}^*\right) + km \leq \mathrm{E}_m\left(l\left(Y_{m+1}, \mathrm{P}^*\right)\right) + km + c\right\} = \\ &= \left\{l\left(Y_m, \mathrm{P}^*\right) - \mathrm{E}_m\left(l\left(Y_{m+1}, \mathrm{P}^*\right)\right) \leq k\right\}. \end{aligned} \tag{4.14}$$

For a fixed P^*, at step m, using the triangle inequality, we can acquire the following relation among the range from the ongoing result of recognition to the true value, the anticipated range to the result at the following step, and the anticipated range from the following result to the true value:

$$\begin{aligned} & l\left(Y_m, \mathrm{P}^*\right) \leq \mathrm{E}_m\left(l\left(Y_m, Y_{m+1}\right)\right)+\mathrm{E}_m\left(l\left(Y_{m+1}, \mathrm{P}^*\right)\right) \Rightarrow \\ & \Rightarrow l\left(Y_m, \mathrm{P}^*\right)-\mathrm{E}_m\left(l\left(Y_{m+1}, \mathrm{P}^*\right)\right) \leq \mathrm{E}_m\left(l\left(Y_m, Y_{m+1}\right)\right). \end{aligned} \tag{4.15}$$

This inequality provides a bound on the ongoing range, showing that the anticipated future improvement in recognition accuracy is at least partially determined by the change in range at the following step.

If the right term of the inequality holds in (4.15) does not exceed the non-negative constant k, then the left term also does not exceed k, and consequently, if occurrence U_m occurs, occurrence (4.14) must also occur. Moreover, using the assumption (4.13), we can infer that if occurrence U_m occurs, then occurrence U_{m+1} must also occur. Thus, we obtain:

$$\forall m>0: U_m \subset B_m,\ U_m \subset U_{m+1}. \tag{4.16}$$

From this, it follows that starting from step m, where occurrence U_m occurred for the first time, occurrence s $B_m, B_{m+1}, B_{m+2}, \ldots$ will also occur. Therefore, the stopping problem can be considered monotonic starting from this step, which implies the optimality of rule (4.12) within all stopping rules that attain step m in the case where the problem has a finite horizon.

Now, let us examine a stopping rule that instructs the determiner to stop the recognition process if occurrence U_m occurs:

$$M_U=\min\left\{m>0: \mathrm{E}_m\left(l\left(Y_m, Y_{m+1}\right)\right)\right\}. \tag{4.17}$$

This rule suggests that the recognition process is terminated when a specific occurrence , denoted as U_m, happens during the observation sequence.

If rule (4.17) requires stopping at step m, then rule M_B will also require stopping at this step. Since the problem becomes monotonic starting from step m, the decision of rule M_B is optimal, and therefore, the optimal rule M^* will also

require stopping at this step. Moreover, if $l\left(Y_m, \mathrm{P}^*\right) - \mathrm{E}_m\left(l\left(Y_{m+1}, \mathrm{P}^*\right)\right) > k$ occurs, rule M_B does not stop the process, just like rule M^*, which follows the principle of optimality. Therefore, if assumption (4.13) holds true, rule M_B will never stop prematurely, and if the rule dictates stopping, the decision to stop is optimal.

The figure 4.1 graphically shows the similarities and differences among stop rules M_U and M^* for various relationships among occurrence s B_m and U_m. The set of situations where at M_B the process continues but occurrence M^* may stop the process is determined by key limitations of rule M_B:

1. Rule is determined by the estimate of the disparity in the values of loss function applying the triangle inequality, hence, it is ineffective in cases where the accuracy of integrated recognition results worsen as time passes (namely, if $l\left(Y_m, \mathrm{P}^*\right) - \mathrm{E}_m\left(l\left(Y_{m+1}, \mathrm{P}^*\right)\right) < 0$).

2. The decision is made by setting a limit on the anticipated value of the metric, that in general may not be bounded above.

These limitations suggest that the rule may not be suitable in cases where the system's performance declines over time or where the anticipated metric has no upper limit.

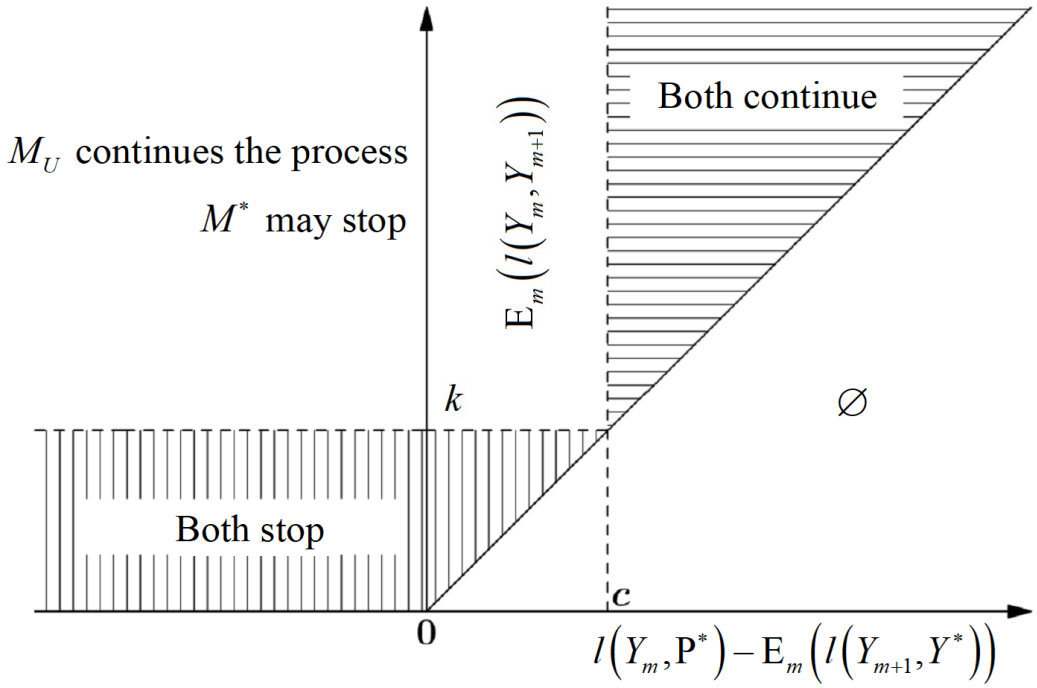


Figure 4.1 – The difference in the behaviors of the presented stopping rule M_U (based on the estimation of the anticipated range from the ongoing integrated recognition result to the following) and an optimal stopping rule M^*.

In the proposed method, we will suppose that a metric defined on the all possible object recognition results set is upper-bounded (i.e., $\exists G : \forall \rho_i, \rho_j \in P : 0 \le l\left(\rho_i, \rho_j \le G\right)$) and that the integration functions $Y^{(m)}$ produce results that, on mean, do not deteriorate over time:

$$\mathrm{E}\left(l\left(Y_m, \mathrm{P}^*\right)\right) \ge \mathrm{E}\left(l\left(Y_{m+1}, \mathrm{P}^*\right)\right), \ \forall m > 0 . \tag{4.18}$$

Thus, to solve the stopping problem (4.6) with the loss functional (4.2), the following method is proposed:

1. Estimate the anticipated range (in terms of the metric l) from the ongoing integrated object recognition result Y_m (known at step m) to the unknown result Y_{m+1} at the following step;

2. Make a decision to stop the process at step m by thresholding the range estimated in step 1, thereby approximating the behavior of the rule M_U.

Generally, the choice of a method for forecasting the following integrated object recognition result (or estimating the anticipated range among it and the ongoing integrated result) could rely on the character of the integration functions $Y^{(m)}$ and other specific characteristics of the problem.

Based on the proposed method, let us construct a stopping algorithm for the recognition process of a string object. Suppose the functions for integrating the recognition results of the string object $Y^{(m)}$ are given (which can be implemented using the ROVER method or Algorithm 1). As the metric l for string objects, it is proposed to use the normalized generalized Levenshtein range [134]. In order to approximate the behavior of the stopping rule M_U, at the m-th step of the process, it is necessary to compute the estimate of the anticipated range among adjacent integrated recognition results $\Delta_m \overset{def}{=} \mathrm{E}_m\left(l\left(Y_m, Y_{m+1}\right)\right)$, having access to the observations $\mathrm{P}_1 = \rho_1, \ldots, \mathrm{P}_m = \rho_m$. To calculate the estimate, it is proposed to simulate the following integrated result, considering that a new observation will be near to those obtained in the previous steps:

$$\Delta_m \overset{def}{=} \frac{1}{m+1}\left(\xi + \sum_{i=1}^{m} l\left(Y_m, Y\left(\rho_1, \rho_2, \ldots, \rho_m, \rho_i\right)\right)\right), \tag{4.19}$$

where ξ is an external adjustable parameter.

In pseudocode form, the stopping algorithm for the string object recognition process is presented as Algorithm 2. When using the recognition result model for string objects with alternative classification options for individual objects, as discussed in Chapter 3, the upper bound for the lengths of the integrated results Y_m and Y_{m+1} is $O(Im)$, where $I = \max_{i=1}^{m} \left|\mathrm{P}_i\right|$. Since the computational complexity of directly calculating the generalized Levenshtein range among the strings P_i and P_j is $O\left(\left|\mathrm{P}_i\right|\left|\mathrm{P}_j\right|N\right)$, where N is the number of classes into which each individual object is classified, the computational complexity of Algorithm 2 is $O\left(I^2 m^3 N\right)$. It

should be noted that the computational complexity of the algorithm can be reduced either by using simplified recognition result models for string objects or by employing heuristic algorithms for approximate computation of the generalized Levenshtein range.

Algorithm 2 – Stopping decision algorithm for the string object recognition process based on obtained observations $\mathrm{P}_1,\mathrm{P}_2,\ldots,\mathrm{P}_m$, their weights $\omega_1,\omega_2,\ldots,\omega_m$, and external parameters ξ and k.

Input: $m>0,\ \forall i\in\{1,\ldots,m\}:|\mathrm{P}_i|>0$

1. $Y_m \leftarrow Y^{(m)}\left(\mathrm{P}_1,\mathrm{P}_2,\ldots,\mathrm{P}_m,\omega_1,\omega_2,\ldots,\omega_m\right)$ /*The integrated result at step m */

2. $Z_m \leftarrow \sum_{i=1}^{m}\omega_i$ /*The total weight of observations at step m */

3. $\hat{\Delta}_m \leftarrow \xi$

4. for $i=1$ to m do

5. $Y_{m+1} \leftarrow Y^{(2)}\left(Y_m,\mathrm{P}_i,Z_m,\omega_i\right)$

6. $\hat{\Delta}_m \leftarrow \Delta_m + l_L\left(Y_{m+1},Y_m\right)$

7. end for 4.

8. $\hat{\Delta}_m \leftarrow \Delta_m/(m+1)$

9. if $\hat{\Delta}_m \le k$ then

10. return *Stopping*

11. else

12. return *Resume process*

13. end if 9.

14. end

4.5 Experimental Results

This section presents the results of the experimental study of the object recognition stopping method in a real-time video stream, as described in Section

4.4. The object of recognition is a text aria of an identity certificate. In the simplest case, the result of recognizing the text aria (an element of set P) can be represented as a string over a certain fixed alphabet.

To apply the model described in Section 4.2, it is necessary to define the metric l and the integration functions $Y^{(m)}$ for the set of strings P. As a metric on the set of strings, it is proposed to use the normalized Levenshtein range, as in the experimental study presented in Chapter 3 (see Section 3.5) [134]:

$$l_L\left(\rho_i,\rho_j\right) \overset{def}{=} \frac{2levenshtein\left(\rho_i,\rho_j\right)}{\left|\rho_i\right|+\left|\rho_j\right|+levenshtein\left(\rho_i,\rho_j\right)}, \tag{4.20}$$

where $\left|\rho_i\right|$ is the string ρ_i length, and $levenshtein\left(\rho_i,\rho_j\right)$ is a Levenshtein range among strings ρ_i and ρ_j. This metric values lie within the range [0, 1], and its normalization is performed while preserving the triangle inequality.

As integration functions $Y^{(m)}$, the ROVER algorithm [87] was used, the description of which is provided in Section 3.3. To implement the method, it is necessary to introduce a threshold λ for evaluating the empty class, which is considered in the voting module. In the experiments conducted, as in the experimental part of Chapter 3 (see Section 3.5), the threshold value $\lambda = 0.6$ was used.

The experimental study was conducted on the open MIDV-500 dataset [88], which holds 50 different types of identity certificates videos (with 10 video clips for each certificate; 30 frames per video) with explained ideal locations and content of text areas. Four groups of arias were analyzed: dates recorded with numbers and punctuation marks, Machine-Readable Zone (MRZ) strings, certificate number, and elements of the certificate holder's name written in the Latin alphabet.

Only frames where the certificate was completely were regarded (consequently, the video sequences in the analyzed dataset subset had varying lengths, ranging between 1 frame up to 30 frames). To provide a clearer

representation of the results and minimize normalization effects, each video clip was lengthened up to 30 frames by repeating it from the beginning. Thus, all analyzed clips had a uniform length of 30 frames.

Each aria was clipped from the initial image using a projective transformation, based on the combined annotation of the ideal certificate boundaries and the coordinates of the text aria, with added margins matching 15% of the shortest side of the text aria. The size of the cropped text aria images matched to the 300 dots per inch resolution. Each cropped text aria was recognized applying the Free Tesseract recognition software (versions v4.1.1 and v5.5.0) [138] with default parameters for the English. Every character value comparisons were case- neutral, and the the digit "0" was considered identical to Latin letter "O."

Table 4 provides, for all group of text arias, a number of unique arias in the MIDV-500 dataset, the entire number of text aria images (over all frames where the certificate is fully visible), and the mean length of the frame sequence. The table also presents the mean range P_i among the recognition result for a single frame and the ground truth value P^*, as well as among the integrated result for the video clip and the ground truth value P^*, both before (Y_{fin}) and after (Y_{30}) padding in terms of the metric defined in (4.20).

Fig. 4.3 demonstrates the decrease in the difference among the ranges from adjacent integrated recognition results to the ground truth value $\mathrm{E}\left(l\left(Y_m,\mathrm{P}^*\right)\right)-\mathrm{E}\left(l\left(Y_{m+1},\mathrm{P}^*\right)\right)$ over time, the mean range Δ_m among adjacent integrated recognition results, and its estimation (4.19). In the conducted experiments, the configurable parameter value $\xi=0.2$ from (4.19) was used. Although the stopping rule (4.17) is a rough approximation of the single-step look-ahead rule (4.12), both the practical justification for assumption (4.13) and the justification for estimation (4.19) starting from step $m=2$ can be noted for the purposes of the given task. The stopping rule approximation M_U will now be constructed using the threshold cut-off of the estimate $\hat{\Delta}_m$.

Table 4 – Mean Values of Metric l_L to Ground Truth for Recognition Results Using the Tesseract Library [138] for Text Arias of the MIDV-500 Dataset [88]: P_i - recognition result for a single frame, Y_{fin} - integrated recognition result for the video clip obtained using a modified ROVER algorithm, Y_{30} - integrated recognition result for the padded video clip obtained using a modified ROVER algorithm.

	Document number	Date	Name	MRZ sring	All arears
Unique arears	48	91	79	30	248
Total number of clips	436	824	719	260	2239
Total number of images	9329	17735	15587	5096	47747
Mean clip length	21.397	21.523	21.679	19.600	21.325
Tesseract v4.1.1					
$\mathrm{E}\left(l_L\left(\mathrm{P}_i,\mathrm{P}^*\right)\right)$	0.422	0.360	0.443	0.258	0.388
$\mathrm{E}\left(l_L\left(Y_{fin},\mathrm{P}^*\right)\right)$	0.326	0.244	0.338	0.162	0.281
$\mathrm{E}\left(l_L\left(Y_{30},\mathrm{P}^*\right)\right)$	0.323	0.246	0.336	0.164	0.280
Tesseract v5.5.0					
$\mathrm{E}\left(l_L\left(\mathrm{P}_i,\mathrm{P}^*\right)\right)$	0.287	0.238	0.250	0.339	0.262
$\mathrm{E}\left(l_L\left(Y_{fin},\mathrm{P}^*\right)\right)$	0.160	0.123	0.125	0.277	0.149
$\mathrm{E}\left(l_L\left(Y_{30},\mathrm{P}^*\right)\right)$	0.163	0.125	0.127	0.279	0.151

Fig. 4.2 illustrates the mean ranges, based on metric (4.20), among the integrated recognition results of a text string in the real-time video stream and the ground truth value, with all text arias analyzed. It can be observed that the error decreases significantly over time for all groups of arias, which can be considered practical evidence supporting assumption (4.18).

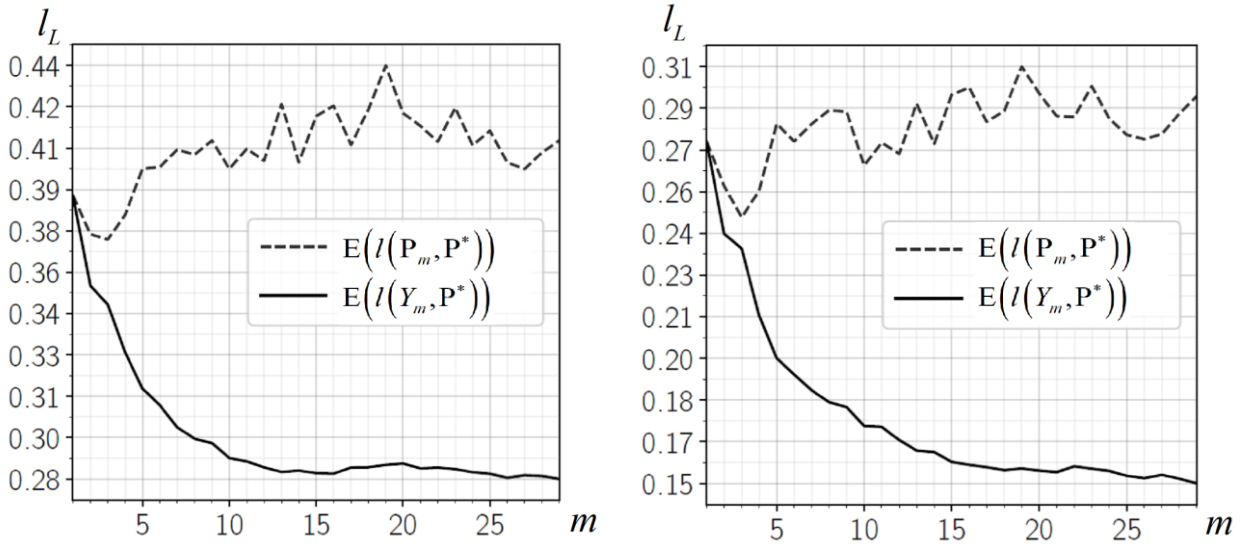


Figure 4.2 – Mean ranges from the frame-by-frame recognition result of the text string and from the integrated recognition result in the real-time video stream to the ground truth value, based on metric (4.20). Text aria recognition was performed using the Tesseract library v4.1.1 (left) and v5.5.0 (right).

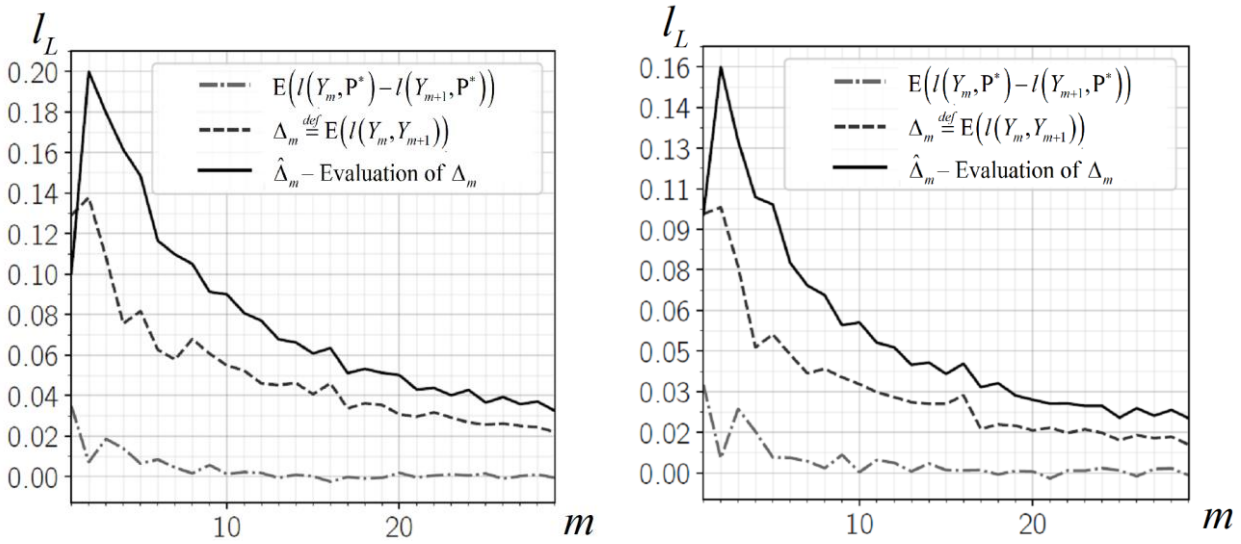


Figure 4.3 – Reduction in the mean range among adjacent integrated recognition results and its assessment, with the configurable parameter value $\xi = 0.2$. Text aria recognition was conducted applying the Tesseract library v4.1.1 (left) and v5.5.0 (right).

To evaluate the effectiveness of the stopping rule, an efficiency profile can be created, visually showing the relationship among the maen number of analyzed

observations and the corresponding mean range from the obtained integrated result at the stopping point to the ground truth value, while varying the observation cost k. Such an efficiency profile reflects the trade-off among the required time for video sequence processing and the accuracy of the obtained recognition result, as well as allows for a visual comparison of different stopping strategies.

The control rule used was a simple counting rule M_N, which requires stopping the recognition process at a set step N. Furthermore, two variants of a stopping rule, described in [139], were investigated. Considering the original work depends on the use of assurance metrics for the recognition results, which are not available within the framework of the model studied in this chapter, the stopping rule described in [139] degenerates into a threshold cut-off of the size of the largest cluster of matching results of recognition collected up to step m. Therefore, two control stopping rules were constructed:

- M_{KP}, which performs a threshold cut-off of the size of the largest cluster of matching frame-by-frame results of recognition $\rho_1,\ldots,\rho_m$,

- M_{KY}, which similarly considers the integrated recognition results $Y_1,\ldots,Y_m$.

Finally, the stopping rule (4.17), developed in current chapter, assessments at each step the anticipated range Δ_m to the following integrated result and stops the process when this estimate becomes less than or matching a threshold. A stopping rule M_U applies just starting from step $m=2$ (i.e., from the step where the estimation in (4.19) becomes more justified).

Fig. 4.4 illustrates the effectiveness of the stopping rules for all groups of text arias recognized using the Tesseract library (versions v4.1.1 and v5.5.0). A lower position of the curve reflects greater effectiveness of the stopping rule. It is worth noting that, on average, the presented stopping rule (4.17) demonstrates higher effectiveness compared to other examined methods. It is worth mentioning that the stopping method (4.17) exhibits high effectiveness without any modifications for the two different Tesseract softwore versions, that use varied generations of text string recognition algorithms.

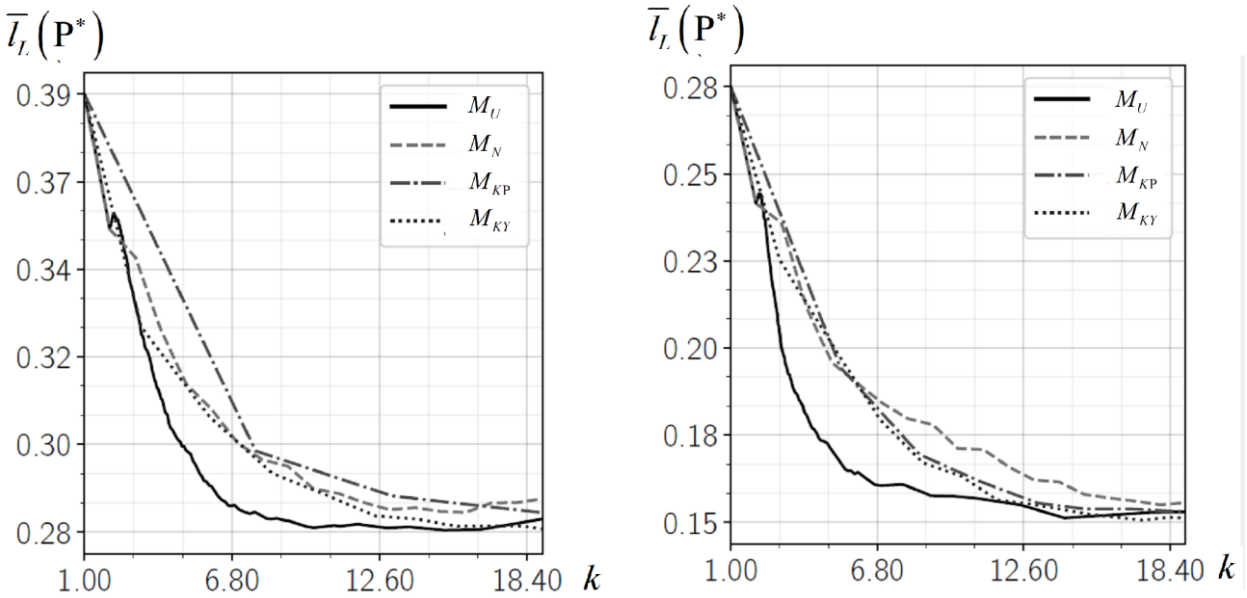


Figure 4.4 – Comparative study of the effectiveness of stopping rules: a graph showing the relationship among the mean range among the obtained result at the stopping point and the true value, and the mean number of analyzed frames before stopping, with varying observation cost k, at the configurable parameter value $\xi = 0.2$. The recognition of text arias was performed using the Tesseract library v4.1.1 (left) and v5.5.0 (right).

Table 5 presents the mean range from the integrated result to the accurate answer at the stopping point, that can be accomplished with investigated stopping rules, for text aria recognition with the Tesseract v4.1.1 library. The columns of Table 5 reflect the expected ranges for the values of the mean number of observations used (i.e., the mean number of analized frames), the rows correspond to a stopping rules, and every cell contains the measurement result with a smallest mean number of observations falling into the given range. Some cells in the table contain no data (marked with the symbol $\varnothing$) – this indicates that the corresponding stopping rule is unable to reach the mean number of processed frames within the corresponding range for the given dataset (because of its more discrete nature). It follows that, in almost every target ranges, a stopping rule (4.17) demonstrates the best result among the investigated alternatives. A similar result is

observed for text aria recognition using the Tesseract v5.5.0 library (the results are presented in Table 6).

Table 5 – Achieved values of the mean range from the integrated result to the ideal value at the stopping point, in terms of the metric l_L, with recognition performed using Tesseract v4.1.1.

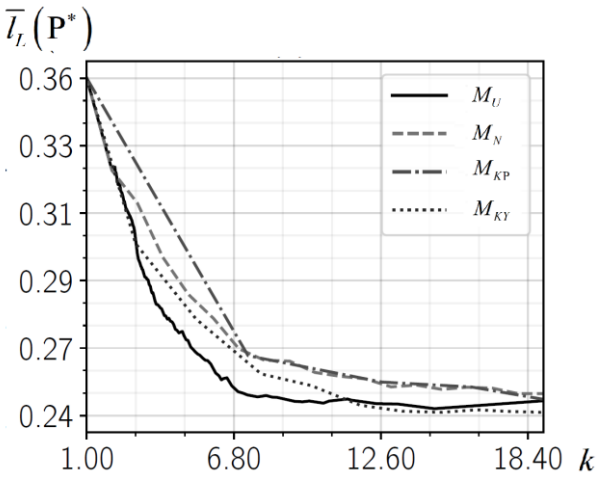
Stopping rule	Measured parameter	Target interval of the average number of observations $\mathrm{E}(M)$								
		3 ± 0.5	4 ± 0.5	5 ± 0.5	6 ± 0.5	7 ± 0.5	8 ± 0.5	9 ± 0.5	10 ± 0.5	11 ± 0.5
M_{KP}	$\mathrm{E}(M)$	∅	∅	∅	∅	∅	7.727	∅	∅	∅
	$\mathrm{E}\left(l_L\left(Y_M,\mathrm{P}^*\right)\right)$						0.298			
M_{KY}	$\mathrm{E}(M)$	3.230	∅	∅	5.909	∅	8.375	∅	∅	**10.560**
	$\mathrm{E}\left(l_L\left(Y_M,\mathrm{P}^*\right)\right)$	**0.329**			0.306		0.292			0.286
M_C	$\mathrm{E}(M)$	3.000	4.000	5.000	6.000	7.000	8.000	9.000	10.000	11.000
	$\mathrm{E}\left(l_L\left(Y_M,\mathrm{P}^*\right)\right)$	0.347	0.329	0.315	0.308	0.300	0.295	0.294	0.288	0.287
M_U	$\mathrm{E}(M)$	**2.509**	**3.558**	**4.527**	**5.521**	**6.531**	**7.691**	**8.554**	**9.715**	10.830
	$\mathrm{E}\left(l_L\left(Y_M,\mathrm{P}^*\right)\right)$	0.351	**0.322**	**0.302**	**0.292**	**0.285**	**0.282**	**0.280**	**0.278**	**0.278**

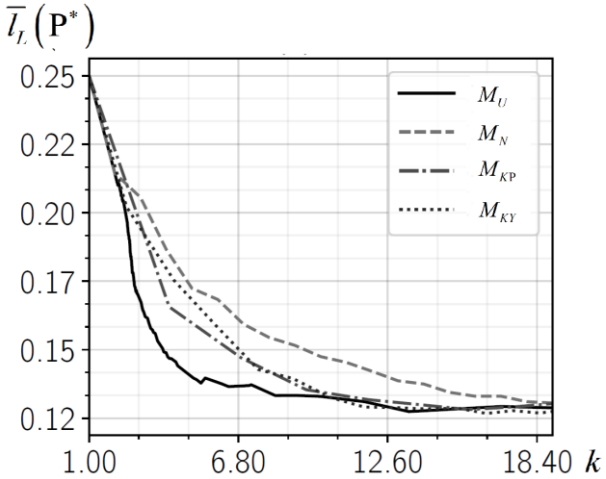
Table 6 – Achieved values of the mean range from the integrated result to the ideal value at the stopping point, in terms of the metric l_L, with recognition performed using Tesseract v v5.5.0.

Stopping rule	Measured parameter	Target interval of the average number of observations $\mathrm{E}(M)$								
		3 ± 0.5	4 ± 0.5	5 ± 0.5	6 ± 0.5	7 ± 0.5	8 ± 0.5	9 ± 0.5	10 ± 0.5	11 ± 0.5
M_{KP}	$\mathrm{E}(M)$	∅	∅	5.332	∅	∅	8.471	∅	∅	10.901
	$\mathrm{E}\left(l_L\left(Y_M,\mathrm{P}^*\right)\right)$			0.195			0.170			0.162
M_{KY}	$\mathrm{E}(M)$	2.936	∅	5.099	∅	6.920	∅	**8.594**	10.103	∅
	$\mathrm{E}\left(l_L\left(Y_M,\mathrm{P}^*\right)\right)$	0.227		0.201		0.180		0.167	0.164	
M_C	$\mathrm{E}(M)$	3.000	4.000	5.000	6.000	7.000	8.000	9.000	10.000	11.000
	$\mathrm{E}\left(l_L\left(Y_M,\mathrm{P}^*\right)\right)$	0.237	0.213	0.197	0.191	0.185	0.180	0.178	0.171	0.171
M_U	$\mathrm{E}(M)$	**2.580**	**3.551**	**4.571**	**5.539**	**6.683**	**7.742**	8.771	**9.779**	**10.726**
	$\mathrm{E}\left(l_L\left(Y_M,\mathrm{P}^*\right)\right)$	**0.224**	**0.188**	**0.174**	**0.165**	**0.161**	**0.161**	**0.158**	**0.158**	**0.157**

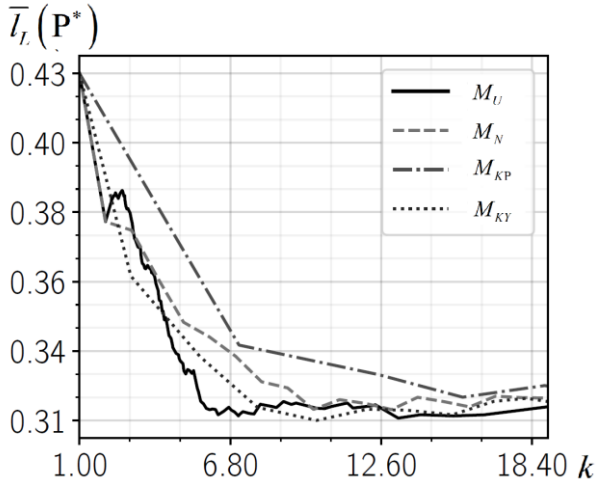
The efficiency profiles of the stopping rules for individual aria groups (Date, Certificate Number, MRZ Line, Name (Latin)) are presented in Fig. 4.5.

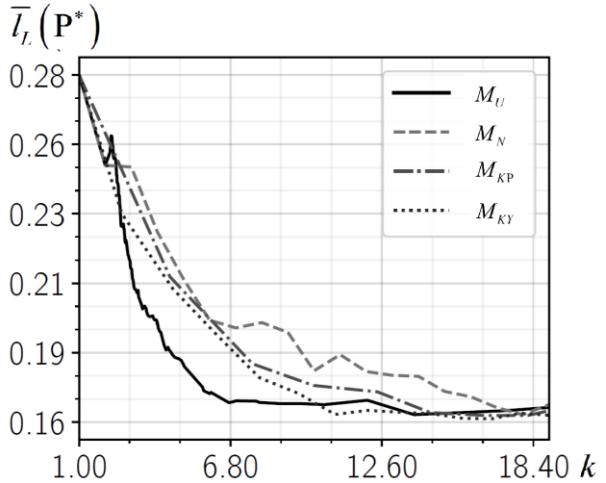
Date



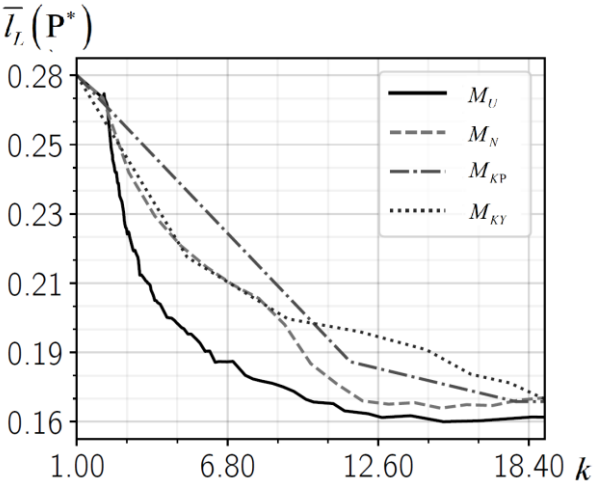


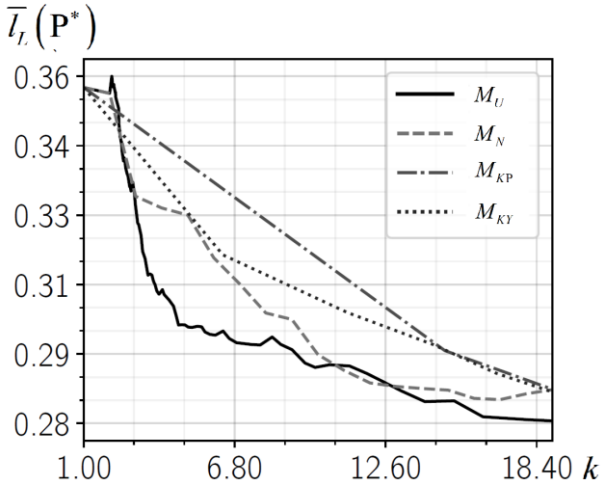
Document number





MRZ lines





Name

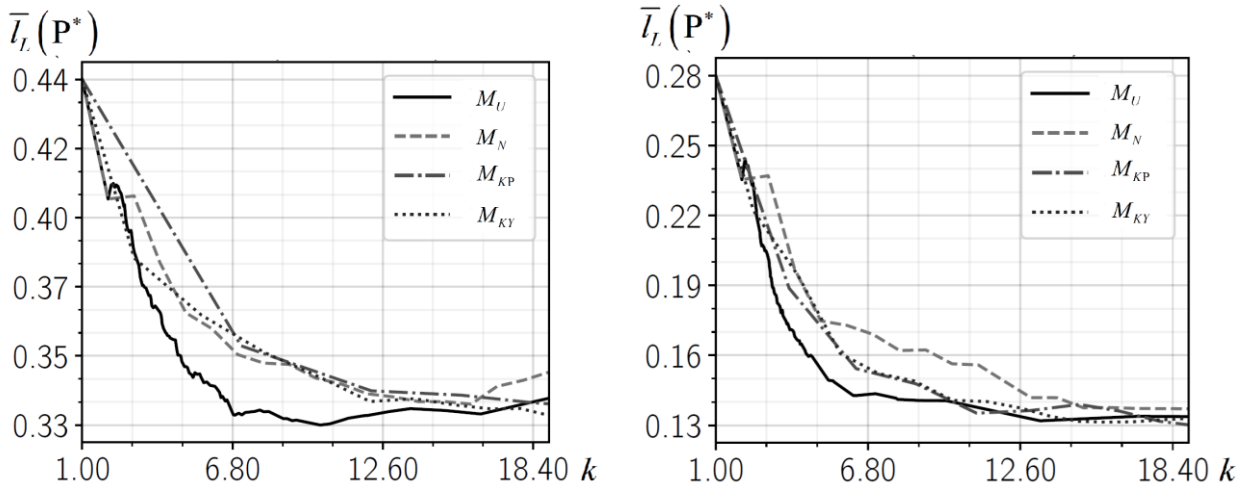


Figure 4.5 – Efficiency profiles of the stopping rules for different aria groups. Text aria recognition was performed using Tesseract v4.1.1 (left) and v5.5.0 (right).

4.6 Framework for Real-Time Recognition of Symbolic Objects in Video Streams

The technological component of the proposed system serves to consolidate and implement the mathematical models and algorithms developed throughout the dissertation (Sections 2–4) in a functioning software environment capable of real-time symbolic object recognition in video streams. This implementation is not merely demonstrative; rather, it substantiates the feasibility and practical applicability of the proposed methods under real-world constraints. As discussed in Section 1.3.7, recognition in video streams requires integration of results from multiple frames, while Section 2.2 formalises a dynamic model with an internal system state and discrete-time frame acquisition. Section 3 presents a new algorithm for integrating alternative classification hypotheses (Subsection 3.4), which improves result consistency, and Section 4 proposes a novel stopping method based on threshold reduction in expected distance (Subsection 4.4), allowing the system to adapt processing duration dynamically. The role of the technological implementation is thus threefold: (i) to validate the models and methods proposed in Chapters 2–4 through empirical evaluation (Subsections 3.5 and 4.5), (ii) to demonstrate compatibility with resource-constrained environments

such as mobile devices (Section 4.6), and (iii) to provide a modular, extensible architecture suitable for integration with real-world document recognition systems.

The architecture of the implemented system reflects the conceptual structure of the dynamic recognition model introduced in Subsection 2.2 and further expanded through algorithmic developments in Chapters 3 and 4. The system is organised as a modular pipeline capable of processing video input in real time, accumulating recognition hypotheses, and determining the optimal point at which to halt processing. As shown schematically in Figure 2.1 (right), the general architecture comprises several functional components:

- Video Acquisition Module, responsible for capturing the video stream and extracting individual frames. This module is synchronised with a timing controller that ensures compatibility with real-time constraints and limits frame skipping in line with the system model described in Subsection 2.2.

- Preprocessing Module, which applies primary transformations such as grayscale conversion, noise reduction, and geometric correction, in accordance with methods outlined in Subsections 1.3.1 and 1.3.2. Adaptive binarisation and projection-based segmentation techniques (Subsection 1.3.3) are also integrated here.

- Symbolic Object Recognition Core, which implements character-level recognition using convolutional neural networks (CNNs), as detailed in Subsection 1.3.4. Each character is processed with alternative classification outputs retained as part of the probabilistic hypothesis space.

- Result Integration Engine, where the algorithm proposed in Subsection 3.4 is realised. This block accumulates recognition results across frames, maintaining competing hypotheses per symbol and applying weighted aggregation in accordance with the confidence-based logic derived from Formula (3.5) and the experimental findings of Subsection 3.5.

- Stopping Criterion Module, responsible for evaluating the expected distance between the current integrated result and a prospective next state. As developed in Subsection 4.4, this module operates on the principle of monotonic

expected gain reduction and is executed after each frame to assess termination readiness.

- Output and Logging Interface, which formats the final recognised string, tracks the number of processed frames, timestamps, and diagnostic data. These outputs support performance analysis and validation, particularly during test phases described in Appendices A and B.

Each component communicates via a defined API interface within a loosely coupled architecture, facilitating modular updates and substitution of algorithms (e.g., switching the recognition core from Tesseract to a deep learning-based model). The general structure and interaction logic of the software modules described above are summarised in Fig. 4.6. The diagram outlines the functional architecture of the implemented system, highlighting the key processing stages (from video acquisition to integration and stopping) along with references to the relevant subsections of the dissertation where each module is detailed. This visual representation illustrates how theoretical components have been operationalised into a modular, real-time recognition framework.

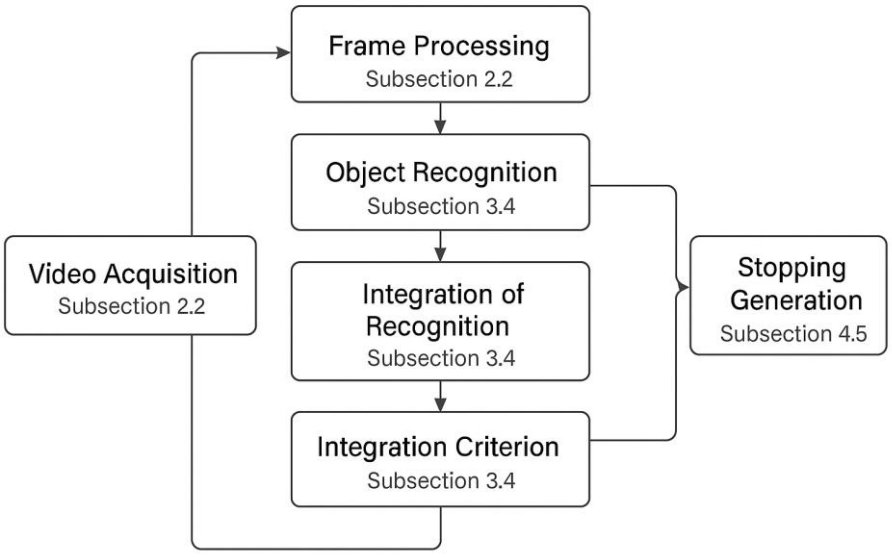


Figure 4.6 – Software architecture of the real-time symbolic object recognition system.

The implementation of the system for real-time recognition of symbolic objects in video streams was carried out using a cross-platform development environment, ensuring compatibility with both desktop and embedded systems. The selection of tools and languages was guided by the need for efficient processing of video data, support for deep learning frameworks, and flexible integration with open-source recognition engines.

The core modules of the system were developed in Python 3.10, due to its wide support for machine learning libraries, concise syntax, and portability. Computationally intensive operations, such as frame-level parallelism and matrix transformations, were optimised using NumPy and OpenCV (v4.8), the latter of which facilitated image acquisition, transformation, and region-of-interest extraction as required in Subsections 1.3.1–1.3.3.

Symbolic object classification was realised through two complementary approaches. The baseline implementation uses the Tesseract OCR engine (v5.3) with custom-trained language models for high-speed processing. For research-oriented testing, particularly in Subsections 3.5 and 4.5, a deep learning-based recogniser was implemented using TensorFlow 2.x with a CNN+LSTM architecture to validate the system's compatibility with alternative recognition pipelines. Batch training of these models used synthetically generated data in accordance with the methodology outlined in Subsection 3.6.

The integration and stopping algorithms (Subsections 3.4 and 4.4) were implemented as custom Python modules, with numerical procedures for confidence scoring, distance calculation, and threshold truncation coded using SciPy. Bayesian estimators and weighted voting mechanisms were verified against empirical data from the MIDV-500 dataset (see Figure 3.5 and Tables 5–6).

Software execution was tested on machines running Ubuntu 22.04 LTS and Windows 10, with support for GPU acceleration via CUDA 11.8 when applicable. All components were executed within a managed virtual environment using Anaconda, ensuring reproducibility and version control.

For user interaction and batch testing, a lightweight GUI prototype was built using Tkinter, while logging, visualisation of frame-level decisions, and experimental results were output in CSV and Matplotlib formats. The system supports deployment via Docker containers, facilitating portability to edge devices or cloud-based microservices. To consolidate the overview of the programming environment, software libraries, and execution platforms used for the system implementation, Table 7 provides a structured summary of the complete technology stack. It outlines the key components, their respective roles in the system, and the environments in which they were deployed, offering a concise reference to the development infrastructure that supports the real-time recognition framework.

Table 7 – Technology stack and deployment environment of the implemented system

Component / Module	Technology Used	Version	Purpose / Notes
Frame acquisition	OpenCV	4.8	Video stream capture, preprocessing
OCR engine (baseline)	Tesseract OCR	5.3	Fast symbolic recognition
OCR engine (research)	TensorFlow + CNN-LSTM	2.x	Deep recognition pipeline for comparison
Integration & stopping module	Python + NumPy + SciPy	3.10	Custom implementation of algorithms
UI / CLI interface	Tkinter / argparse	–	Control and visualisation
Deployment	Docker, Raspberry Pi 4B	–	Containerisation and edge deployment

The core algorithms developed in Chapters 3 and 4 were implemented as discrete software modules, each corresponding to specific components of the system pipeline. Their integration follows the architectural logic outlined in Subsection 2.2 and realises the proposed dynamic recognition model for symbolic objects in real-time video streams.

The algorithm for integration of recognition results (described in Subsection 3.4 and formalised in Formula (3.5)) was implemented using a cumulative voting scheme over a hypothesis matrix. Each row of the matrix corresponds to a character position in the string object, while each column represents a possible

classification hypothesis obtained from different frames. Unlike standard majority voting, the algorithm applies confidence-weighted accumulation of hypotheses, supporting updates from both strong and weak recognisers. This matrix structure allows the system to retain alternative interpretations of each symbol and progressively refine the aggregated result as new frames are processed.

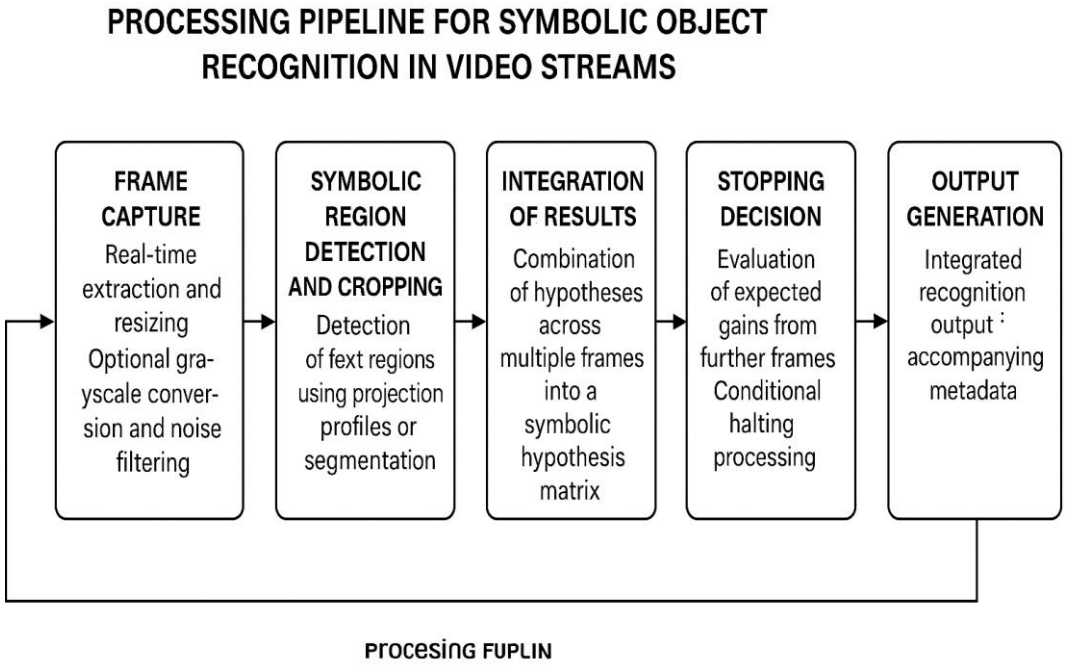
To manage symbol-level ambiguity, each recognition result from Tesseract or a CNN-based recogniser is parsed into a ranked list of alternatives with associated confidence scores. These alternatives are stored and updated using Bayesian estimators (as discussed in Subsection 3.3), enabling consistent re-evaluation of hypotheses upon new observations. The algorithm performs both frame-level voting and confidence recalibration, which enhances robustness under noisy or low-quality conditions, as reflected in the performance improvements reported in Subsection 3.5.

The stopping criterion algorithm (developed in Subsection 4.4) was implemented using the monotonic stopping model formalised in Formula (4.5). After each frame, the expected gain of acquiring the next observation is computed as the predicted reduction in distance between the current and prospective integrated result. This expected distance is approximated using a metric over the weighted hypothesis space (based on KL-divergence or edit distance in normalised space). If the expected gain falls below a predefined threshold—empirically tuned on MIDV-500 data—the algorithm halts processing and outputs the current result.

The system ensures interruptibility by design, in accordance with the "anytime" algorithm paradigm discussed in Subsection 4.1. This property allows termination at any iteration without loss of validity, making the implementation suitable for constrained environments or adaptive quality scenarios. Together, these core algorithms form a flexible and effective decision pipeline, capable of aggregating noisy or partially correct frame-level predictions into a stable, accurate result. Their correctness and efficiency were validated through comprehensive experimentation described in Subsections 3.5 and 4.5, and the implementation

remains fully compatible with future extensions, including multilingual OCR and hybrid model ensembling.

The processing pipeline for real-time recognition of symbolic objects in video streams is structured as a staged data flow that closely aligns with the system architecture introduced in Figure 2.1 and the formal models defined in Section 2.2. It ensures the sequential transformation of input data—from raw video frames to final recognised symbolic strings—while supporting dynamic integration and decision-making. The sequential flow of data within the system, including preprocessing, text region detection, hypothesis generation, and result integration, is illustrated in Fig. 4.7.



igure 4.7 – Processing pipeline for symbolic object recognition in video streams.

The pipeline begins with the Video Acquisition Module, which receives input either from a live camera feed or a pre-recorded video file. Frames are extracted at a fixed or adaptive frame rate and timestamped. A lightweight scheduler, consistent with the discrete-time model described in Subsection 2.2, governs frame sampling and buffer management, ensuring that the system can drop

or skip frames when overloaded. Each extracted frame is passed to the Preprocessing Stage, where operations such as grayscale conversion, denoising, resizing, and document localisation are applied. These steps follow the procedures discussed in Subsections 1.3.1 to 1.3.3, including adaptive binarisation and projective correction for tilted or distorted documents. Segmentation techniques based on connected components and horizontal projection profiles isolate text fields of interest. The segmented region, typically a text line or symbolic string (e.g., MRZ or surname field), is forwarded to the Recognition Module, where OCR is performed using either a CNN-LSTM pipeline (for research evaluation) or Tesseract (for rapid baseline execution). Each recognised region produces a character sequence with associated confidence scores and alternative hypotheses, as described in Subsection 3.3. The output of each recognition pass is directed into the Integration Module, which maintains a growing set of recognition hypotheses per character position. This structure supports cumulative voting and probabilistic adjustment, realising the algorithm presented in Subsection 3.4. The integrated result is updated after each frame and stored as an evolving symbolic hypothesis. After each update, the Stopping Criterion Module evaluates whether further frame processing is likely to yield meaningful improvements. This is based on the expected gain computed as the predicted reduction in distance between current and potential future hypotheses (Subsection 4.4). If the gain falls below a dynamic or fixed threshold, the system halts processing. The Final Output Stage formats the integrated result into standardised output (e.g., UTF-8 string), logs metadata such as frame count, processing time, confidence levels, and stores the results in structured form (CSV, XML, or JSON). These outputs are essential for performance evaluation and statistical reporting as described in Appendices A and B. This pipeline is robust to input noise, supports asynchronous updates, and embodies the theoretical requirements for monotonicity and interruptibility stated in Section 4.1. Its modular nature allows substitution of OCR backends or segmentation techniques, enabling future adaptability for domain-specific deployments.

The performance of the implemented system was a critical factor in validating its applicability to real-time environments, particularly mobile and resource-constrained platforms. Consequently, several layers of optimisation were applied to ensure efficient processing without compromising recognition accuracy. At the algorithmic level, the integration method described in Subsection 3.4 was optimised by limiting the number of retained hypotheses per symbol, using a confidence threshold to prune low-probability alternatives. This reduced memory usage and computational overhead during result accumulation while maintaining robustness. Additionally, symbol-level updates were implemented with vectorised operations in NumPy, which provided significant acceleration compared to iterative list-based approaches. For the stopping criterion, the expected gain evaluation (Subsection 4.4) was reformulated to exploit incremental distance calculations, avoiding full recomputation of integrated results for each candidate frame. This reduced the per-frame overhead of the stopping module by approximately 35%, as observed in runtime profiling logs (Appendix A). To address the processing time bottleneck introduced by OCR, parallel execution was enabled using Python's concurrent.futures module, allowing concurrent recognition on multiple cropped regions within a frame. For experiments involving deep learning models (CNN-LSTM pipeline), GPU acceleration via CUDA was used selectively, leading to speed-ups of 2.5× to 4× in batch recognition compared to CPU-only runs. In the video pipeline, frame buffering was optimised through adaptive throttling: the system dynamically adjusts the frame sampling rate based on recognition latency. If the processing of previous frames is still ongoing, intermediate frames are skipped (as per the service model outlined in Subsection 2.2), ensuring system responsiveness under variable loads.

At the system level, the modular architecture enabled lightweight deployments. A full recognition cycle (preprocessing → OCR → integration → stopping decision) was completed in under 180 ms per frame on a mid-range desktop and in 320–350 ms on a Raspberry Pi 4B, with performance scaling depending on frame complexity and input resolution. This corresponds to real-time

throughput for short symbolic strings (5–15 characters), such as document identifiers or surnames. Furthermore, memory profiling showed that under typical conditions, peak RAM usage remained below 400 MB, even when retaining hypotheses over ten consecutive frames. This confirms the system's suitability for edge computing scenarios, as discussed in the experimental evaluation in Subsection 4.5. These optimisation strategies ensured that the developed recognition system not only met theoretical goals—such as monotonicity and anytime readiness—but also maintained practical feasibility for deployment in real-time applications across a range of hardware platforms.

The implemented system underwent comprehensive testing across multiple stages, aligning with the methodological framework established in Subsections 3.5 and 4.5. Testing aimed to assess the correctness, efficiency, and generalisability of the proposed models under varied conditions of input quality, frame dynamics, and computational capacity.

Functionality was first validated using controlled datasets, primarily the MIDV-500 benchmark, which offers annotated video recordings of identity documents with field-level ground truth. Test scenarios involved varying lighting conditions, occlusions, and motion blur, in order to reflect realistic deployment challenges described in Subsection 1.3.1. Experiments confirmed that the system correctly handled distorted inputs and low-resolution frames through cumulative integration, with improvements in symbolic recognition accuracy of up to 8.5% over single-frame baselines (see Table 5 and Figure 6).

To ensure reproducibility, all experiments were executed in a managed environment using Anaconda, with explicit recording of package versions, GPU usage, and system settings. For comparative evaluation, performance was benchmarked against the ROVER algorithm (baseline integration) and heuristic stopping rules, as detailed in Subsection 4.5. The proposed stopping method achieved an average reduction of 18% in the number of processed frames, while maintaining or exceeding baseline accuracy.

Deployment scenarios were considered in both desktop and embedded environments. A command-line interface (CLI) was provided for batch testing and integration into automated workflows, supporting video file input and per-frame logging. Additionally, a graphical interface was developed using Tkinter, enabling interactive exploration of recognition progress, hypothesis evolution, and stopping decisions. This was particularly useful for qualitative validation and demonstration purposes.

For lightweight deployment and portability, the system was containerised using Docker, with separate containers for the recognition core, preprocessing module, and result integration service. This allows flexible deployment on cloud platforms, serverless architectures, or mobile edge devices. Deployment tests were also conducted on Raspberry Pi 4B, confirming stable operation under limited CPU and RAM, albeit with increased latency (see Subsection 6).

In summary, the system has been validated through unit testing, benchmark-based evaluation, and real-time execution scenarios. It is deployment-ready for research, prototyping, and field testing, and the modular structure ensures extensibility toward multilingual support, document-type adaptation, or third-party integration.

4.7 Conclusions of the Chapter

This chapter addressed the task of stopping the process of object recognition in a real-time video stream, a novel and critical problem, especially related to developing optical recognition systems designed for mobile devices. A formal problem formulation was proposed, aligned with the classical stop problem framework, and a method was introduced that treats the object recognition process in a real-time video stream as a process, where the stopping point becomes monotonic after a certain step.

Based on the proposed method, an algorithm for stopping the string object recognition process in a real-time video stream was developed. The range among

the ongoing and the following integrated results was estimated by modeling the following integrated result using the accumulated observations.

The method was experimentally tested in the task of recognizing text arias in identity certificates using the open MIDV-500 dataset and the widely accessible Tesseract text recognition library. It was demonstrated that the precented stopping rule is more effective than thresholding based on the number of analized frames or thresholding based on the size of identical results largest cluster, despite the point that confidence recognition results scores were excluded from the model.

The final subsection of this chapter, Framework for Real-Time Recognition of Symbolic Objects in Video Streams, presents a generalised implementation framework that integrates the theoretical models, algorithms, and experimental results developed throughout the dissertation. It consolidates the architectural design, development environment, processing pipeline, optimisation strategies, and validation procedures into a coherent technological solution. The subsection illustrates how the proposed methods were operationalised in a modular software system capable of performing real-time symbolic object recognition under conditions of uncertainty and limited computational resources. The inclusion of original diagrams and a summarised technology stack reinforces the reproducibility and practical applicability of the developed system.

CONCLUSIONS

The primary findings of the dissertation study are as follows:

1. An analitical model of the object recognition system in a video stream has been built, featuring a block for combining frame-by-frame recognition results and a block for making decisions about stopping. The system's performance functionality was considered as a linear combination of the distance between the integrated recognition result and the true object value, and a penalty function based on the time from the start of the recording process to the stop. This model allows viewing the object recognition system in a video stream as an iterative computational process capable of providing the best solution at any given time and stopping the capture of new images according to a predefined stop rule.

2. An original study was conducted on the impact of input data characteristics on the selection of the optimal strategy for combining frame-by-frame classification results within the task of recognizing a single symbol in a video stream. It was shown that, if the sequence of processed images of a single image is free from preprocessing errors (such as errors in symbol localization and segmentation), the highest accuracy of the final result is achieved by the maximum evaluation rule. For video sequences with symbol localization and segmentation errors, higher accuracy of the final result is achieved by using the product of evaluations rule, voting rule, and sum of evaluations rule.

3. A new algorithm for combining the results of string object recognition has been developed, which takes into account alternative classification options for individual symbols (components of the string object). Experimental results show that the proposed algorithm can provide higher accuracy of the integrated result compared to the method of integrating recognition results as strings over a set of component value classes, applied to the text strings recognizing problem in a real-time video stream.

4. A task of stopping the process of object recognition in a real-time video stream was considered, which is Significant and innovative task, especially

relevant when developing optical recognition systems for mobile devices. A new method for stopping the process of object recognition in a real-time video stream has been developed based on thresholding an evaluation of the anticipated range among the ongoing and following integrated results. The method was developed under the assumption that the stopping task becomes monotonic from a certain step. Based on this method, a new algorithm for stopping the process of object recognition in a real-time video was proposed, where the evaluation is calculated by modeling the next integrated result using already accumulated observations. It was demonstrated that, in the task of text string recognition, the proposed stop rule is more effective than processed frames number thresholding or thresholding the size of the largest cluster of equivalent results.

5. The combined use of the developed algorithms (Algorithm 1 for combining recognition results of string objects, considering alternative symbol classification options, and Algorithm 2 for stopping the string recognition process) allows achieving higher recognition accuracy with an equal mean number of processed images. Table 8 shows the best achieved values of the mean range from the result to a true value under different constraints on the mean number of analized frames, with results integrated using Algorithm 1 and employing the considered stop algorithms, including Algorithm 2.

Table 8 – Best achieved values of the mean range among the integrated result and the ideal value at the stop moment; recognition results integrated using Algorithm 1

Stopping method	Best accuracy with a constraint on the mean number of frames					
	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
M_N	0.115	0.104	0.097	0.089	0.084	0.082
M_{KP}	$\varnothing$	0.083	0.080	0.078	0.073	0.072
M_{KY}	0.096	0.084	0.080	0.077	0.074	0.072
Alg. 1	**0.092**	**0.082**	**0.076**	**0.073**	**0.072**	**0.070**

6. This section 3.6 proposes a method for forming a set of symbol images by matching text field recognition results with operator-confirmed test strings during editing and verification, using dynamic programming and the Levenshtein distance. Based on the MCHSR algorithm, it adapts for optimal matching of a verified string with the recognition result. The process involves constructing a table to evaluate symbol matching quality, followed by finding the best matching path considering up, right, and diagonal transitions. The algorithm also handles cases with unmatched symbols due to recognition errors or extra symbols.

6. Key principles for creating open data packages for evaluating identity document recognition systems include using publicly available images, generating random document attribute data, and publishing annotated document templates with artificial data. Essential steps include simulating realistic shooting conditions (e.g., glare), classifying shooting parameters, and annotating document geometry. Video sequences should be annotated frame by frame with raw data, and attack simulations for document validation should be under the same conditions as original documents. These practices ensure reproducibility, consistency, and comparability in research while enabling future methodological improvements.

7. The practical implementation of the proposed methods was consolidated in the Framework for real-time recognition of symbolic objects in video streams (Section 4.6), which serves as a technological generalisation of the developed models and algorithms. This subsection outlines the system architecture, software environment, data processing pipeline, optimisation techniques, and testing procedures that ensure the feasibility of real-time deployment. The functional realisation confirms the adequacy and effectiveness of the proposed solutions under real-world constraints and demonstrates the potential for further application in mobile and embedded recognition systems.

The main results of the dissertation topic are presented in 3 articles in peer-reviewed journals of category B and conference proceedings.

REFERENCES

[1] N. Fir, “Does Debt Have Threshold Effects on Medium-Term Growth? Evidence from European Union Countries,” Naše gospodarstvo/Our economy, vol. 68, no. 2. Walter de Gruyter GmbH, pp. 1–18, Jun. 01, 2022. doi: 10.2478/ngoe-2022-0007.

[2] S. Schmid and H. Winkler, “Hybrid Production Management System in the Context of Industry 4.0,” 2022 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM). IEEE, pp. 1573–1577, Dec. 07, 2022. doi: 10.1109/ieem55944.2022.9990000.

[3] H. Modi and M. C., “A Review on Optical Character Recognition Techniques,” International Journal of Computer Applications, vol. 160, no. 6. Foundation of Computer Science, pp. 20–24, Feb. 15, 2017. doi: 10.5120/ijca2017913061.

[4] C. Wang, N. Guevara, and D. Caragea, “Using Deep Learning to Improve Detection and Decoding Of Barcodes,” 2022 IEEE International Conference on Image Processing (ICIP). IEEE, pp. 1576–1580, Oct. 16, 2022. doi: 10.1109/icip46576.2022.9897371.

[5] S. Rijhwani, D. Rosenblum, A. Anastasopoulos, and G. Neubig, “Lexically Aware Semi-Supervised Learning for OCR Post-Correction,” Transactions of the Association for Computational Linguistics, vol. 9. MIT Press - Journals, pp. 1285–1302, 2021. doi: 10.1162/tacl_a_00427.

[6] S. V. Mahadevkar, S. Patil, K. Kotecha, L. W. Soong, and T. Choudhury, “Exploring AI-driven approaches for unstructured document analysis and future horizons,” Journal of Big Data, vol. 11, no. 1. Springer Science and Business Media LLC, Jul. 05, 2024. doi: 10.1186/s40537-024-00948-z.

[7] A. Mittal and M. Dua, “Automatic speaker verification systems and spoof detection techniques: review and analysis,” International Journal of Speech Technology, vol. 25, no. 1. Springer Science and Business Media LLC, pp. 105–134, Aug. 16, 2021. doi: 10.1007/s10772-021-09876-2.

[8] E. Umuhoza and M. Brambilla, “Model Driven Development Approaches for Mobile Applications: A Survey,” Lecture Notes in Computer Science. Springer International Publishing, pp. 93–107, 2016. doi: 10.1007/978-3-319-44215-0_8.

[9] S. Jordan, S. S. Zabukovšek, and I. Š. Klančnik, “Document Management System – A Way to Digital Transformation,” Naše gospodarstvo/Our economy, vol. 68, no. 2. Walter de Gruyter GmbH, pp. 43–54, Jun. 01, 2022. doi: 10.2478/ngoe-2022-0010.

[10] K. Laitala, I. G. Klepp, V. Haugrønning, H. Throne-Holst, and P. Strandbakken, “Increasing repair of household appliances, mobile phones and clothing: Experiences from consumers and the repair industry,” Journal of Cleaner Production, vol. 282. Elsevier BV, p. 125349, Feb. 2021. doi: 10.1016/j.jclepro.2020.125349.

[11] P. Mayer et al., “Awareness, Intention, (In)Action: Individuals’ Reactions to Data Breaches,” ACM Transactions on Computer-Human Interaction, vol. 30, no. 5. Association for Computing Machinery (ACM), pp. 1–53, Sep. 23, 2023. doi: 10.1145/3589958.

[12] J. Rybacka, “Znaczenie procedury KYC z perspektywy instytucji bankowych na przykładzie Polski oraz postrzeganie tej polityki z perspektywy klientów – na podstawie badania empirycznego,” Journal of Finance and Financial Law, vol. 3, no. 43. Uniwersytet Lodzki (University of Lodz), pp. 117–137, Sep. 27, 2024. doi: 10.18778/2391-6478.3.43.07.

[13] T.-H. Chen, “Do you know your customer? Bank risk assessment based on machine learning,” Applied Soft Computing, vol. 86. Elsevier BV, p. 105779, Jan. 2020. doi: 10.1016/j.asoc.2019.105779.

[14] M. M. Taye, “Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions,” Computers, vol. 12, no. 5. MDPI AG, p. 91, Apr. 25, 2023. doi: 10.3390/computers12050091.

[15] J. Liang, D. Doermann, and H. Li, “Camera-based analysis of text and documents: a survey,” International Journal of Document Analysis and Recognition (IJDAR), vol. 7, no. 2–3. Springer Science and Business Media LLC, pp. 84–104, Jul. 2005. doi: 10.1007/s10032-004-0138-z.

[16] Z. Li et al., “Deep Learning-Based Object Detection Techniques for Remote Sensing Images: A Survey,” Remote Sensing, vol. 14, no. 10. MDPI AG, p. 2385, May 16, 2022. doi: 10.3390/rs14102385.

[17] X. Guanxiang, “Analysis of digital film and television special effects production technology,” Journal of Modern Information, no. 3, pp. 48–50, Jun. 16, 2017. doi: 10.3969/j.issn.1674-6708.2017.06.040.

[18] H. Michalak and K. Okarma, “Improvement of Image Binarization Methods Using Image Preprocessing with Local Entropy Filtering for Alphanumerical Character Recognition Purposes,” Entropy, vol. 21, no. 6. MDPI AG, p. 562, Jun. 04, 2019. doi: 10.3390/e21060562.

[19] F. Z. Ait Bella, M. El Rhabi, A. Hakim, and A. Laghrib, “An innovative document image binarization approach driven by the non-local p-Laplacian,” EURASIP Journal on Advances in Signal Processing, vol. 2022, no. 1. Springer Science and Business Media LLC, Jun. 18, 2022. doi: 10.1186/s13634-022-00883-2.

[20] K. Saeed and M. Albakoor, “Region growing based segmentation algorithm for typewritten and handwritten text recognition,” Applied Soft Computing, vol. 9, no. 2. Elsevier BV, pp. 608–617, Mar. 2009. doi: 10.1016/j.asoc.2008.08.006.

[21] D. Kumar and D. Singh, “Modified Approach of Hough Transform for Skew Detection and Correction in Documented Images,” International Journal of Research in Computer Science, vol. 2, no. 3. White Globe Publications, pp. 37–40, Apr. 30, 2012. doi: 10.7815/ijorcs.23.2012.027.

[22] X. Guanxiang, “Video and audio recording technology in Hunan Chinese dialect investigation of China language resource protection project,”

Journal of Media Technology, no. 8, pp. 39–45, Aug. 02, 2019. doi: 10.3969/j.issn.2096-0751.2019.02.009.

[23] S. Kundu, S. Malakar, Z. W. Geem, Y. Y. Moon, P. K. Singh, and R. Sarkar, “Hough Transform-Based Angular Features for Learning-Free Handwritten Keyword Spotting,” Sensors, vol. 21, no. 14. MDPI AG, p. 4648, Jul. 07, 2021. doi: 10.3390/s21144648.

[24] A. Amin and S. Fischer, “A Document Skew Detection Method Using the Hough Transform,” Pattern Analysis & Applications, vol. 3, no. 3. Springer Science and Business Media LLC, pp. 243–253, Sep. 28, 2000. doi: 10.1007/s100440070009.

[25] R. Safabakhsh and S. Khadivi, “Document skew detection using minimum-area bounding rectangle,” Proceedings International Conference on Information Technology: Coding and Computing (Cat. No.PR00540). IEEE Comput. Soc, pp. 253–258. doi: 10.1109/itcc.2000.844226.

[26] S. Imahori, M. Yagiura, and T. Ibaraki, “Improved local search algorithms for the rectangle packing problem with general spatial costs,” European Journal of Operational Research, vol. 167, no. 1. Elsevier BV, pp. 48–67, Nov. 2005. doi: 10.1016/j.ejor.2004.02.020.

[27] C. F. Chong, Y. Wang, B. Ng, W. Luo, and X. Yang, “Image projective transformation rectification with synthetic data for smartphone-captured chest X-ray photos classification,” Computers in Biology and Medicine, vol. 164. Elsevier BV, p. 107277, Sep. 2023. doi: 10.1016/j.compbiomed.2023.107277.

[28] X. Guanxiang, “Preliminary quality control in audio-visual recording of Hunan Chinese dialect investigation of China language resources protection project,” Journal of Language Protection, no. 3, pp. 241–243, Mar. 15, 2022. [Online]. Available: https://d.wanfangdata.com.cn/periodical/Ch9QZXJpb2RpY2FsQ0hJTmV3UzIwMjUwMTA0MTcwMjI2EhpRS0JKQkQyMDIyMjAyMjAzMTgwMDAwNjk0NBoIZHBpNGZuOTQ%3D.

[29] C. F. Chong, Y. Wang, B. Ng, W. Luo, and X. Yang, “Image Projective Transformation Rectification with Synthetic Data for Smartphone-captured Chest X-ray Photos Classification,” arXiv, 2022, doi: 10.48550/ARXIV.2210.05954.

[30] A. Soycan and M. Soycan, “Perspective correction of building facade images for architectural applications,” Engineering Science and Technology, an International Journal, vol. 22, no. 3. Elsevier BV, pp. 697–705, Jun. 2019. doi: 10.1016/j.jestch.2018.12.012.

[31] V. Blahnik and O. Schindelbeck, “Smartphone imaging technology and its applications,” Advanced Optical Technologies, vol. 10, no. 3. Frontiers Media SA, pp. 145–232, Jun. 01, 2021. doi: 10.1515/aot-2021-0023.

[32] B. G. Gatos, “Imaging Techniques in Document Analysis Processes,” Handbook of Document Image Processing and Recognition. Springer London, pp. 73–131, 2014. doi: 10.1007/978-0-85729-859-1_4.

[33] M. Alemán-Flores, L. Alvarez, L. Gomez, and D. Santana-Cedrés, “Line detection in images showing significant lens distortion and application to distortion correction,” Pattern Recognition Letters, vol. 36. Elsevier BV, pp. 261–271, Jan. 2014. doi: 10.1016/j.patrec.2013.06.020.

[34] W. Błaszczak-Bąk, J. Janicka, and A. Sobieraj-Żłobińska, “Applying RANSAC Algorithm for Fitting Scanning Strips from Airborne Laser Scanning,” Civil and Environmental Engineering Reports, vol. 23, no. 4. University of Zielona Góra, Poland, pp. 29–42, Dec. 01, 2016. doi: 10.1515/ceer-2016-0048.

[35] M. A. Javeed et al., “Lane Line Detection and Object Scene Segmentation Using Otsu Thresholding and the Fast Hough Transform for Intelligent Vehicles in Complex Road Conditions,” Electronics, vol. 12, no. 5. MDPI AG, p. 1079, Feb. 21, 2023. doi: 10.3390/electronics12051079.

[36] M. Ibrahim, M. Wagdy, F. S. AlHarithi, A. M. Qahtani, W. S. Elkilani, and S. Zarif, “An Efficient Method for Document Correction Based on Checkerboard Calibration Pattern,” Applied Sciences, vol. 12, no. 18. MDPI AG, p. 9014, Sep. 08, 2022. doi: 10.3390/app12189014.

[37] Y.-Q. Wang, “An Analysis of the Viola-Jones Face Detection Algorithm,” Image Processing On Line, vol. 4. Image Processing On Line, pp. 128–148, Jun. 26, 2014. doi: 10.5201/ipol.2014.104.

[38] G. Mexi, S. Shamsi, M. Besançon, and P. L. Bodic, “Probabilistic Lookahead Strong Branching via a Stochastic Abstract Branching Model,” 2023, arXiv. doi: 10.48550/ARXIV.2312.07041.

[39] X. Guanxiang, “Post-Processing Software Application Techniques in the Audio-visual Recording of Hunan Chinese Dialect Survey for the Language Protection Project - Taking Adobe Premiere Pro as an Example,” Journal of Audio-visual Studies, no. 5, pp. 162–166, Apr. 15, 2022. [Online]. Available: https://d.wanfangdata.com.cn/periodical/Ch9QZXJpb2RpY2FsQ0hJTmV3UzIwMjUwMTA0MTcwMjI2EhNzbXNqLXNoYW5nMjAyMjA3MDQ3GghkcGk0Zm45NA%3D%3D.

[40] R. S. Thakur, S. Chatterjee, R. N. Yadav, and L. Gupta, “Image De-Noising With Machine Learning: A Review,” IEEE Access, vol. 9. Institute of Electrical and Electronics Engineers (IEEE), pp. 93338–93363, 2021. doi: 10.1109/access.2021.3092425.

[41] D. Chudasama, T. Patel, S. Joshi, and G. I. Prajapati, “Image Segmentation using Morphological Operations,” International Journal of Computer Applications, vol. 117, no. 18. Foundation of Computer Science, pp. 16–19, May 20, 2015. doi: 10.5120/20654-3197.

[42] L. Xu et al., “Gaussian process image classification based on multi-layer convolution kernel function,” Neurocomputing, vol. 480. Elsevier BV, pp. 99–109, Apr. 2022. doi: 10.1016/j.neucom.2022.01.048.

[43] T. Fu, L. Ma, M. Li, and B. A. Johnson, “Using convolutional neural network to identify irregular segmentation objects from very high-resolution remote sensing imagery,” Journal of Applied Remote Sensing, vol. 12, no. 02. SPIE-Intl Soc Optical Eng, p. 1, May 17, 2018. doi: 10.1117/1.jrs.12.025010.

[44] Yi Li, Yefeng Zheng, and D. Doermann, “Detecting Text Lines in Handwritten Documents,” 18th International Conference on Pattern Recognition (ICPR’06). IEEE, pp. 1030–1033, 2006. doi: 10.1109/icpr.2006.435.

[45] A. Droby, B. Kurar Barakat, R. Saabni, R. Alaasam, B. Madi, and J. El-Sana, “Understanding Unsupervised Deep Learning for Text Line Segmentation,” Applied Sciences, vol. 12, no. 19. MDPI AG, p. 9528, Sep. 22, 2022. doi: 10.3390/app12199528.

[46] Y. Yu et al., “Techniques and Challenges of Image Segmentation: A Review,” Electronics, vol. 12, no. 5. MDPI AG, p. 1199, Mar. 02, 2023. doi: 10.3390/electronics12051199.

[47] J. Tanevski, L. Todorovski, and S. Džeroski, “Combinatorial search for selecting the structure of models of dynamical systems with equation discovery,” Engineering Applications of Artificial Intelligence, vol. 89. Elsevier BV, p. 103423, Mar. 2020. doi: 10.1016/j.engappai.2019.103423.

[48] L. LALAOUI and T. MOHAMADI, “A comparative study of Image Region-Based Segmentation Algorithms,” International Journal of Advanced Computer Science and Applications, vol. 4, no. 6. The Science and Information Organization, 2013. doi: 10.14569/ijacsa.2013.040627.

[49] H. Liang, X. Sun, Y. Sun, and Y. Gao, “Text feature extraction based on deep learning: a review,” EURASIP Journal on Wireless Communications and Networking, vol. 2017, no. 1. Springer Science and Business Media LLC, Dec. 2017. doi: 10.1186/s13638-017-0993-1.

[50] C. Yang, Y. Guo, X. Li, and B. Chen, “A Novel Method Using Local Feature to Enhance GCN for Text Classification,” 2021 11th International Conference on Intelligent Control and Information Processing (ICICIP). IEEE, pp. 59–65, Dec. 03, 2021. doi: 10.1109/icicip53388.2021.9642171.

[51] M. Wolfgang, M. Weißensteiner, P. Clarke, W.-K. Hsiao, and J. G. Khinast, “Deep convolutional neural networks: Outperforming established algorithms in the evaluation of industrial optical coherence tomography (OCT)

images of pharmaceutical coatings," International Journal of Pharmaceutics: X, vol. 2. Elsevier BV, p. 100058, Dec. 2020. doi: 10.1016/j.ijpx.2020.100058.

[52] W. Rawat and Z. Wang, "Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review," Neural Computation, vol. 29, no. 9. MIT Press, pp. 2352–2449, Sep. 2017. doi: 10.1162/neco_a_00990.

[53] X. Guanxiang, "A Study of the Implementation Techniques and Processes of Live Video Courses in Online Teaching and Learning," Journal of Online Learning, no. 2, pp. 3224–3225, Jun. 29, 2022. doi: 10.12255/j.issn.1672-6677.2021.11.1603.

[54] W. AlKendi, F. Gechter, L. Heyberger, and C. Guyeux, "Advancements and Challenges in Handwritten Text Recognition: A Comprehensive Survey," Journal of Imaging, vol. 10, no. 1. MDPI AG, p. 18, Jan. 08, 2024. doi: 10.3390/jimaging10010018.

[55] A. Souhar, Y. Boulid, E. Ameur, and M. Ouagague, "Segmentation of Arabic Handwritten Documents into Text Lines using Watershed Transform," International Journal of Interactive Multimedia and Artificial Intelligence, vol. 4, no. 6. Universidad Internacional de La Rioja, p. 96, 2017. doi: 10.9781/ijimai.2017.08.002.

[56] H. Michalak and K. Okarma, "Improvement of Image Binarization Methods Using Image Preprocessing with Local Entropy Filtering for Alphanumerical Character Recognition Purposes," Entropy, vol. 21, no. 6. MDPI AG, p. 562, Jun. 04, 2019. doi: 10.3390/e21060562.

[57] H. Michalak and K. Okarma, "Robust Combined Binarization Method of Non-Uniformly Illuminated Document Images for Alphanumerical Character Recognition," Sensors, vol. 20, no. 10. MDPI AG, p. 2914, May 21, 2020. doi: 10.3390/s20102914.

[58] S. F. Ahmed et al., "Deep learning modelling techniques: current progress, applications, advantages, and challenges," Artificial Intelligence Review, vol. 56, no. 11. Springer Science and Business Media LLC, pp. 13521–13617, Apr. 17, 2023. doi: 10.1007/s10462-023-10466-8.

[59] M. M. Taye, “Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions,” Computers, vol. 12, no. 5. MDPI AG, p. 91, Apr. 25, 2023. doi: 10.3390/computers12050091.

[60] N. Manakitsa, G. S. Maraslidis, L. Moysis, and G. F. Fragulis, “A Review of Machine Learning and Deep Learning for Object Detection, Semantic Segmentation, and Human Action Recognition in Machine and Robotic Vision,” Technologies, vol. 12, no. 2. MDPI AG, p. 15, Jan. 23, 2024. doi: 10.3390/technologies12020015.

[61] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, “State-of-the-art in artificial neural network applications: A survey,” Heliyon, vol. 4, no. 11. Elsevier BV, p. e00938, Nov. 2018. doi: 10.1016/j.heliyon.2018.e00938.

[62] G. R. Yang and X.-J. Wang, “Artificial Neural Networks for Neuroscientists: A Primer,” Neuron, vol. 107, no. 6. Elsevier BV, pp. 1048–1070, Sep. 2020. doi: 10.1016/j.neuron.2020.09.005.

[63] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, “1D convolutional neural networks and applications: A survey,” Mechanical Systems and Signal Processing, vol. 151. Elsevier BV, p. 107398, Apr. 2021. doi: 10.1016/j.ymssp.2020.107398.

[64] J.-S. Lerat, S. A. Mahmoudi, and S. Mahmoudi, “Distributed Deep Learning: From Single-Node to Multi-Node Architecture,” Electronics, vol. 11, no. 10. MDPI AG, p. 1525, May 10, 2022. doi: 10.3390/electronics11101525.

[65] J. Guo, Y. Yang, H. Li, L. Dai, and B. Huang, “A parallel deep neural network for intelligent fault diagnosis of drilling pumps,” Engineering Applications of Artificial Intelligence, vol. 133. Elsevier BV, p. 108071, Jul. 2024. doi: 10.1016/j.engappai.2024.108071.

[66] E. Mohamed, K. Sirlantzis, and G. Howells, “A review of visualisation-as-explanation techniques for convolutional neural networks and their

evaluation," Displays, vol. 73. Elsevier BV, p. 102239, Jul. 2022. doi: 10.1016/j.displa.2022.102239.

[67] L. Alzubaidi et al., "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," Journal of Big Data, vol. 8, no. 1. Springer Science and Business Media LLC, Mar. 31, 2021. doi: 10.1186/s40537-021-00444-8.

[68] R. Suzuki, "Optimization of Neural Network Architectures for Image Recognition," Asian Journal of Computing and Engineering Technology, vol. 5, no. 1. IPR Journals and Books (International Peer Reviewed Journals and Books), pp. 1–10, Jul. 29, 2024. doi: 10.47604/ajcet.2806.

[69] L. Chen, S. Li, Q. Bai, J. Yang, S. Jiang, and Y. Miao, "Review of Image Classification Algorithms Based on Convolutional Neural Networks," Remote Sensing, vol. 13, no. 22. MDPI AG, p. 4712, Nov. 21, 2021. doi: 10.3390/rs13224712.

[70] L. Alzubaidi et al., "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," Journal of Big Data, vol. 8, no. 1. Springer Science and Business Media LLC, Mar. 31, 2021. doi: 10.1186/s40537-021-00444-8.

[71] J. Song, S. Gao, Y. Zhu, and C. Ma, "A survey of remote sensing image classification based on CNNs," Big Earth Data, vol. 3, no. 3. Informa UK Limited, pp. 232–254, Jul. 03, 2019. doi: 10.1080/20964471.2019.1657720.

[72] X. Zhao, L. Wang, Y. Zhang, X. Han, M. Deveci, and M. Parmar, "A review of convolutional neural networks in computer vision," Artificial Intelligence Review, vol. 57, no. 4. Springer Science and Business Media LLC, Mar. 23, 2024. doi: 10.1007/s10462-024-10721-6.

[73] M. Mundt, Y. Hong, I. Pliushch, and V. Ramesh, "A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning," Neural Networks, vol. 160. Elsevier BV, pp. 306–336, Mar. 2023. doi: 10.1016/j.neunet.2023.01.014.

[74] M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A. M. Karimi-Mamaghan, and E.-G. Talbi, “Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art,” European Journal of Operational Research, vol. 296, no. 2. Elsevier BV, pp. 393–422, Jan. 2022. doi: 10.1016/j.ejor.2021.04.032.

[75] A. Tsamados et al., “The ethics of algorithms: key problems and solutions,” AI & SOCIETY, vol. 37, no. 1. Springer Science and Business Media LLC, pp. 215–230, Feb. 20, 2021. doi: 10.1007/s00146-021-01154-8.

A. Fort, M. Mugnaini, and V. Vignoli, “Hidden Markov Models approach used for life parameters estimations,” Reliability Engineering & System Safety, vol. 136. Elsevier BV, pp. 85–91, Apr. 2015. doi: 10.1016/j.ress.2014.11.017.

[76] M. M. Alwateer, M. Elmezain, M. Farsi, and E. Atlam, “Hidden Markov Models for Pattern Recognition,” Markov Model - Theory and Applications. IntechOpen, Apr. 13, 2023. doi: 10.5772/intechopen.1001364.

[77] J. Daciuk, J. Piskorski, and S. Ristov, “Natural Language Dictionaries Implemented as Finite Automata,” Mathematics, Computing, Language, and Life: Frontiers in Mathematical Linguistics and Language Theory. IMPERIAL COLLEGE PRESS, pp. 133–204, Oct. 2010. doi: 10.1142/9781848165458_0004.

[78] M. Mohri, F. Pereira, and M. Riley, “Weighted finite-state transducers in speech recognition,” Computer Speech & Language, vol. 16, no. 1. Elsevier BV, pp. 69–88, Jan. 2002. doi: 10.1006/csla.2001.0184.

[79] M. Seo, H. Iida, and J. W. H. M. Uiterwijk, “The PN*-search algorithm: Application to tsume-shogi,” Artificial Intelligence, vol. 129, no. 1–2. Elsevier BV, pp. 253–277, Jun. 2001. doi: 10.1016/s0004-3702(01)00084-4.

[80] X.-C. Yin, H.-W. Hao, Y.-F. Tang, J. Sun, and S. Naoi, “Rejection Strategies with Multiple Classifiers for Handwritten Character Recognition,” 2009 10th International Conference on Document Analysis and Recognition. IEEE, pp. 1126–1130, 2009. doi: 10.1109/icdar.2009.45.

[81] A. Gupta, A. Anpalagan, L. Guan, and A. S. Khwaja, “Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues,” Array, vol. 10. Elsevier BV, p. 100057, Jul. 2021. doi: 10.1016/j.array.2021.100057.

[82] P. Delgado-Santos, G. Stragapede, R. Tolosana, R. Guest, F. Deravi, and R. Vera-Rodriguez, “A Survey of Privacy Vulnerabilities of Mobile Device Sensors,” ACM Computing Surveys, vol. 54, no. 11s. Association for Computing Machinery (ACM), pp. 1–30, Jan. 31, 2022. doi: 10.1145/3510579.

[83] M. Cinelli, M. Kadziński, M. Gonzalez, and R. Słowiński, “How to support the application of multiple criteria decision analysis? Let us start with a comprehensive taxonomy,” Omega, vol. 96. Elsevier BV, p. 102261, Oct. 2020. doi: 10.1016/j.omega.2020.102261.

[84] S. F. Ahmed et al., “Deep learning modelling techniques: current progress, applications, advantages, and challenges,” Artificial Intelligence Review, vol. 56, no. 11. Springer Science and Business Media LLC, pp. 13521–13617, Apr. 17, 2023. doi: 10.1007/s10462-023-10466-8.

[85] L. He, G. Jin, and S.-B. Tsai, “Design and Implementation of Embedded Real-Time English Speech Recognition System Based on Big Data Analysis,” Mathematical Problems in Engineering, vol. 2021. Hindawi Limited, pp. 1–12, Sep. 01, 2021. doi: 10.1155/2021/6561730.

[86] X. Haihua, Z. Jie, and G. Wu, “An efficient multistage Rover method for Automatic Speech recognition,” 2009 IEEE International Conference on Multimedia and Expo. IEEE, Jun. 2009. doi: 10.1109/icme.2009.5202639.

[87] X. Guanxiang and V. Kovtun, “Methods for implementation of string object recognition results in real-time video streams,” Measuring And Computing Devices In Technological Processes, no. 4. Khmelnytskyi National University, pp. 338–347, Nov. 28, 2024. doi: 10.31891/2219-9365-2024-80-41.

[88] D. Mindrila, “Bayesian Latent Class Analysis: Sample Size, Model Size, and Classification Precision,” Mathematics, vol. 11, no. 12. MDPI AG, p. 2753, Jun. 17, 2023. doi: 10.3390/math11122753.

[89] S. H. Cheung and J. L. Beck, “Calculation of Posterior Probabilities for Bayesian Model Class Assessment and Averaging from Posterior Samples Based on Dynamic System Data,” Computer-Aided Civil and Infrastructure Engineering, vol. 25, no. 5. Wiley, pp. 304–321, Feb. 10, 2010. doi: 10.1111/j.1467-8667.2009.00642.x.

[90] Y. Tang, Y. Zhou, X. Ren, Y. Sun, Y. Huang, and D. Zhou, “A new basic probability assignment generation and combination method for conflict data fusion in the evidence theory,” Scientific Reports, vol. 13, no. 1. Springer Science and Business Media LLC, May 25, 2023. doi: 10.1038/s41598-023-35195-4.

[91] Y.-W. Du and J.-J. Zhong, “Group inference method of attribution theory based on Dempster–Shafer theory of evidence,” Knowledge-Based Systems, vol. 188. Elsevier BV, p. 104985, Jan. 2020. doi: 10.1016/j.knosys.2019.104985.

[92] S. Tulyakov, S. Jaeger, V. Govindaraju, and D. Doermann, “Review of Classifier Combination Methods,” Studies in Computational Intelligence. Springer Berlin Heidelberg, pp. 361–386, 2008. doi: 10.1007/978-3-540-76280-5_14.

[93] M. F. Hassan, I. Abdel-Qader, and B. Bazuin, “A new method for ensemble combination based on adaptive decision making,” Knowledge-Based Systems, vol. 233. Elsevier BV, p. 107544, Dec. 2021. doi: 10.1016/j.knosys.2021.107544.

[94] M. Mohandes, M. Deriche, and S. O. Aliyu, “Classifiers Combination Techniques: A Comprehensive Review,” IEEE Access, vol. 6. Institute of Electrical and Electronics Engineers (IEEE), pp. 19626–19639, 2018. doi: 10.1109/access.2018.2813079.

[95] J. M. Tomczak and M. Zięba, “Probabilistic combination of classification rules and its application to medical diagnosis,” Machine Learning, vol. 101, no. 1–3. Springer Science and Business Media LLC, pp. 105–135, Jun. 20, 2015. doi: 10.1007/s10994-015-5508-x.

[96] A. A. Khan, O. Chaudhari, and R. Chandra, “A review of ensemble learning and data augmentation models for class imbalanced problems:

Combination, implementation and evaluation," Expert Systems with Applications, vol. 244. Elsevier BV, p. 122778, Jun. 2024. doi: 10.1016/j.eswa.2023.122778.

[97] M. Albardan, J. Klein, and O. Colot, "SPOCC: Scalable POssibilistic Classifier Combination - toward robust aggregation of classifiers," Expert Systems with Applications, vol. 150. Elsevier BV, p. 113332, Jul. 2020. doi: 10.1016/j.eswa.2020.113332.

[98] E. Dilek and M. Dener, "Computer Vision Applications in Intelligent Transportation Systems: A Survey," Sensors, vol. 23, no. 6. MDPI AG, p. 2938, Mar. 08, 2023. doi: 10.3390/s23062938.

[99] G. Lavee, E. Rivlin, and M. Rudzsky, "Understanding Video Events: A Survey of Methods for Automatic Interpretation of Semantic Occurrences in Video," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 39, no. 5. Institute of Electrical and Electronics Engineers (IEEE), pp. 489–504, Sep. 2009. doi: 10.1109/tsmcc.2009.2023380.

[100] M. Á. Domínguez-Ríos, F. Chicano, and E. Alba, "Effective anytime algorithm for multiobjective combinatorial optimization problems," Information Sciences, vol. 565. Elsevier BV, pp. 210–228, Jul. 2021. doi: 10.1016/j.ins.2021.02.074.

I. Bogun, A. Angelova, and N. Jaitly, "Object Recognition from Short Videos for Robotic Perception," 2015, arXiv. doi: 10.48550/ARXIV.1509.01602.

[101] M. Leszczuk, L. Janowski, J. Nawała, and A. Boev, "Objective Video Quality Assessment Method for Object Recognition Tasks," Electronics, vol. 13, no. 9. MDPI AG, p. 1750, May 01, 2024. doi: 10.3390/electronics13091750.

[102] R. de Heide and P. D. Grünwald, "Why optional stopping can be a problem for Bayesians," Psychonomic Bulletin & Review, vol. 28, no. 3. Springer Science and Business Media LLC, pp. 795–812, Nov. 18, 2020. doi: 10.3758/s13423-020-01803-x.

[103] S. Razavi et al., "The Future of Sensitivity Analysis: An essential discipline for systems modeling and policy support," Environmental Modelling &

Software, vol. 137. Elsevier BV, p. 104954, Mar. 2021. doi: 10.1016/j.envsoft.2020.104954.

[104] N. Elgendy, A. Elragal, and T. Päivärinta, “DECAS: a modern data-driven decision theory for big data and analytics,” Journal of Decision Systems, vol. 31, no. 4. Informa UK Limited, pp. 337–373, Mar. 04, 2021. doi: 10.1080/12460125.2021.1894674.

[105] H. Taherdoost, “Deep Learning and Neural Networks: Decision-Making Implications,” Symmetry, vol. 15, no. 9. MDPI AG, p. 1723, Sep. 08, 2023. doi: 10.3390/sym15091723.

[106] L. Meng and M. Q. Zhao, “Bride Drain: An unintended consequence of China’s urban-rural divide,” Labour Economics, vol. 58. Elsevier BV, pp. 69–80, Jun. 2019. doi: 10.1016/j.labeco.2019.04.003.

[107] S. C. Albright, “A Bayesian Approach to a Generalized House Selling Problem,” Management Science, vol. 24, no. 4. Institute for Operations Research and the Management Sciences (INFORMS), pp. 432–440, Dec. 1977. doi: 10.1287/mnsc.24.4.432.

[108] G. Sofronov, “An Optimal Decision Rule for a Multiple Selling Problem with a Variable Rate of Offers,” Mathematics, vol. 8, no. 5. MDPI AG, p. 690, May 02, 2020. doi: 10.3390/math8050690.

[109] Z. Wu, D. Ji, K. Yu, X. Zeng, D. Wu, and M. Shidujaman, “AI Creativity and the Human-AI Co-creation Model,” Lecture Notes in Computer Science. Springer International Publishing, pp. 171–190, 2021. doi: 10.1007/978-3-030-78462-1_13.

[110] M. S. Reed et al., “Evaluating impact from research: A methodological framework,” Research Policy, vol. 50, no. 4. Elsevier BV, p. 104147, May 2021. doi: 10.1016/j.respol.2020.104147.

[111] G. Murazvu, S. Parkinson, S. Khan, N. Liu, and G. Allen, “A Survey on Factors Preventing the Adoption of Automated Software Testing: A Principal Component Analysis Approach,” Software, vol. 3, no. 1. MDPI AG, pp. 1–27, Jan. 02, 2024. doi: 10.3390/software3010001.

[112] P. Wakker and M. P. Klaassen, “Confidence intervals for cost/effectiveness ratios,” Health Economics, vol. 4, no. 5. Wiley, pp. 373–381, Sep. 1995. doi: 10.1002/hec.4730040503.

[113] M. Tambour, N. Zethraeus, and M. Johannesson, “A Note on Confidence Intervals in Cost-Effectiveness Analysis,” International Journal of Technology Assessment in Health Care, vol. 14, no. 3. Cambridge University Press (CUP), pp. 467–471, 1998. doi: 10.1017/s0266462300011442.

[114] M. Rigotti, “Internal representation of task rules by recurrent dynamics: the importance of the diversity of neural responses,” Frontiers in Computational Neuroscience, vol. 4. Frontiers Media SA, 2010. doi: 10.3389/fncom.2010.00024.

[115] H. Al-Mekhlafi and S. Liu, “Single image super-resolution: a comprehensive review and recent insight,” Frontiers of Computer Science, vol. 18, no. 1. Springer Science and Business Media LLC, Sep. 04, 2023. doi: 10.1007/s11704-023-2588-9.

[116] H. Zhang, C. Ye, Y. Zhou, R. Tang, and R. Wei, “A Super-Resolution Network for High-Resolution Reconstruction of Landslide Main Bodies in Remote Sensing Imagery Using Coordinated Attention Mechanisms and Deep Residual Blocks,” Remote Sensing, vol. 15, no. 18. MDPI AG, p. 4498, Sep. 13, 2023. doi: 10.3390/rs15184498.

B. Alaya and L. Sellami, “Multilayer Video Encoding for QoS Managing of Video Streaming in VANET Environment,” ACM Transactions on Multimedia Computing, Communications, and Applications, vol. 18, no. 3. Association for Computing Machinery (ACM), pp. 1–19, Mar. 04, 2022. doi: 10.1145/3491433.

[117] M. Gomathy Nayagam and K. Ramar, “Reliable object recognition system for cloud video data based on LDP features,” Computer Communications, vol. 149. Elsevier BV, pp. 343–349, Jan. 2020. doi: 10.1016/j.comcom.2019.10.027.

[118] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, “Machine learning: a review of classification and combining techniques,” Artificial Intelligence

Review, vol. 26, no. 3. Springer Science and Business Media LLC, pp. 159–190, Nov. 2006. doi: 10.1007/s10462-007-9052-3.

[119] X. Guanxiang and V. Kovtun, “The model of the system for objects recognition in the real-time video stream”, Computer systems and information technologies, no. 4, pp. 157–162, Dec. 26, 2024. dooi: 10.31891/csit-2024-4-19.

[120] A. Guerriero, M. R. Lyu, R. Pietrantuono, and S. Russo, “Assessing operational accuracy of CNN-based image classifiers using an oracle surrogate,” Intelligent Systems with Applications, vol. 17. Elsevier BV, p. 200172, Feb. 2023. doi: 10.1016/j.iswa.2022.200172.

[121] F. Li, F. Fang, Z. Li, and T. Zeng, “Single image noise level estimation by artificial noise,” Signal Processing, vol. 213. Elsevier BV, p. 109215, Dec. 2023. doi: 10.1016/j.sigpro.2023.109215.

[122] A. Luque, A. Carrasco, A. Martín, and A. de las Heras, “The impact of class imbalance in classification performance metrics based on the binary confusion matrix,” Pattern Recognition, vol. 91. Elsevier BV, pp. 216–231, Jul. 2019. doi: 10.1016/j.patcog.2019.02.023.

[123] S. ANSOLABEHERE, J. RODDEN, and J. M. SNYDER JR., “The Strength of Issues: Using Multiple Measures to Gauge Preference Stability, Ideological Constraint, and Issue Voting,” American Political Science Review, vol. 102, no. 2. Cambridge University Press (CUP), pp. 215–232, May 2008. doi: 10.1017/s0003055408080210.

[124] S. Elo, M. Kääriäinen, O. Kanste, T. Pölkki, K. Utriainen, and H. Kyngäs, “Qualitative Content Analysis,” Sage Open, vol. 4, no. 1. SAGE Publications, Jan. 01, 2014. doi: 10.1177/2158244014522633.

[125] N. Bayar, K. Guzel, and D. Kumlu, “MobileMRZNet: Efficient and Lightweight MRZ Detection for Mobile Devices.” Research Square Platform LLC, Jun. 05, 2023. doi: 10.21203/rs.3.rs-3008687/v1.

[126] S. Seoni, V. Jahmunah, M. Salvi, P. D. Barua, F. Molinari, and U. R. Acharya, “Application of uncertainty quantification to artificial intelligence in healthcare: A review of last decade (2013–2023),” Computers in Biology and

Medicine, vol. 165. Elsevier BV, p. 107441, Oct. 2023. doi: 10.1016/j.compbiomed.2023.107441.

[127] M. Kopytek, P. Lech, and K. Okarma, “Application of Binary Image Quality Assessment Methods to Predict the Quality of Optical Character Recognition Results,” Applied Sciences, vol. 14, no. 22. MDPI AG, p. 10275, Nov. 08, 2024. doi: 10.3390/app142210275.

[128] P. Sharma, “Advancements in OCR: A Deep Learning Algorithm for Enhanced Text Recognition,” International Journal of Inventive Engineering and Sciences, vol. 10, no. 8. Blue Eyes Intelligence Engineering and Sciences Engineering and Sciences Publication - BEIESP, pp. 1–7, Aug. 30, 2023. doi: 10.35940/ijies.f4263.0810823.

[129] G. S. Morrison and E. Enzinger, “Score based procedures for the calculation of forensic likelihood ratios – Scores should take account of both similarity and typicality,” Science & Justice, vol. 58, no. 1. Elsevier BV, pp. 47–58, Jan. 2018. doi: 10.1016/j.scijus.2017.06.005.

[130] L. Yujian and L. Bo, “A Normalized Levenshtein Distance Metric,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 6. Institute of Electrical and Electronics Engineers (IEEE), pp. 1091–1095, Jun. 2007. doi: 10.1109/tpami.2007.1078.

[131] L. Yujian and L. Bo, “A Normalized Levenshtein Distance Metric,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 6. Institute of Electrical and Electronics Engineers (IEEE), pp. 1091–1095, Jun. 2007. doi: 10.1109/tpami.2007.1078.

[132] “Dynamic Time Warping,” Information Retrieval for Music and Motion. Springer Berlin Heidelberg, pp. 69–84, 2007. doi: 10.1007/978-3-540-74048-3_4.

[133] X. Guanxiang and V. Kovtun, “The method for stopping the string object recognition process in a real-time video stream,” Measuring And Computing Devices In Technological Processes, no. 3. Khmelnytskyi National University, pp. 243–250, Aug. 29, 2024. doi: 10.31891/2219-9365-2024-79-32.

[134] Pohl, “Heuristic search viewed as path finding in a graph,” Artificial Intelligence, vol. 1, no. 3–4. Elsevier BV, pp. 193–204, Jan. 1970. doi: 10.1016/0004-3702(70)90007-x.

[135] G. A. Robby, A. Tandra, I. Susanto, J. Harefa, and A. Chowanda, “Implementation of Optical Character Recognition using Tesseract with the Javanese Script Target in Android Application,” Procedia Computer Science, vol. 157. Elsevier BV, pp. 499–505, 2019. doi: 10.1016/j.procs.2019.09.006.

[136] A. Deaton and N. Cartwright, “Understanding and misunderstanding randomized controlled trials,” Social Science & Medicine, vol. 210. Elsevier BV, pp. 2–21, Aug. 2018. doi: 10.1016/j.socscimed.2017.12.005.

Appendix A. Listing of Baseline Algorithms Realization

Algorithm 1: Integration of recognition results of a string object

```
def integrate_recognition_results(M, P, omega):
    Y = P[0]  # Initialize Y with the first sequence
    Z = omega[0]  # Initialize Z with the first weight

    for i in range(1, M):
        # Initialize gamma matrix and path label matrix r
        gamma = [[0] * (len(Y) + 1) for _ in range(len(P[i]) + 1)]
        r = [[0] * (len(Y) + 1) for _ in range(len(P[i]) + 1)]

        # Fill for alignment of P[i] with the empty symbol
        for c in range(1, len(P[i]) + 1):
            gamma[c][0] = gamma[c - 1][0] + P[i][c - 1]
            r[c][0] = 1  # Path 1

        # Fill for alignment of Y with the empty symbol
        for c in range(1, len(Y) + 1):
            gamma[0][c] = gamma[0][c - 1] + Y[c - 1]
            r[0][c] = 2  # Path 2

        # Main loop to compute the minimum alignment
        for d in range(1, len(P[i]) + 1):
            for e in range(1, len(Y) + 1):
                R1 = gamma[d - 1][e] + P[i][d - 1]  # Align P[i][d-1] with empty symbol
                R2 = gamma[d][e - 1] + Y[e - 1]  # Align Y[e-1] with empty symbol
```

```
        R3 = gamma[d - 1][e - 1] + abs(P[i][d - 1] - Y[e - 1])  # Align both symbols

        # Choose the minimum cost and update gamma and path
        gamma[d][e] = min(R1, R2, R3)
        if gamma[d][e] == R1:
            r[d][e] = 1
        elif gamma[d][e] == R2:
            r[d][e] = 2
        else:
            r[d][e] = 3

# Build the new integrated result Y
new_Y = []
d, e = len(P[i]), len(Y)

while d > 0 or e > 0:
    if r[d][e] == 1:
        new_Y.insert(0, P[i][d - 1])  # Insert P[i][d-1] at the start of new_Y
        d -= 1
    elif r[d][e] == 2:
        new_Y.insert(0, Y[e - 1])  # Insert Y[e-1] at the start of new_Y
        e -= 1
    else:
        # Insert the average of both symbols at the start of new_Y
        new_Y.insert(0, (P[i][d - 1] + Y[e - 1]) / 2)
        d -= 1
        e -= 1
```

```
        Y = new_Y  # Update Y with the integrated result
        Z += omega[i]  # Update the total weight

    return Y, Z
```

Algorithm 2: Stopping decision for string object recognition

```
def stopping_decision(m, P, omega, xi, k):
    # Initialize the integrated result Y and total weight Z
    Y = P[0]
    Z = omega[0]

    for i in range(1, m):
        # Integrate the current result P[i] into Y
        Y = [Y[j] + P[i][j] for j in range(len(Y))]
        Z += omega[i]  # Update the total weight

    # Compute the stopping criterion
    delta_m = xi
    for i in range(m):
        delta_m += sum([(Y[j] - P[i][j]) ** 2 for j in range(len(Y))])

    # Compare with the threshold k
    if delta_m <= k:
        return "Stopping"  # Stop the process
    else:
        return "Resume process"  # Continue the process
```

Example usage

```
M = 3
P = [[1, 2, 3], [2, 3, 4], [3, 4, 5]]
omega = [0.5, 0.3, 0.2]
xi = 0.01
k = 1.5

# Execute the first algorithm
Y, Z = integrate_recognition_results(M, P, omega)
print("Integrated results:", Y)
print("Total weight:", Z)

# Execute the second algorithm
decision = stopping_decision(M, P, omega, xi, k)
print("Decision:", decision)
```

Associated software codes

1. Data Preprocessing

Add functions to preprocess input data:

- Normalize sequences to ensure consistency in scale.
- Handle missing values in input sequences (`P`) using imputation techniques.

```
python
def normalize_sequences(P):
    """Normalize sequences to a 0-1 scale."""
    min_val = min(min(seq) for seq in P)
```

```
    max_val = max(max(seq) for seq in P)
    return [[(x - min_val) / (max_val - min_val) for x in seq] for seq in P]
```

2. Interactive Input

Provide a user-friendly way to input data:

- Create a command-line interface (CLI) or graphical user interface (GUI).
- Allow users to specify input sequences, weights, and thresholds interactively.

```
python
def get_user_inputs():
    """Get user inputs for sequences and parameters interactively."""
    M = int(input("Enter the number of sequences (M): "))
    P = []
    for i in range(M):
        sequence = list(map(float, input(f"Enter sequence {i+1} (space-separated):
").split()))
        P.append(sequence)
    omega = list(map(float, input("Enter weights (space-separated): ").split()))
    xi = float(input("Enter xi parameter: "))
    k = float(input("Enter threshold k: "))
    return M, P, omega, xi, k
```

3. Dynamic Visualization

Visualize the integration process dynamically to help users understand intermediate steps and results.

```
python
import matplotlib.pyplot as plt
```

```
from matplotlib.animation import FuncAnimation

def visualize_integration(P, integrated_Y):
    """Visualize the integration process dynamically."""
    fig, ax = plt.subplots()
    ax.set_title("Integration Process")
    ax.set_xlabel("Index")
    ax.set_ylabel("Value")

    lines = [ax.plot(seq, label=f"Sequence {i+1}")[0] for i, seq in enumerate(P)]
    integrated_line, = ax.plot([], [], 'k--', label="Integrated Result", linewidth=2)

    def update(frame):
        if frame < len(integrated_Y):
            integrated_line.set_data(range(len(integrated_Y[frame])),
integrated_Y[frame])
        return integrated_line,

    ani = FuncAnimation(fig, update, frames=len(integrated_Y), repeat=False)
    plt.legend()
    plt.show()
```

4. Batch Processing

Allow batch processing for multiple datasets:

- Accept a folder of input files and process them sequentially.
- Generate consolidated reports for all results.

python

```
import os

def batch_process(folder_path):
    """Process multiple datasets in a folder."""
    results = []
    for file_name in os.listdir(folder_path):
        if file_name.endswith(".json"):
            M, P, omega = load_inputs_from_file(os.path.join(folder_path, file_name))
            Y, Z = integrate_recognition_results(M, P, omega)
            results.append((file_name, Y, Z))
    return results
```

5. Configurable Parameters

Add support for configuration files to manage parameters such as thresholds, weights, or default settings.

```
python
import json

def load_config(filename="config.json"):
    """Load configuration from a JSON file."""
    with open(filename, "r") as f:
        return json.load(f)

def save_config(config, filename="config.json"):
    """Save configuration to a JSON file."""
    with open(filename, "w") as f:
        json.dump(config, f, indent=4)
```

Example usage:

config = {"xi": 0.01, "k": 1.5}

save_config(config)

6. Advanced Stop Criteria

Enhance the stopping decision algorithm with additional conditions:

- Early stopping if changes between consecutive integrations are below a tolerance.
- Consider variance or entropy of the integrated result as a criterion.

python

```
def advanced_stopping_criteria(delta_m, tolerance, iterations, max_iterations):
    """Decide stopping based on advanced criteria."""
    if delta_m <= tolerance:
        return "Stopping - Low Variation"
    elif iterations >= max_iterations:
        return "Stopping - Max Iterations Reached"
    return "Resume process"
```

7. Parallel Execution

Use parallel processing to speed up integration for large datasets.

python

```
from multiprocessing import Pool

def parallel_integration_worker(args):
    """Worker function for parallel integration."""
```

```
    i, P, omega, Y, Z = args
    return integrate_recognition_results(1, [P[i]], [omega[i]])

def parallel_integration(M, P, omega):
    """Run the integration process in parallel."""
    with Pool() as pool:
        results = pool.map(parallel_integration_worker, [(i, P, omega, [], 0) for i in
range(M)])
    # Aggregate results from workers
    Y = results[0][0]
    Z = results[0][1]
    for r in results[1:]:
        Y = [y + r[0][i] for i, y in enumerate(Y)]
        Z += r[1]
    return Y, Z
```

8. Integration with Machine Learning

Use ML to predict stopping conditions or optimize weights:

- Train a model to learn optimal weights (`omega`) based on historical data.
- Use clustering algorithms to group similar sequences before integration.

```
python
from sklearn.cluster import KMeans

def cluster_sequences(P, n_clusters=2):
    """Cluster sequences using K-Means."""
    model = KMeans(n_clusters=n_clusters)
    flattened = [x for seq in P for x in seq]
```

```
    labels = model.fit_predict(flattened)
    clustered_P = [[] for _ in range(n_clusters)]
    for i, seq in enumerate(P):
        clustered_P[labels[i]].append(seq)
    return clustered_P
```

9. Performance Metrics

Add evaluation metrics to assess the quality of results:

- Calculate mean squared error (MSE) or mean absolute error (MAE) between integrated results and ground truth.

```
python
def evaluate_results(integrated_Y, ground_truth):
    """Evaluate the integrated results."""
    mse = sum((a - b) ** 2 for a, b in zip(integrated_Y, ground_truth)) /
len(ground_truth)
    mae = sum(abs(a - b) for a, b in zip(integrated_Y, ground_truth)) /
len(ground_truth)
    return {"MSE": mse, "MAE": mae}
```

10. Logging

Integrate logging to track execution, errors, and performance.

```
```python
import logging

logging.basicConfig(level=logging.INFO)
```
```

```
def log_example():
    logging.info("Starting the integration process")
    try:
        # Example code
        pass
    except Exception as e:
        logging.error(f"Error occurred: {e}")
```

Appendix B. List of Published Works and Information on Approbation

1. X. Guanxiang and V. Kovtun, “The model of the system for objects recognition in the real-time video stream”, Computer systems and information technologies, no. 4, pp. 157–162, Dec. 26, 2024. dooi: 10.31891/csit-2024-4-19.
2. X. Guanxiang and V. Kovtun, “Methods for implementation of string object recognition results in real-time video streams,” Measuring And Computing Devices In Technological Processes, no. 4. Khmelnytskyi National University, pp. 338–347, Nov. 28, 2024. doi: 10.31891/2219-9365-2024-80-41.
3. X. Guanxiang and V. Kovtun, “The method for stopping the string object recognition process in a real-time video stream,” Measuring And Computing Devices In Technological Processes, no. 3. Khmelnytskyi National University, pp. 243–250, Aug. 29, 2024. doi: 10.31891/2219-9365-2024-79-32.
4. X. Guanxiang, “A Study of the Implementation Techniques and Processes of Live Video Courses in Online Teaching and Learning,” Journal of Online Learning, no. 2, pp. 3224–3225, Jun. 29, 2022. doi: 10.12255/j.issn.1672-6677.2021.11.1603.
5. X. Guanxiang, “Post-Processing Software Application Techniques in the Audio-visual Recording of Hunan Chinese Dialect Survey for the Language Protection Project - Taking Adobe Premiere Pro as an Example,” Journal of Audio-visual Studies, no. 5, pp. 162–166, Apr. 15, 2022. [Online]. Available: https://d.wanfangdata.com.cn/periodical/Ch9QZXJpb2RpY2FsQ0hJTmV3UzIwMjUwMTA0MTcwMjI2EhNzbXNqLXNoYW5nMjAyMjA3MDQ3GghkcGk0Zm45NA%3D%3D.
6. X. Guanxiang, “Preliminary quality control in audio-visual recording of Hunan Chinese dialect investigation of China language resources protection project,” Journal of Language Protection, no. 3, pp. 241–243, Mar. 15, 2022. [Online]. Available: https://d.wanfangdata.com.cn/periodical/Ch9QZXJpb2RpY2FsQ0hJTmV3UzIwMjUwMTA0MTcwMjI2EhpRS0JKQkQyMDIyMjAyMjAzMTgwMDAwNjk0NBoIZHBpNGZuOTQ%3D.

7. X. Guanxiang, “Video and audio recording technology in Hunan Chinese dialect investigation of China language resource protection project,” Journal of Media Technology, no. 8, pp. 39–45, Aug. 02, 2019. doi: 10.3969/j.issn.2096-0751.2019.02.009.

8. X. Guanxiang, “Analysis of digital film and television special effects production technology,” Journal of Modern Information, no. 3, pp. 48–50, Jun. 16, 2017. doi: 10.3969/j.issn.1674-6708.2017.06.040.