People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

University Larbi Tébessi - Tébessa

Faculty of Exact Sciences and Natural and Life Sciences

Department: Mathematics and Computer Science

Final thesis

For obtaining of the MASTER diploma

Domain: Mathematics and Computer Science

Field: Computer Science

Specialty: Systems and Multimedia

Segmentation Based Method For Handwritten Connected Digits Using Image Encryption Technique

Presented by:

Daani Tahar

In front of the jury:

Pr Amroune Mohammed	Pr Larbi Tebessi University	President
Mr. Zebdi Abdelmoumen	MAA Larbi Tebessi University	Thesis Supervisor
Dr. Abbes Faycel	MCB Larbi Tebessi University	Examiner

College year:

2022/2023

Thank

First of all, I would like to express my gratitude to the editors of this dissertation, Mr. Zebdi Abdelmoumen, for their patience, availability, and, above all, their sound advice. Their understanding was particularly valuable, as it enabled us to make significant progress and perform exceptionally well throughout these months.

I would also like to thank the members of the jury:

- Pr: Amroune Mohammed
- Dr: Abbes Faycel

I would also like to thank all the students with whom I have been happy to study during these years of university.

I thank all the teachers that I have been able to meet and benefit from.

I would like to thank my colleagues in the Mathematics and Computer Science department and all those who have helped me directly or indirectly in the realization and accomplishment of this work.

ملخص

يندرج هذا المشروع ضمن مجال تعلم العمق، وهو فرع من فروع الذكاء الاصطناعي الذي يركز على تطوير النماذج الحاسوبية التي تتعلم وتتطور ذاتيًا من خلال البيانات. يهدف هذا المشروع إلى تحسين عملية التعرف على الأرقام باستخدام تقنيات تعلم العمق.

تعتبر فوائد التعرف على الأرقام باستخدام تعلم العمق هامة ومتعددة. على سبيل المثال، يمكن استخدامها في تطبيقات التعرف على الخط اليدوي والتعرف على الأعداد في الصور وفهم المعلومات الرقمية الموجودة في الصور والوثائق. يمكن أيضًا استخدامها في تحسين الأمان والتحقق من الهوية من خلال تعرف الأجهزة على الأرقام السرية أو الرموز المكونة من الأرقام.

في هذا المشروع، تم استخدام نموذج(CNN (Convolutional Neural Network ، حيث تم تدريب النموذج باستخدام مجموعة البيانات CVL التي تحتوي على صور للأرقام الفردية.

ثم تم استخدام طريقة تجزئة الصورة، حيث تم تقسيم سلسلة الأرقام إلى أجزاء فردية و تمرير هذه الاجزاء على نظام التعرف على الصور و ذلك لتنبأ بكل رقم باستخدام نموذج CNN المدرب. وأخيرًا، تم جمع الأرقام المتوقعة للحصول على الرقم الكامل للسلسلة.

تمت عملية التجزئة باستخدام بعض الطرق المقترحة في عملنا مثل تحديد الحواف و النافذة المنزلقة و تابعنا النتائج المتحصل عليها بعدها قمنا باجراء طريقة اخرى لتجزئة و هي دمج بين الطريقتين المقترحتين و ذلك لتحسين من جودة العملية.

وبهدف زيادة جودة التجزئة، تم استخدام طريقة تشفير الصورة لتحسين عملية التجزئة حيث كانت هذه الطريقة تعتمد على تشفير البكسلات المكونة للصورة الخاصة بالأعداد المتصلة حيث تتكون لنا صورة جديدة و نطبق العملية على مجموعة البيانات للارقام المتصلة الخاصة بنا لتتكون لنا مجموعة بيانات جديدة و بعدها نقوم بتطبيق نفس طرق التجزئة المقترحة و نتابع النتائج اذا حققت تحسن او لا.

Abstract

This project falls within the field of deep learning, which is a branch of artificial intelligence that focuses on developing computer models that learn and evolve autonomously through data. The aim of this project is to improve the process of number recognition using deep learning techniques.

The benefits of number recognition using deep learning are significant and varied. For example, it can be used in applications such as handwriting recognition, number recognition in images, understanding digital information present in images and documents. It can also be used to enhance security and identity verification by devices recognizing secret numbers or numeric codes.

In this project, a Convolutional Neural Network (CNN) model was used. The model was trained using the CVL dataset, which contains images of individual numbers.

Then an image segmentation method was employed, where the number sequence was divided into individual parts and passed through the image recognition system to predict each number using the trained CNN model. Finally, the predicted numbers were combined to obtain the complete number sequence.

The segmentation process was performed using several proposed methods in our work, such as edge detection and sliding window. We evaluated the results and then applied another method, which was a combination of the two proposed methods, to improve the quality of the process.

To enhance the quality of segmentation, an image encryption method was used. This method aimed to improve the segmentation process by encrypting the pixels of the image corresponding to the connected numbers. This resulted in a new image, and the process was applied to our dataset of connected numbers, resulting in a new dataset. Then, the same proposed segmentation methods were applied, and the results were evaluated for improvement.

Résumé

Ce projet fait partie du domaine de l'apprentissage en profondeur, une branche de l'intelligence artificielle qui se concentre sur le développement de modèles informatiques capables d'apprendre et de s'améliorer de manière autonome à partir des données. L'objectif de ce projet est d'améliorer le processus de reconnaissance des chiffres en utilisant des techniques d'apprentissage en profondeur.

Les avantages de la reconnaissance des chiffres grâce à l'apprentissage en profondeur sont importants et variés. Par exemple, cela peut être utilisé dans des applications de reconnaissance d'écriture manuscrite, de reconnaissance de chiffres dans des images, de compréhension des informations numériques présentes dans des images et des documents. Cela peut également être utilisé pour améliorer la sécurité et la vérification de l'identité en reconnaissant des numéros secrets ou des codes composés de chiffres.

Dans ce projet, un modèle CNN (Convolutional Neural Network) a été utilisé. Le modèle a été entraîné en utilisant l'ensemble de données CVL, qui contient des images de chiffres individuels.

Ensuite, une méthode de découpage d'image a été utilisée. La séquence de chiffres a été divisée en parties individuelles, et ces parties ont été passées à un système de reconnaissance d'images pour prédire chaque chiffre à l'aide du modèle CNN entraîné. Enfin, les chiffres prévus ont été collectés pour obtenir le numéro complet de la séquence.

Le découpage a été effectué en utilisant certaines méthodes proposées dans notre travail, telles que la détection des contours et la fenêtre glissante. Nous avons ensuite effectué une autre méthode de découpage en combinant les deux méthodes proposées afin d'améliorer la qualité du processus.

Dans le but d'améliorer la qualité du découpage, une méthode de codage d'image a été utilisée pour améliorer le processus de découpage. Cette méthode était basée sur le codage des pixels composant l'image des chiffres connectés, créant ainsi une nouvelle image. Nous avons appliqué cette méthode à notre ensemble de données de chiffres connectés pour obtenir un nouvel ensemble de données, puis nous avons appliqué les mêmes méthodes de découpage proposées et suivi les résultats pour évaluer s'ils montraient une amélioration ou non.

Table of Contents

General Introduction

Deep learning has emerged as a prominent field within the realm of artificial intelligence. It focuses on the development of computer models that can autonomously learn from and improve on data. Where is our project of deep learning In the domain of digit recognition, which holds significant importance in various areas of research and practical implementation.

Recognizing and accurately interpreting numbers has numerous applications across different fields. It plays a pivotal role in handwriting recognition, enabling the extraction of valuable information from handwritten documents. Additionally, it aids in identity verification and security systems by recognizing and validating numeric codes or secret numbers. Moreover, digit recognition contributes to the analysis of digital images by enabling the extraction of numerical data embedded within images or documents.

This thesis consists of four chapters, each covering essential aspects of the project. The first chapter delves into the domain of handwritten digit recognition. It explores the challenges and techniques employed to recognize and interpret handwritten numbers effectively.

The second chapter provides a comprehensive overview of the state-of-the-art approaches and prior works in the field of digit recognition. It highlights the advancements, methodologies, and breakthroughs that have been achieved thus far, setting the stage for the current research.

The third chapter focuses on the methods and techniques employed in this project. It discusses the utilization of convolutional neural networks (CNN) as a powerful tool for image processing and recognition. Additionally, it elaborates on the image segmentation techniques used to partition digit images into individual components for accurate prediction.

In the fourth chapter, the experimental results obtained from the implemented methods are presented and analyzed. The accuracy and performance of the developed model in recognizing handwritten digits are evaluated and discussed.

Finally, the thesis concludes with a summary of the key findings, contributions, and implications of the research. The conclusion provides a concise overview of the achieved results and their significance in enhancing digit recognition through deep learning techniques.

Overall, this thesis aims to contribute to the field of digit recognition by leveraging the power of deep learning and exploring novel methodologies to improve accuracy and efficiency in recognizing handwritten digits.

Chapter1: Theoretical concepts

1. Introduction:

Handwritten number recognition is a challenging task that has garnered significant attention in the field of deep learning. The ability to accurately identify and interpret a sequence of handwritten numbers has wide-ranging applications, including optical character recognition, digitized document analysis, and automated data extraction. Deep learning techniques have shown remarkable success in tackling this problem by leveraging the power of artificial neural networks to learn complex patterns and features inherent in handwritten digits.

2. Handwritten Digit Recognition:

Handwritten digit recognition refers to the task of automatically identifying and classifying handwritten digits. It is an important problem in computer vision and machine learning. Deep learning techniques, specifically convolutional neural networks (CNNs), have greatly improved the accuracy of handwritten digit recognition. [1]

Handwriting recognition systems can be categorized into two types: online and offline recognition.

Online Handwriting Recognition:

Online handwriting recognition involves the real-time conversion of a user's handwritten input on a tablet or smartphone. This process requires immediate recognition and interpretation of the handwritten strokes as they are being drawn. It is considered more challenging compared to offline recognition due to the need for real-time processing and interpretation. [2]

Offline Handwriting Recognition:

Offline handwriting recognition, on the other hand, is a comparatively easier process. It involves the conversion of scanned images or digital representations of handwritten text into computer-readable text. This recognition method does not require real-time processing as it operates on pre-existing images or documents. The scanned or digitalized handwritten content is analyzed and translated into text using machine learning algorithms. [2]

In summary, online handwriting recognition deals with the immediate interpretation of handwritten input in real-time, while offline handwriting recognition focuses on the conversion of scanned or digitalized handwritten text into computer-readable format. [2]

3. Convolutional Neural Network:

A Convolutional Neural Network (CNN) is a deep learning algorithm commonly used for image analysis and computer vision tasks. It is designed to automatically learn and extract relevant features from images through multiple layers of interconnected neurons. CNNs utilize convolutional layers to apply filters and detect patterns in local regions of the input data, followed by pooling layers to reduce spatial dimensions and increase computational efficiency. The extracted features are then fed into fully connected layers for classification or regression [3].

4. CNN layers:

In a Convolutional Neural Network (CNN), the network architecture is composed of several layers that perform different operations on the input data. These layers include:

4.1. The Convolutional Layer:

The convolutional layer is a key component of CNNs used in deep learning. It applies filters to the input data, extracting features and patterns. It captures spatial hierarchies and local patterns, enhancing the network's ability to learn meaningful representations. The output is passed through activation functions to introduce non-linearities [4].

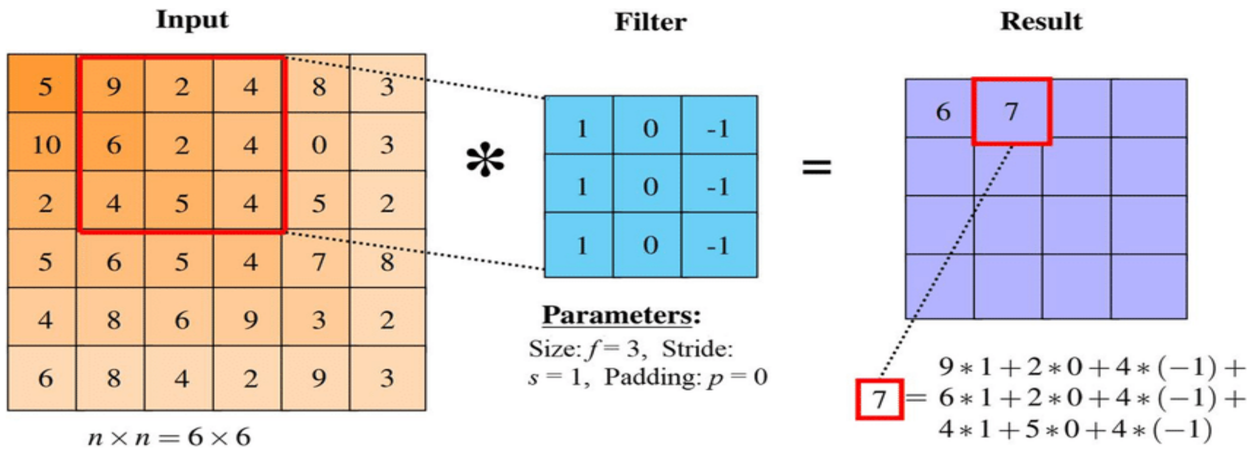


Figure 1: Convolution layer [5]

4.2. The Pooling Layer

The Pooling Layer is another important component of CNNs used in deep learning. It performs down sampling by reducing the spatial dimensions of the input, effectively reducing the number of parameters and computations in the network. It helps in extracting the most important features and improving the network's robustness to variations in input. Common pooling operations include max pooling and average pooling. [4]

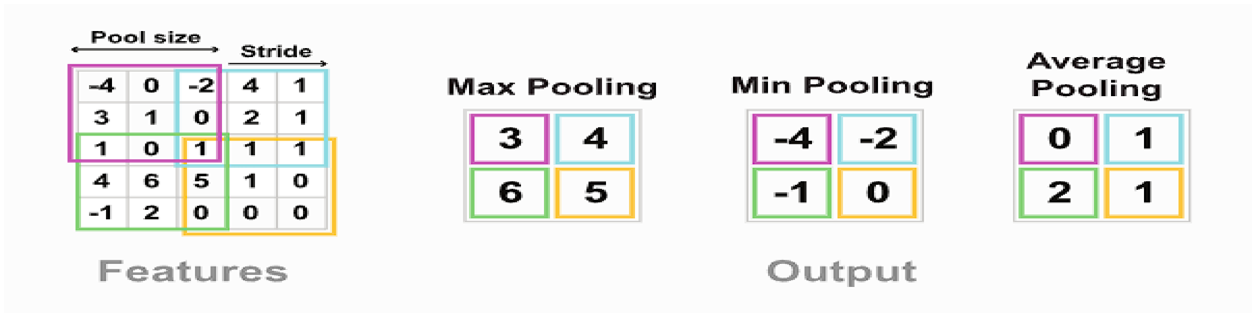


Figure 2: pooling layer [5]

4.3. Activation Layer:

The Activation Layer, also known as the non-linear layer, is a fundamental component in convolutional neural networks (CNNs). It introduces non-linearity into the network by applying a mathematical function element-wise to the output of the previous layer. This allows the network to learn complex, non-linear relationships between the input and output.

The Activation Layer is essential for enabling the network to model and capture non-linear patterns and make more expressive representations of the data. Popular activation functions include the Rectified Linear Unit (ReLU), sigmoid, and hyperbolic tangent (tanh) [4].

4.4. Fully Connected Layer:

The Fully Connected Layer, also known as the dense layer, is a type of layer commonly used in neural networks, including convolutional neural networks (CNNs). In this layer, every neuron is connected to every neuron in the previous layer, forming a fully connected graph. Each connection is associated with a weight that determines the strength of the connection.

The Fully Connected Layer performs computations on the input data by applying matrix multiplication between the input and the weights, followed by the application of a bias term and an activation function. This process allows the network to learn complex relationships and make predictions based on the learned features.

The Fully Connected Layer is typically used at the end of the network to transform the learned features into a final output. It is commonly used for tasks such as classification or regression. [4]

5. Existing architecture:

There are several existing CNN architectures that have been developed and widely used in deep learning applications. Here are some notable ones:

5.1.VGGNet:

VGGNet, or the Visual Geometry Group Network, is a convolutional neural network (CNN) architecture introduced in 2014. It is characterized by its deep structure, consisting of 16 or 19 layers, all of which are convolutional layers, followed by fully connected layers. VGGNet focuses on using small 3x3 filters throughout the network, which leads to better feature representation. It achieved outstanding performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014. [6]

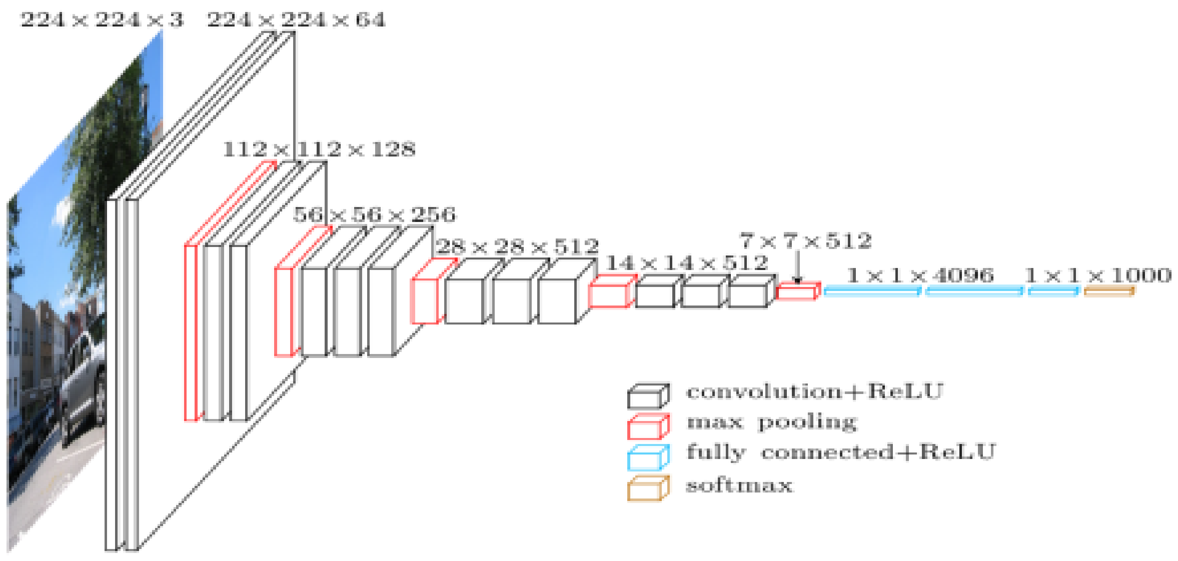


Figure 3: VGGNet architecture [7]

5.2.AlexNet:

Alex Net is a groundbreaking convolutional neural network (CNN) architecture introduced in 2012. It consists of eight layers, including five convolutional layers, three fully connected layers, and employs techniques like ReLU activation, max-pooling, local response normalization, and dropout regularization. Alex Net achieved significant advancements in image classification and was trained on the ImageNet dataset. [8].

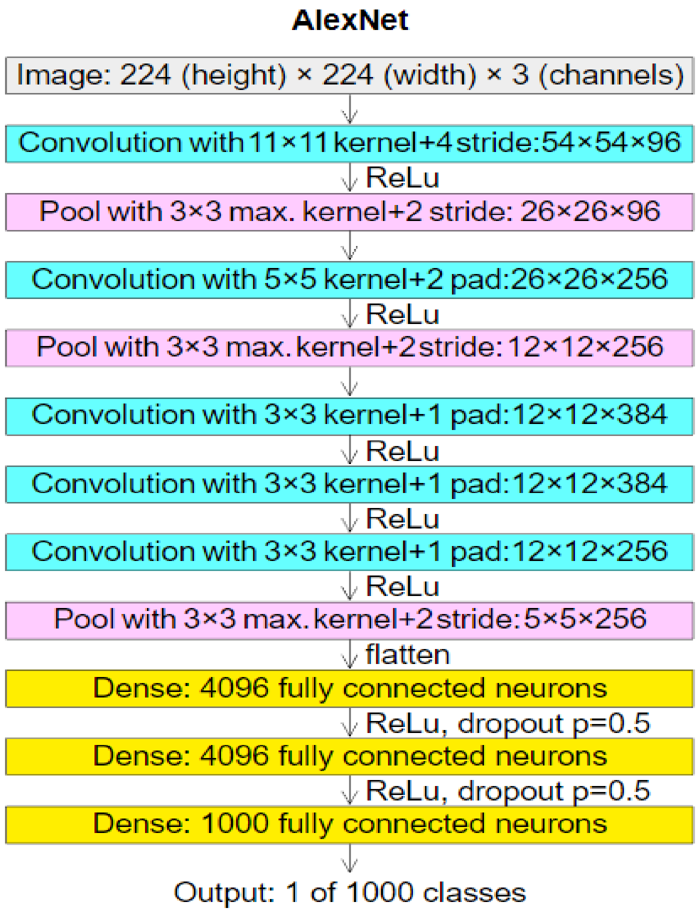

AlexNet
Image: 224 (height) × 224 (width) × 3 (channels)
Convolution with 11×11 kernel+4 stride:54×54×96
ReLu
Pool with 3×3 max. kernel+2 stride: 26×26×96
Convolution with 5×5 kernel+2 pad:26×26×256
ReLu
Pool with 3×3 max. kernel+2 stride: 12×12×256
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×256
ReLu
Pool with 3×3 max. kernel+2 stride:5×5×256
flatten
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 1000 fully connected neurons
Output: 1 of 1000 classes

Figure 4: AlexNet architecture [9]

5.3.ResNet:

ResNet, short for Residual Network, is a widely used deep learning architecture that revolutionized the field of image recognition. It introduced the concept of residual connections, which allow the network to bypass and learn residual mappings. This enables the training of extremely deep networks with improved gradient flow and reduced vanishing gradient problem.

ResNet has demonstrated outstanding performance on various computer vision tasks, surpassing previous models in accuracy and achieving state-of-the-art results. [10]

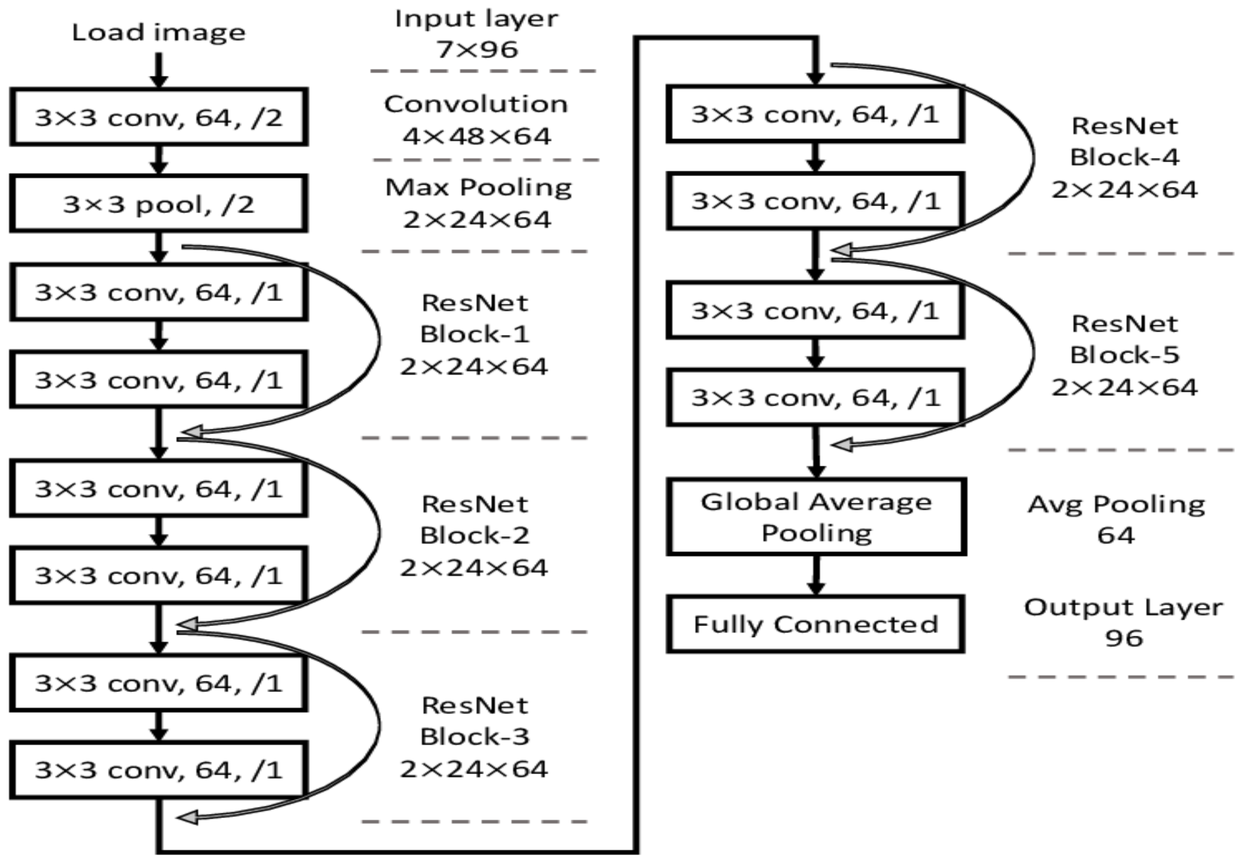


Figure 5: ResNet architecture [11]

6. CNN training methods:

6.1.Training from scratch:

Training a model from scratch refers to training a deep learning model from randomly initialized weights, without utilizing any pre-trained models or transfer learning. It involves training the model on a specific dataset starting from the beginning, learning the representations and features directly from the data.

6.2.Transfer learning:

Transfer learning refers to a technique in deep learning where a pre-trained model, which has been trained on a large dataset, is used as a starting point for training a new model on a different but related task or dataset. By leveraging the knowledge and learned features from the pre-trained model, transfer learning can significantly speed up the training process and improve the performance of the new model.

7. Datasets:

Datasets refer to collections of structured or unstructured data that are used for training and evaluating machine learning models. They play a crucial role in the development and advancement of various AI applications.

7.1.MNIST:

MNIST stands for "Modified National Institute of Standards and Technology." It is a well-known dataset in the field of machine learning and computer vision. The dataset consists of a collection of grayscale digital images representing hand-written numbers from 0 to 9 [12].

MNIST is typically composed of 60,000 images used for training and 10,000 images used for validation and testing. The images are sized at 28x28 pixels and are converted into a digital dataset with values ranging from 0 to 255, where white color represents a value of 0, and black color represents a value of 255.

The MNIST dataset is widely used for developing and testing machine learning models in the field of optical image classification. The main objective is to correctly identify and classify the handwritten numbers in the images.

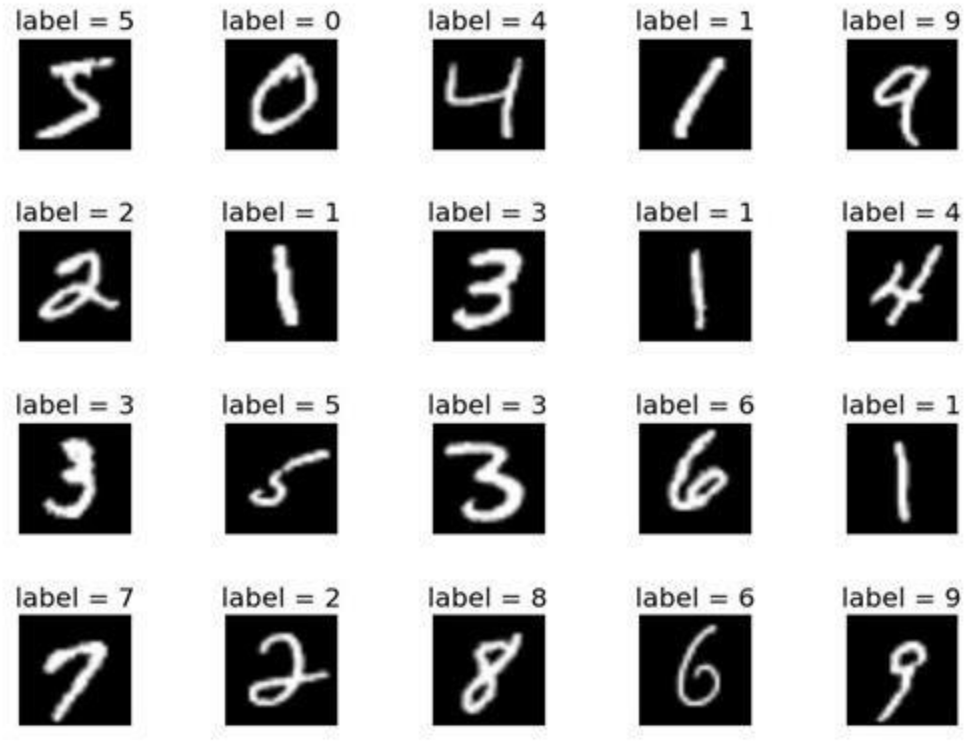


Figure 6: The MNIST dataset [13]

7.2.CVL:

The CVL Digit Strings dataset is a collection of digit strings extracted from the CVL Single Digit dataset. It includes 10 different digit strings written by approximately 120 different writers. This dataset contains a total of 1,262 training images.

The CVL Single Digit dataset, on the other hand, consists of 7,000 single digits, with 700 digits per class. These single digits were extracted from the digit strings in the CVL Digit Strings dataset. The CVL Single Digit dataset provides a focused collection of individual digits for training and recognition purposes.

It is important to note that the validation set, which has the same size as the training set, consists of different writers. The validation set is primarily used for parameter estimation and validation, rather than supervised training. This ensures that the models are evaluated on unseen data, enhancing the reliability of the validation results.

In summary, the CVL Digit Strings dataset contains digit strings written by various writers, while the CVL Single Digit dataset is derived from these strings and consists of individual digits. Both datasets play a crucial role in training and evaluating models for handwritten digit recognition [14].

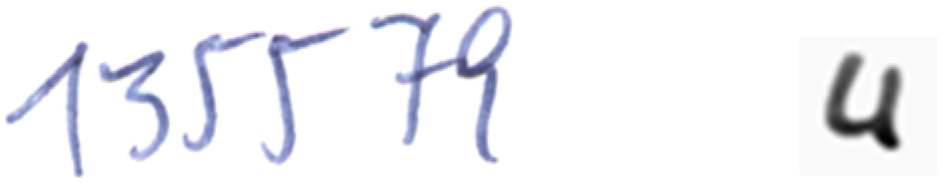


Figure 7: cvl dataset [14]

7.3.ARDIS:

This is a new image-based handwritten historical digit dataset named ARDIS (Arkiv Digital Sweden). The images in ARDIS dataset are extracted from 15.000 Swedish church records which were written by different priests with various handwriting styles in the nineteenth and twentieth centuries. The constructed dataset consists of three single digit datasets and one digit strings dataset. The digit strings dataset includes 10.000 samples in Red-Green-Blue (RGB) color space, whereas, the other datasets contain 7.600 single digit images in different color spaces [15].

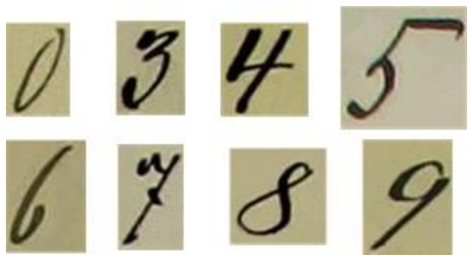


Figure 8: ARDIS dataset [15]

7.4.ORAND-CAR:

The ORAND-CAR-2014 dataset refers to the "Optical Recognition of Arabic Numerals for Cars - 2014" dataset. It is a specific dataset focused on Arabic digit recognition specifically in the context of car license plates.

The ORAND-CAR-2014 dataset contains images of Arabic digits captured from car license plates. It includes various images representing different Arabic numerals (0-9) found on license plates. The dataset is annotated, meaning that it provides labels or ground truth for the Arabic digits present in the images [16].

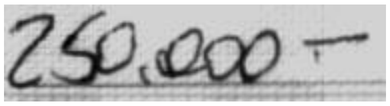


Figure 9: ORAND-CAR dataset [16]

7.5.SVHN

Street View House Numbers (SVHN) is a digit classification benchmark dataset that contains 600,000 32×32 RGB images of printed digits (from 0 to 9) cropped from pictures of house number plates. The cropped images are centered in the digit of interest, but nearby digits and other distractors are kept in the image. SVHN has three sets: training, testing sets and an extra set with 530,000 images that are less difficult and can be used for helping with the training process [17].

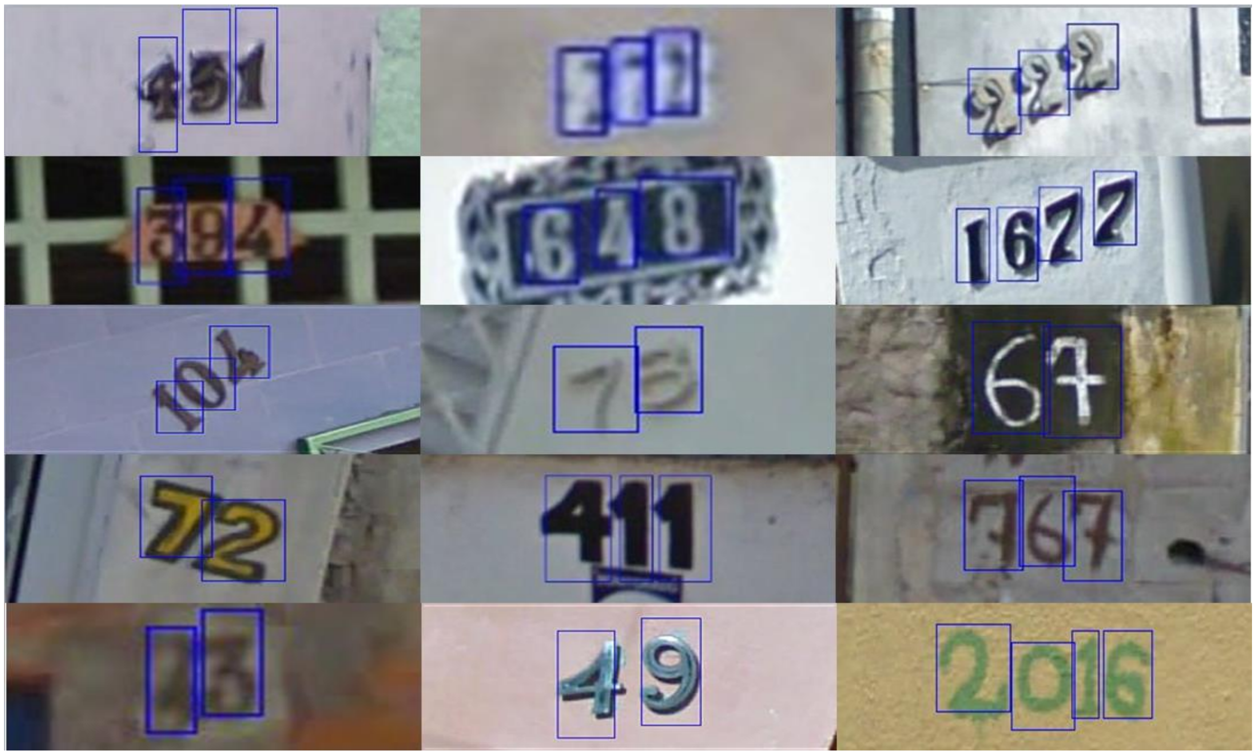


Figure 10: SVHN dataset [17]

8. Pre-processing:

Pre-processing plays a crucial role in the field of Handwritten Document Recognition as it serves as the initial and fundamental step. Its primary purpose is to optimize the performance of recognition algorithms by addressing various objectives.

The main goals of pre-processing include noise reduction in handwritten data, error correction, and extraction of essential information from the representative form. By performing these tasks, the pre-processing stage aims to enhance the quality of the input data before it is fed into the recognition system [18].

The impact of pre-processing on the overall recognition performance should not be underestimated. As there are multiple steps involved in achieving successful recognition, it is important to highlight some of these steps:

8.1.Noise reduction:

In order to enhance discrimination, noise reduction techniques such as dropping, smoothing, and correction of outliers with significant angular variations are employed, as outlined in the paper referenced as [19]. The primary objective is to minimize noise interference to the maximum extent possible.

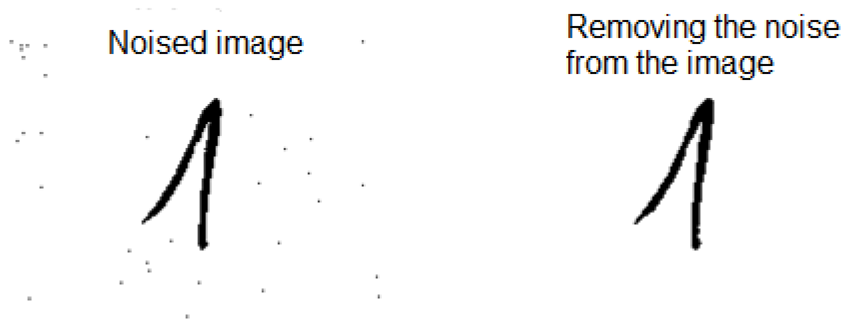


Figure 11: Noise Reduction

8.2.Binarization:

The process of Binarization involves converting a grayscale image, which consists of 0 to 255 gray levels, into a binary image with only two values, black (0) and white (1). The success of Handwritten Document Recognition heavily relies on the effectiveness of the binarization technique. A high-quality binarized image can significantly improve accuracy compared to the original image, as it helps eliminate the noise present in the original image [20].

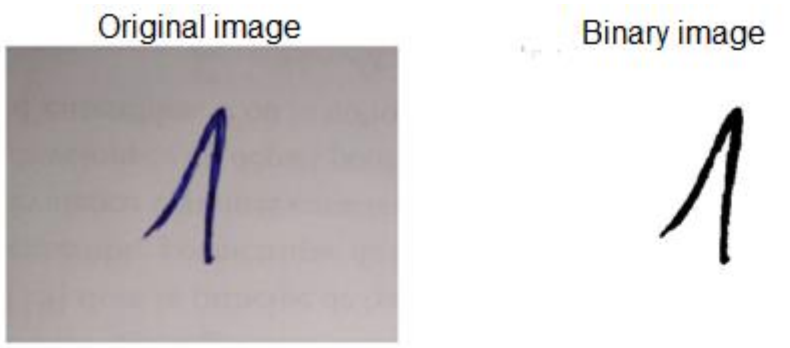


Figure 12: Binarization

8.3.Skeletonization:

Skeletonization, also referred to as thinning, holds great significance as a crucial step in the pre-processing stage. It plays a vital role in various applications such as Optical Character Recognition and writer identification.

The skeletonization process serves multiple purposes, including reducing and compacting the image size while preserving the overall shape of the object. It achieves this by identifying a median axis, which consists of pixels that have equal distances from the surrounding border pixels [21].

The utilization of skeletal representation offers several advantages:

- It provides an effective approach to represent the structural relationships between different components of the model.
- It finds wide application in systems designed for recognizing characters, words, signatures, and imprints.

By employing skeletonization, the pre-processed data becomes more compact, while retaining essential shape information and enabling subsequent recognition algorithms to focus on the structural characteristics of the objects of interest.

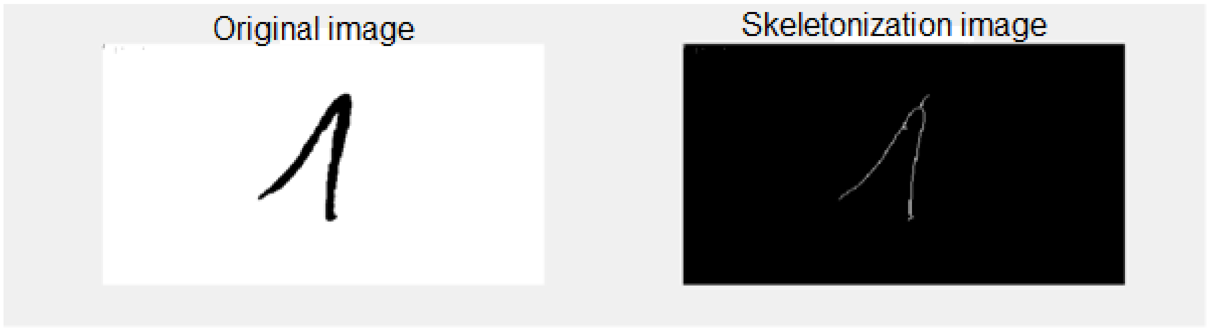


Figure 13: Skeletonization

8.4.Size normalization:

In order to facilitate the recognition stage in character recognition, it is essential to normalize the size of the digits using sampling techniques. Size normalization serves as a significant pre-processing technique in this context.

However, the selection of an appropriate size is a crucial consideration. If the size is too small, there is a risk of losing important information. On the other hand, if the size is too large, it may result in slower processing during the recognition stage [22].

Finding the optimal balance in size normalization is vital to ensure that the normalized digits retain the necessary details for accurate recognition while maintaining efficiency in the subsequent recognition algorithms.

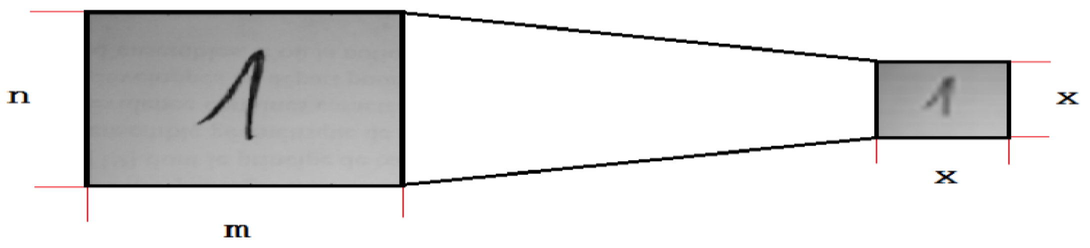


Figure 14: Size normalization

9. Segmentation:

The segmentation stage is a critical step in handwriting digit recognition. It involves dividing a handwritten document into individual digits or character components, which are then analyzed and recognized separately. Proper segmentation is essential to ensure accurate recognition of each digit.

One approach to segmentation in handwriting digit recognition is based on connected component analysis. This method identifies and separates individual components based on their connectivity and spatial relationships. Various algorithms and techniques have been proposed for segmentation, such as contour-based segmentation, region-growing segmentation, or stroke-based segmentation [23].

There are two main categories of segmentation:

- Explicit segmentation
- Implicit segmentation

9.1.Explicit segmentation

Explicit segmentation refers to the process of dividing an entity or a dataset into distinct and well-defined segments based on predefined criteria or characteristics. This segmentation approach relies on explicit information or features to determine the boundaries between segments. For example, in image analysis, explicit segmentation involves manually drawing outlines or regions of interest around specific objects or areas within the image [24].

- Local minima of the upper contour.
- Spaces between characters or sub words.
- The most likely intersection points by an analysis of the components in the word.

9.2.Implicit segmentation

Implicit segmentation, involves dividing an entity or dataset into segments without relying on explicit criteria or predefined features. It leverages patterns, relationships, or underlying structures within the data itself to identify meaningful segments. Implicit segmentation techniques often utilize statistical algorithms, machine learning, or clustering algorithms to discover patterns and group similar data points together [24].

- The advantage of this segmentation is that the information is localized by the models of the letters and the validation is done by its models. There will be no segmentation error and finally we get around Sayre's dilemma because by knowing the letters, we do not generate segmentation error.

- The shortcomings of this segmentation come from the fact that the searching space for the limits is greatly increased and the problem is reduced to a problem of searching for areas where these limits are found.

Unlike explicit segmentation, implicit segmentation does not require prior knowledge or explicit labeling of segments. Instead, it relies on the inherent properties of the data to reveal underlying segments. This approach can be useful when dealing with large datasets or complex data where manually defining explicit criteria for segmentation is challenging or impractical [24].

Both explicit and implicit segmentation have their advantages and applications depending on the specific context and requirements. Explicit segmentation provides precise control over segment boundaries, making it suitable for tasks where specific regions of interest need to be delineated. Implicit segmentation, on the other hand, allows for the discovery of hidden patterns and structures within the data, enabling unsupervised learning and exploratory analysis [24].

10. Conclusion:

In conclusion, the first chapter of the study focused on the utilization of convolutional neural networks (CNNs) in recognizing handwritten numbers through deep learning. The significance of this task was highlighted, and the role of deep learning in achieving precise outcomes was discussed. The architecture of CNNs, encompassing convolutional, pooling, activation, and fully connected layers, was elucidated. The effectiveness of CNNs in classifying and recognizing handwritten numbers was demonstrated through the utilization of extensive data sets. Furthermore, prominent data sets in this domain were mentioned. The chapter also covered image pre-processing techniques and various methods and categories of segmentation. This chapter establishes the foundation for subsequent chapters, which will delve into specific techniques and experimental results pertinent to the project.

Chapter 2: State of the art

1. Introduction:

Recent advancements in handwritten digit recognition have led to the development of state-of-the-art techniques. These techniques leverage deep learning, particularly convolutional neural networks (CNNs), to achieve highly accurate and efficient classification of handwritten digits. State-of-the-art approaches utilize large-scale datasets to benchmark performance and compare different models. This chapter explores the current state of the art in handwritten digit recognition, highlighting key techniques, architectures, and datasets used to achieve superior results. Understanding the current state of the art provides valuable insights and inspiration for further advancements in this field.

2. Related works:

2.1. Y. Lecun et al [25] 1998:

The researchers [25] presented a model in this project for recognizing handwritten digits using deep learning techniques.

The well-known MNIST dataset was used in this study, which consists of 60,000 training images and 10,000 test images. A deep neural network with three layers was employed to recognize the handwritten digits.

The results of the study yielded excellent accuracy in recognizing handwritten digits. A test accuracy of 99.05% was achieved, making this model one of the best models for handwritten digit recognition at that time. This work demonstrated that the use of deep neural networks can be effective in improving the accuracy of recognizing handwritten digits.

2.2. P. Sermanet et al [26] 2012:

In this project [26], they utilized convolutional neural networks (ConvNets) to classify digits found in real-world house numbers. ConvNets are neural networks that employ a hierarchical feature learning structure inspired by biological systems. Unlike many conventional vision approaches that require manual design, ConvNets have the ability to automatically learn a distinct set of features specifically optimized for a given task.

To enhance the traditional ConvNet architecture, they incorporated the learning of multi-stage features and employed Lp pooling. This resulted in a significant improvement in accuracy, achieving a new state-of-the-art performance of 95.10% on the SVHN dataset, which represents a 48% reduction in error. Additionally, they conducted an analysis to evaluate the advantages of different pooling methods and the utilization of multi-stage features within ConvNets.

2.3. A. Gattal et al [27] 2016:

This research work [27] illustrates the effective utilization of a combination of oriented Basic Image Features (oBIFs) and background concavity features to enhance the performance of isolated digit recognition systems. The features are extracted without size normalization from both the entire image and various regions of the image by employing a uniform grid sampling approach. Classification is performed using a one-against-all support vector machine (SVM) classifier. The experimental study is conducted on the standard CVL single digit database. Through a series of evaluations using different feature configurations and combinations, the proposed approach achieves high recognition rates, which are then compared to the state-of-the-art methods in the field.

2.4. U. R. Babu et al [28] 2016:

This research paper [28] introduces a novel approach for off-line handwritten digit recognition that relies on structural features, eliminating the need for thinning operations and size normalization techniques. Four different types of structural features are employed in this study, namely the number of holes, water reservoirs in four directions, maximum profile distances in four directions, and fill-hole density. These features are utilized for digit recognition, and the effectiveness of the system heavily relies on the selection of features. The primary objective of this paper is to present efficient and reliable techniques for recognizing handwritten digits. The Euclidean minimum distance criterion is employed to determine minimum distances, and a k-nearest neighbor classifier is used for digit classification. The MNIST database is utilized for both training and testing the system. A total of 5000 numeral images are tested, achieving a recognition rate of 96.94%.

2.5.A. Khan et al [29]2017:

In the study [29] , a novel approach called Multiple-Cell Size (MCS) is proposed for efficient classification of Handwritten Digits using Histogram of Oriented Gradient (HOG) features and a Support Vector Machine (SVM) based classifier. The effectiveness of the HOG-based technique highly depends on the selection of cell size during feature extraction. To address this, the MCS approach is introduced to perform HOG analysis and compute the corresponding features. The system is evaluated on the Benchmark MNIST Digit Database, which contains handwritten digits, achieving an impressive classification accuracy of 99.36% using an Independent Test set strategy. Cross-Validation analysis is also conducted using the 10-Fold Cross-Validation strategy, resulting in a 10-Fold classification accuracy of 99.26%. Notably, the proposed system outperforms existing techniques that rely on complex procedures, as it achieves comparable or even better results while utilizing simpler operations in both the Feature Space and the Classifier Space. The Confusion Matrix and Receiver Operating Characteristics (ROC) plots of the system demonstrate the superior performance of the novel MCS HOG and SVM based digit classification system.

2.6.K. T. Islam et al [30] 2017:

In this research [30], we developed a multi-layer fully connected neural network with a single hidden layer to achieve handwritten digit recognition. The testing phase was carried out using the widely available MNIST handwritten database. For training purposes, we extracted 28,000 digit images from the MNIST database, while 14,000 digit images were reserved for testing. The performance of our multi-layer artificial neural network resulted in an impressive accuracy of 99.60% during the testing phase.

2.7.A. Gattal et al [31] 2017:

The objective of the research [31] is to propose a segmentation and recognition system for handwritten digit strings of unknown length. The system combines multiple explicit segmentation methods based on the configuration link between digits. Specifically, three segmentation methods, namely histogram of the vertical projection, contour analysis, and sliding window Radon transform, are integrated. To analyze and determine the acceptance or rejection of each segmented digit image, a recognition and verification module employing support vector machine classifiers is utilized. Additionally, the proposed system includes various submodules to enhance its robustness. Experimental results obtained from the benchmark dataset demonstrate the effectiveness of the proposed system in segmenting handwritten digit strings without prior knowledge of their length, showcasing its superiority compared to state-of-the-art methods.

2.8.Shujing Lyu et al [32] 2018:

This research [32] introduces a novel CNN-based architecture designed specifically for recognizing handwritten digit sequences. The study was conducted using two datasets: CVL and ORAND-CAR. The results of the proposed approach yielded an accuracy of 42.69% on the CVL dataset, while achieving 92.2% and 94.02% accuracy on the ORAND-CAR-A and ORAND-CAR-B datasets, respectively.

2.9.Rabia KARAKAYA et al [33]2018:

As technology continues to play an increasingly prominent role in our daily lives, algorithms have become integral components that streamline tasks and reduce workloads. Machine learning algorithms, in particular, have been steadily improving by emulating human behaviors. Handwriting recognition systems are notable examples of this progress. In this study [33], the process of handwriting digit recognition was conducted using various algorithms with different working methods. These algorithms include Support Vector Machine (SVM), Decision Tree, Random Forest, Artificial Neural Networks (ANN), K-Nearest Neighbor (KNN), and K-Means Algorithm. The operational principles of the handwriting digit recognition process were examined, and the effectiveness of different algorithms on the same database was evaluated. A comprehensive report was generated by comparing the accuracy achieved by each algorithm.

2.10. R. Dey et al [34] 2020:

In this study [34], a novel technique is proposed for recognizing offline handwritten digit images, utilizing a string edit distance algorithm. The recognition system aims to construct a string representation for each digit and then predicts the digit class by comparing the test string with the existing strings from the training set. The similarity between the test string and the training strings is determined by calculating the edit distance, with lower distances indicating higher similarity. To evaluate the robustness of the proposed system, various digit databases are utilized, including mixed datasets designed specifically for validation purposes. The system is trained on one dataset and tested on another, demonstrating its ability to generalize across different datasets. The robustness of the system is validated using four different types of datasets, showcasing its excellent performance. Additionally, experiments are conducted where the recognition system is trained with a limited number of samples from a specific dataset, but tested using a large number of samples from a different dataset. The results of these experiments are presented in tabular and visual formats, highlighting the performance and effectiveness of the proposed system.

2.11. Amit Choudhary et al [35] 2020:

The main goal of this study [35] is to attain a high level of accuracy using a pure CNN architecture without resorting to ensemble architectures. The use of ensemble architectures typically comes with added computational costs and testing complexity. Therefore, the proposed CNN architecture aims to achieve even better accuracy than ensemble architectures while also reducing operational complexity and costs.

Furthermore, we introduce an effective combination of learning parameters in the design of the CNN, which allows us to establish a new record in accurately classifying MNIST handwritten digits. Through extensive experimentation, we successfully achieved a remarkable recognition accuracy of 99.87% on the MNIST dataset.

3. State of the art Summary:

For illustration purposes, the following figure briefly summarizes all the works mentioned in this chapter.

	study	approach	dataset	Data type	accuracy
1	Gattal et al.	SVM	NSTRING SD19	offline	96.91%
2	R. Dey et al.	matching	MNIST+ARDIS	Offline + online	72.70%
3	Shujing Lyu et al.	CNN	ORAND-CAR-A	offline	92.20%
			ORAND-CAR-B		94.02%
4	P. Sermanet et al.	ConvNets	SVHN	Online + offline	95.10%

Table 1: State of the art Summary (String)

	study	approach	dataset	Data type	accuracy
1	A. Khan et al.	HOG + SVM	MNIST	offline	99.36%
2	Gattal et al.	oBIFs + SVM	CVL	offline	95.21%
3	U. R. Babu et al.	KNN	MNIST	offline	96.94%.
4	R. Dey et al.	matching	MNIST	Offline + online	96.83%
5	K. T. Islam et al.	CNN	MNIST	offline	99.60%
6	Rabia karakaya et al.	SVM	MNIST	offline	90.00%
		Decision tree			87.00%
		Random forest			97.00%
		ANN			97.00%
		KNN			98.00%
		KM			98.00%
7	Shujing Lyu et al.	CNN	CVL	offline	42.69%
8	A.Choudhary et al.	CNN	MNIST	offline	99.87%
9	Y. Lecun et al.	CNN	MNIST	offline	99.05%

Table 2: State of the art Summary (digit)

4. Conclusion:

In this chapter, we dedicated our efforts to conducting an extensive review of the most up-to-date research in the field, up until the year of writing this dissertation (2023). Our goal was to thoroughly examine the datasets that were utilized in previous studies, the methodologies that were employed, and the outcomes that were achieved by each of these studies. To ensure a comprehensive overview, we compiled a summary table that encompasses all the papers that were cited in this chapter. This table serves as a valuable resource, providing a consolidated reference of the key findings and contributions of each study mentioned. By presenting this summary, we aim to provide readers with a holistic understanding of the existing body of work and establish the context for our own research endeavors in subsequent chapters.

Moving forward to the next chapter, we will explain our own approach in detail. We will discuss the training methodology, the datasets we utilized, and the preprocessing and augmentation techniques applied to the data. Additionally, we will present the models we employed to address the research questions. This will offer a clear understanding of our research methodology and pave the way for the subsequent chapters.

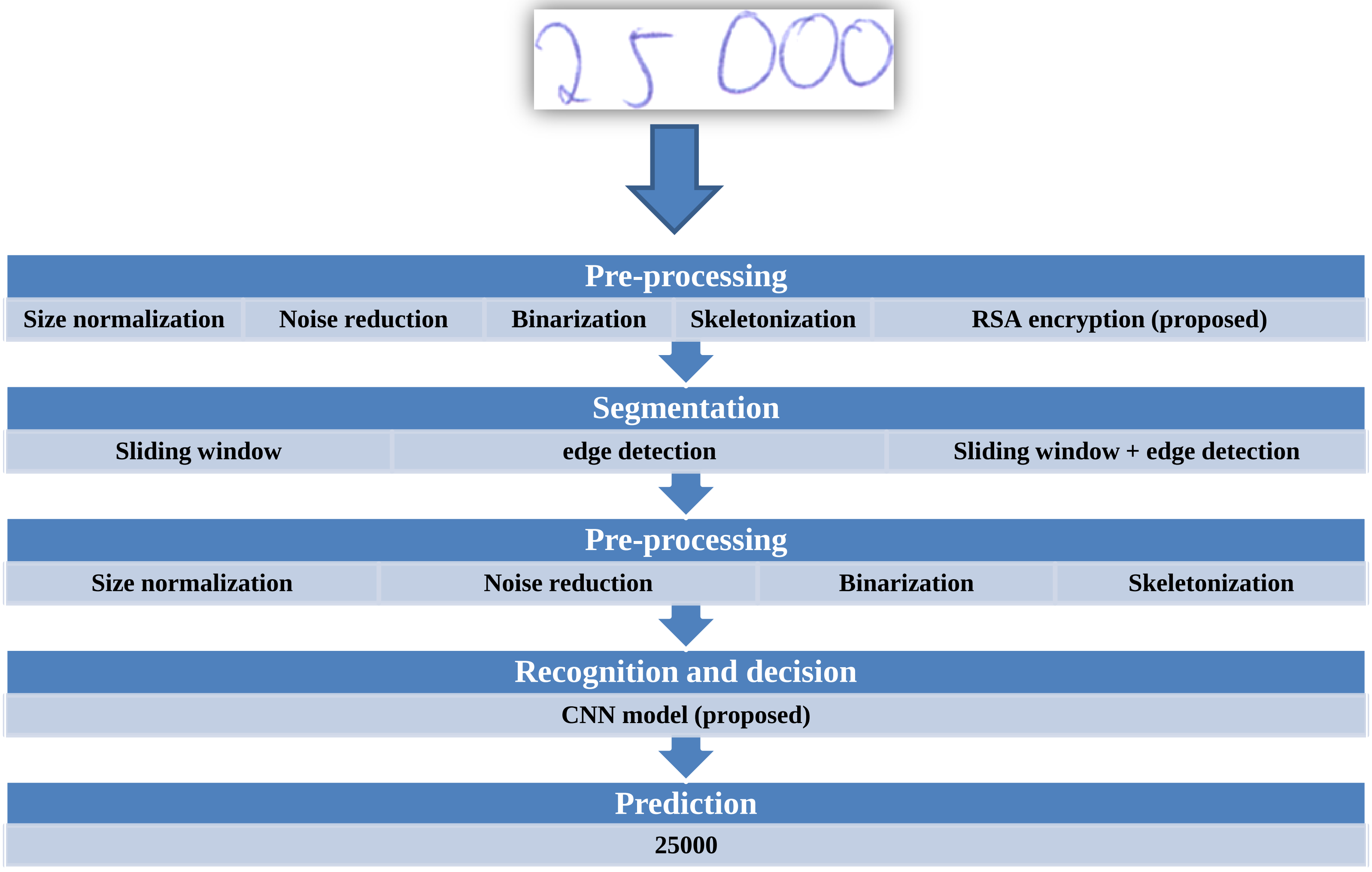
Chapter 3: Our approach

1. Introduction:

In this chapter, we discuss the techniques, dataset, models, and processing methods used in the project. It includes a review of the dataset used, an explanation of the developed models, and an illustration of the data processing methods. The technical details and tools used to achieve the project's specific objectives are documented.

2. Training method:

In order to create a program capable of recognizing a sequence of handwritten numbers with an unknown length, we followed a systematic approach outlined in a diagram. The diagram illustrates the step-by-step process we followed to achieve the desired outcome:



In this chapter, we will delve into the development of a system capable of recognizing a String of digits with an unknown length. This work involved three crucial stages, with some stages occasionally being combined. We will discuss and explain each of these stages in detail, highlighting their significance in achieving the desired objective.

3. Pre-processing:

As discussed in the first chapter, pre-processing plays a vital role in the field of handwritten document recognition as it serves as the initial and fundamental step. The main objective of pre-processing is to enhance the performance of recognition algorithms by addressing specific targets.

Throughout our experimental study, we utilized various pre-processing techniques, including Size normalization, Noise reduction, and Binarization, which were previously defined in the first chapter. These techniques were consistently applied with fixed values to a string of handwritten digits images.

In the pre-processing stage, the size of the image was maintained without alteration to preserve all the features present in the image for subsequent operations. Noise reduction was then applied to improve the clarity of the digits in the image. Following that, the image was converted to a binary system, where black color represented 0 and white color represented 1. This conversion facilitated the Skeletonization process, which was discussed in the first chapter and aimed to compress the image while preserving its fundamental shape, thereby aiding in the subsequent fragmentation stage.

Binarization:

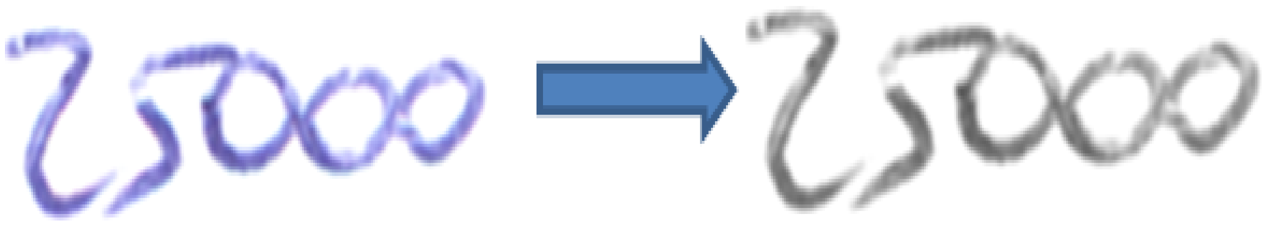


Figure 15: Binarization

Noise reduction:

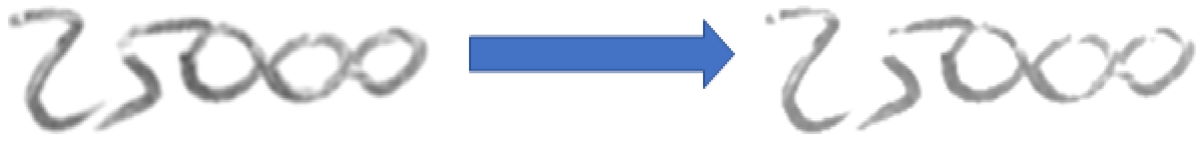


Figure 16: Noise reduction

Skeletonization:

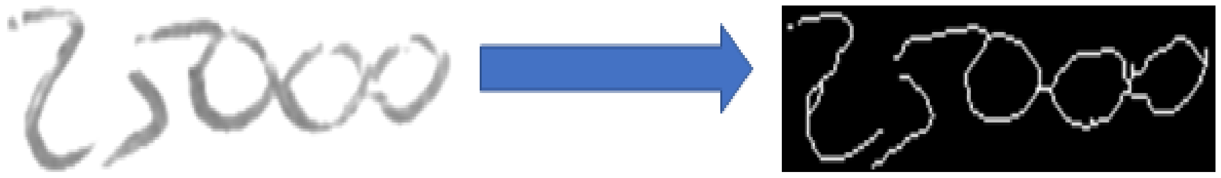


Figure 17: Skeletonization

We suggested a procedure where we utilized the RSA encryption algorithm to encrypt the pixel values in the image. This encryption process altered the format of each pixel, aiming to enhance the accuracy of image segmentation. It is important to note that this method involves applying the encryption algorithm to the original image, rather than the modified version. This approach ensures the preservation of the overall shape and any distinctive features that may be specific to the original image.

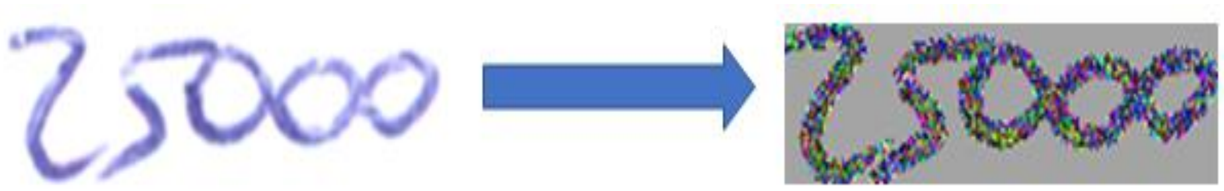


Figure 18: RSA encryption algorithm

In the subsequent pre-processing stage, which followed the segmentation stage, similar settings were employed as in the initial stage. However, in the resizing step, the image containing the string of digits was divided into individual images, each containing a single digit, with a size of

28 * 28. This size was chosen because the CNN model used in the study was trained on a dataset consisting of images with one digit per image, all sized at 28 * 28.

Size normalization:

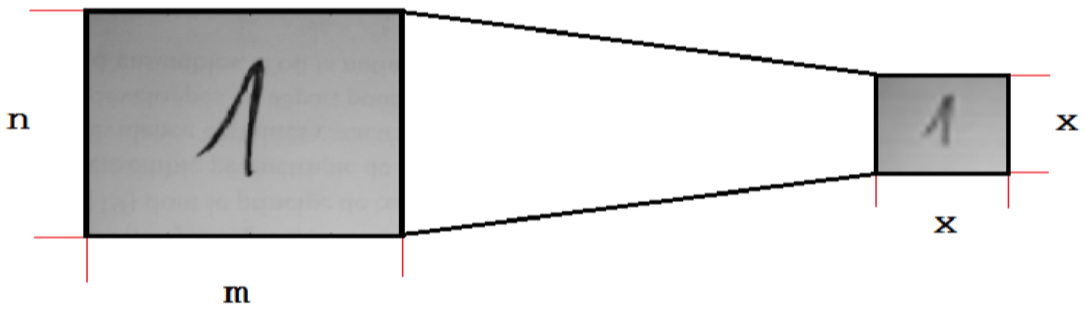


Figure 19: Size normalization

Binarization:

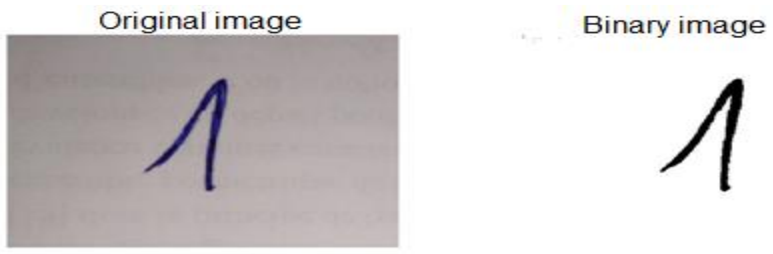


Figure 20: Binarization

Noise reduction:

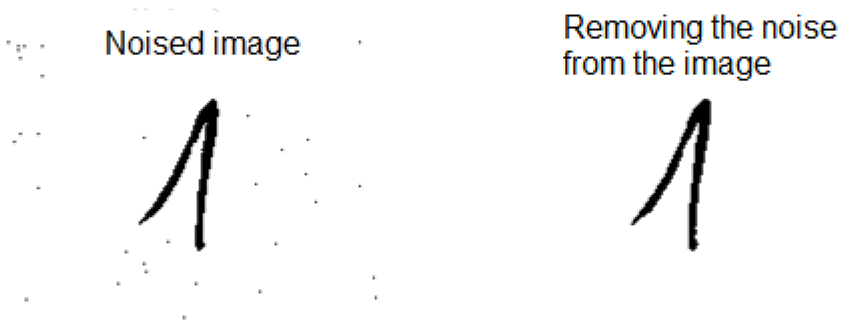


Figure 21: Noise reduction

Skeletonization:

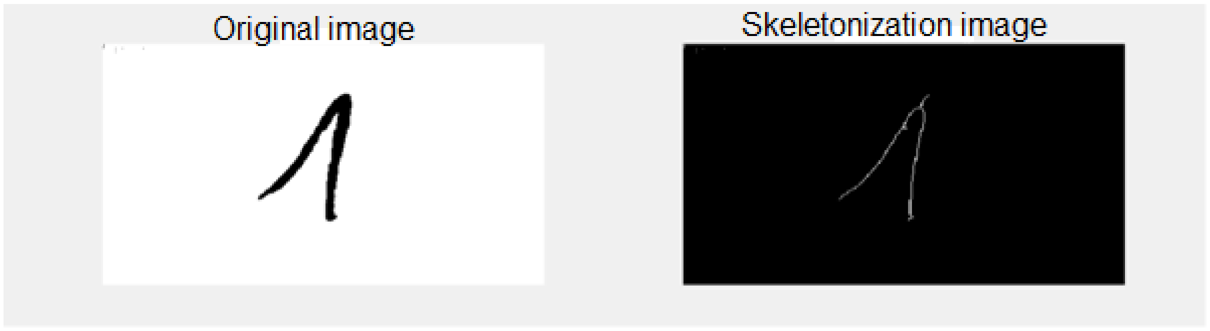


Figure 22: Skeletonization

For the purpose of improving the Handwritten Connected Digits segmentation process, we proposed a method of encoding image pixels, making it an encrypted image, passing it on the segmentation methods that were suggested, and comparing the given results with the previous results.

Benefits:

Data simplification: The Binarization process converts the image into a binary image (black and white), thereby simplifying the data and reducing the complexity of the image. This can facilitate subsequent processing operations and image analysis.

Noise reduction: The Noise reduction process aims to remove noise, extra points, or unnecessary elements from the image. This contributes to improving the image quality and eliminating any noise that may negatively affect subsequent processing operations.

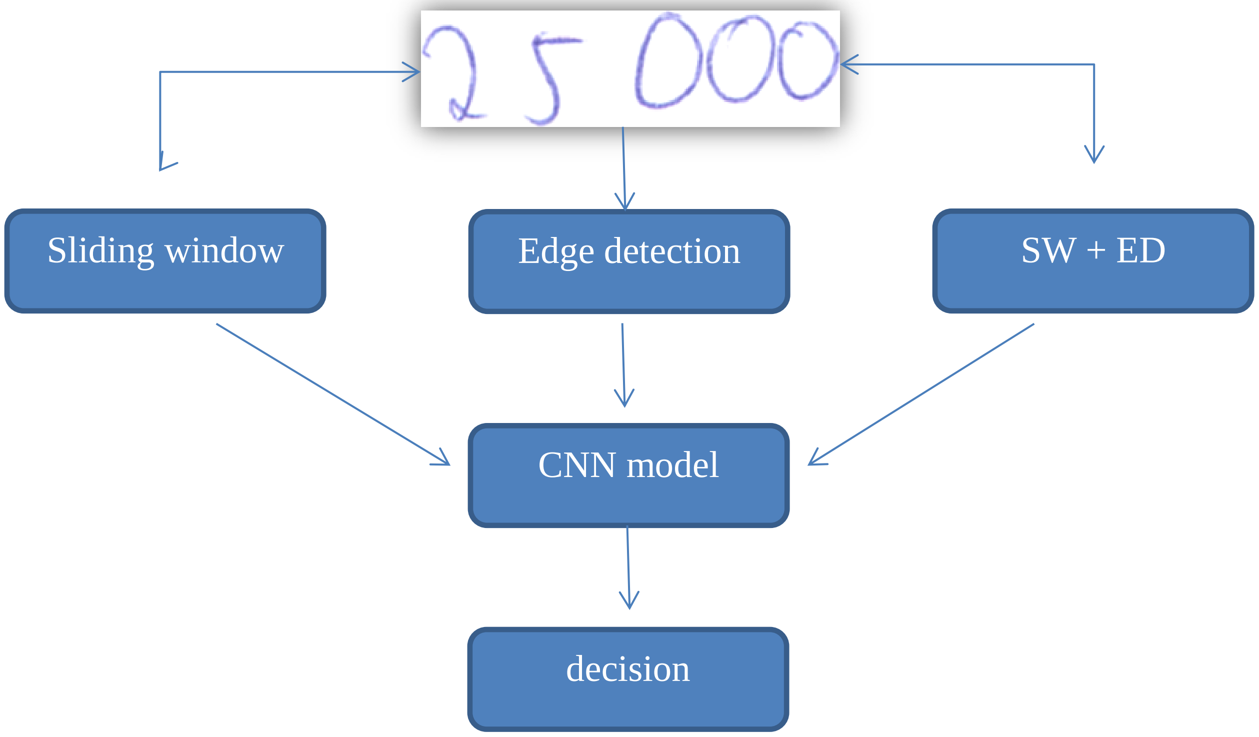
Structure analysis: The Skeletonization process reduces the overall structure of the image into a set of connected thin lines. This can help in identifying and understanding the general structure of the image, which is useful in shape recognition or object identification in the image.

Drawbacks:

Loss of details: The Binarization and Skeletonization processes may lead to the loss of some fine details in the image since it is converted into a binary image. This means that some important details in the image that might be crucial for subsequent processing may be ignored.

Undesirable effects: The Noise reduction process may sometimes have undesirable effects on the image. Important details may be removed along with the noise, which can affect the accuracy of subsequent processing.

4. Segmentation:



4.1.Sliding window:

Sliding window segmentation is a technique used in handwriting digit recognition to extract individual digits from an image. The primary goal is to divide the input image into smaller regions, known as windows, and classify the content within each window as a digit or non-digit [36].

The process begins by applying a sliding window over the input image. The sliding window is a rectangular region that moves across the image in a systematic manner. It starts from a specified position, usually the top-left corner, and moves horizontally with a defined stride (step size).

As the sliding window moves, it extracts the content within the window. In the case of handwriting digit recognition, this content corresponds to a small region of the image that potentially contains a single digit. The size of the sliding window is typically set to match the expected size of the digits in the image.

After extracting the content within each window, a classification algorithm is applied to determine whether the content represents a digit or a non-digit. This classification step aims to differentiate between handwritten digits and other elements in the image, such as background noise or irrelevant markings.

Once the content within each window is classified, a thresholding step is often applied to make a final decision. This thresholding step sets a confidence threshold or probability value, above which the content is considered a digit and below which it is considered a non-digit. The specific threshold value can be determined based on the desired level of accuracy and false-positive and false-negative rates.

The sliding window segmentation process continues until the entire image has been covered, generating a set of smaller regions containing individual digits. These segmented digit regions can then be further processed and recognized using additional techniques.

Overall, sliding window segmentation is a widely used technique in handwriting digit recognition as it allows for the extraction of individual digits from an image by systematically analyzing smaller regions and classifying their content.

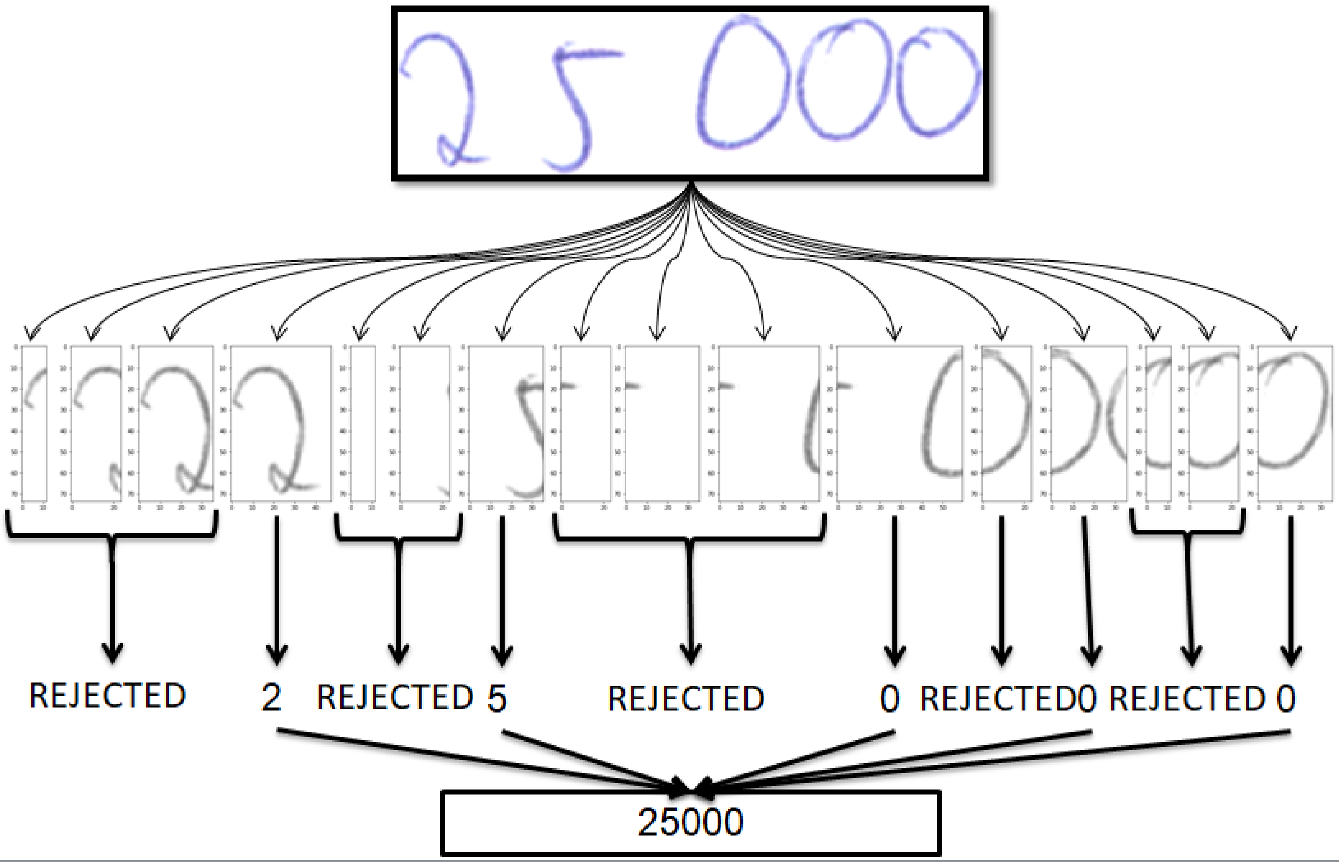


Figure 23: Sliding window

We experimented with various configurations of the sliding window method to determine the most effective settings for our system. We monitored the performance of different settings and eventually found the ideal ones that we used for our study. These settings involve using a window of equal height as each image, which was determined based on the image's height prior to segmentation. For the width, we set it to 12 pixels. These settings proved to be the most optimal and yielded positive results in our research.

4.2.Edge detection:

Edge detection is a fundamental technique employed in handwritten digit recognition for segmentation purposes. Its primary objective is to identify and locate the boundaries or edges of the digits within an input image. By utilizing edge detection, the distinct edges separating the digits from the background or adjacent elements can be identified [37].

The process of edge detection involves analyzing the variations in intensity or color within the image. Various edge detection algorithms, such as the canny edge detector, Sobel operator, or roberts operator, can be employed to accomplish this task.

Once the edges are detected, they form an edge map or binary image, where the edges are represented as white pixels, while the rest of the image is black. This edge map emphasizes the boundaries between different regions within the image, including the boundaries of the handwritten digits.

The segmented edge map is then utilized to extract individual regions or components corresponding to the digits. Connected component analysis or contour-based methods can be employed for this purpose. Connected component analysis groups adjacent edge pixels together to form regions, while contour-based methods trace the contours of the connected edge pixels.

After segmenting the individual regions corresponding to the digits, additional processing techniques can be applied to refine and separate the digit regions. This may involve bounding box detection, where rectangular boundaries are determined around each digit, or region growing, where the segmented regions are expanded based on certain criteria such as intensity or color similarity.

The segmented and refined digit regions can then be passed on to subsequent recognition and classification algorithms. utilize the extracted digit regions to accurately recognize and interpret the digits. The improved accuracy of the handwritten digit recognition system is achieved by effectively extracting the digit boundaries through the application of edge detection, thereby providing more reliable input for the subsequent recognition algorithms.

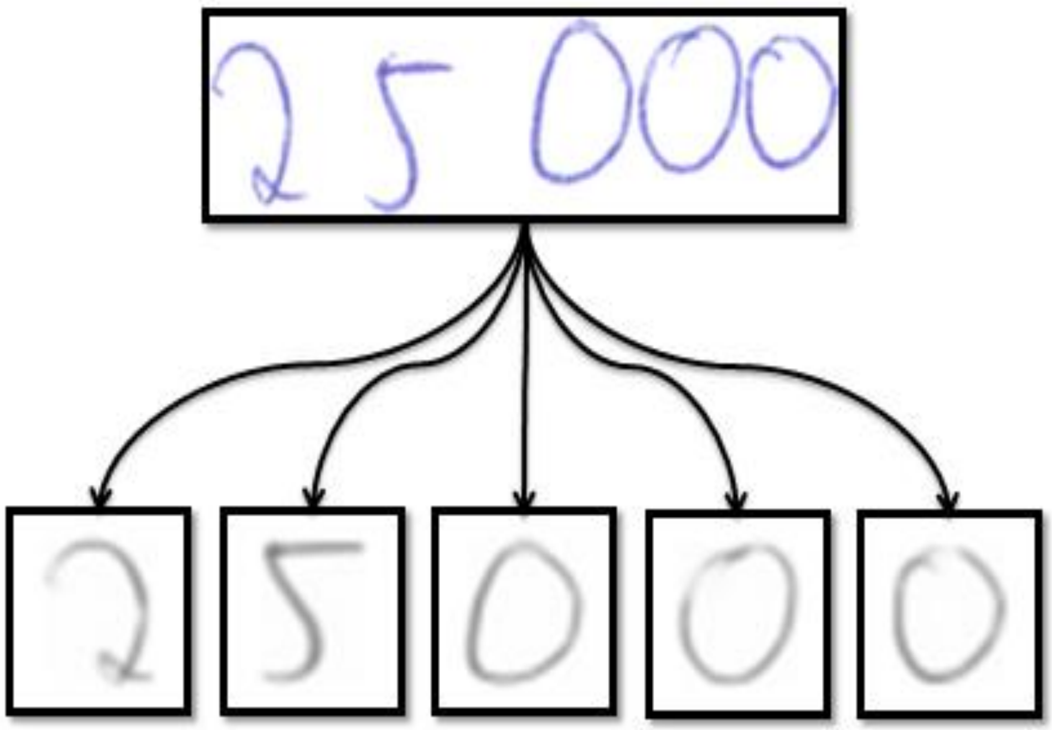


Figure 24: Edge detection

4.3. Edge Detection + Sliding Window

To enhance our image segmentation system, we have integrated various previous techniques into a unified system. This system primarily focuses on detecting edges. In cases where the edges of the numbers appear connected, we employ a sliding window approach, as depicted in the provided image.

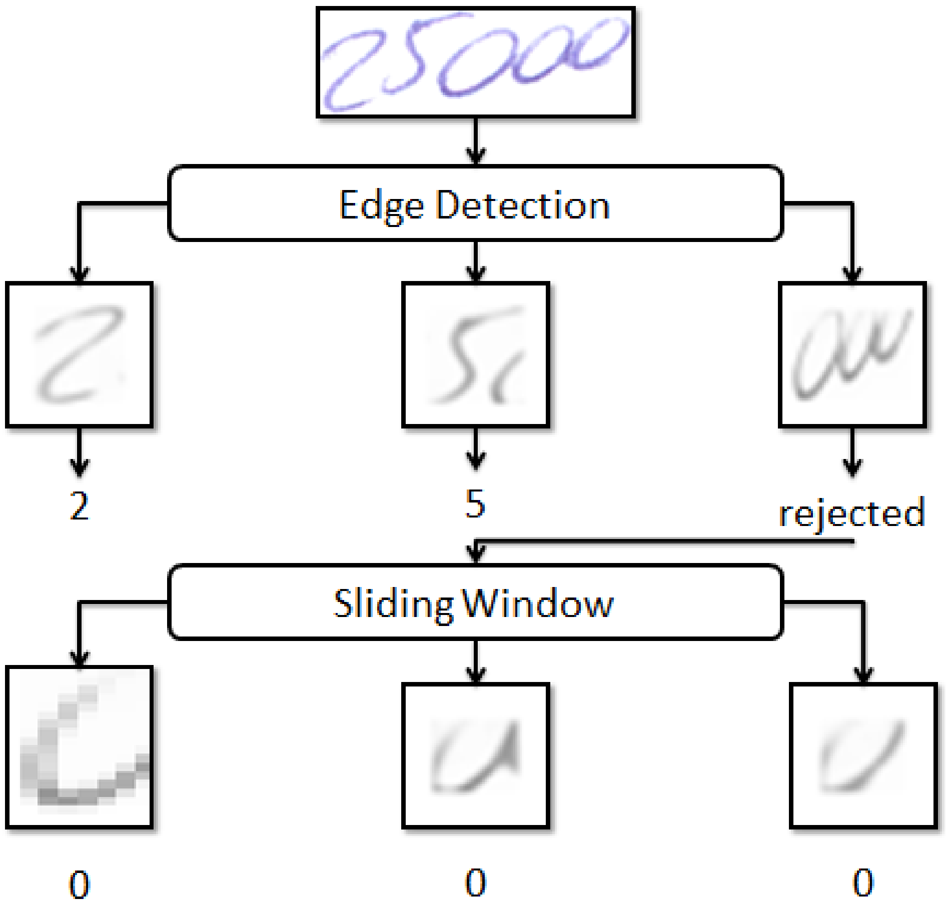


Figure 25: Edge Detection + Sliding Window

5. Proposed CNN architecture:

We have developed a modified CNN model based on AlexNet specifically designed to handle small-sized input images. Our model is tailored for grayscale images with a resolution of 28x28 pixels and a single channel (28,28,1).

The architecture consists of the following key layers:

- **Input Layer:** The initial layer takes input images of size 28x28 with a single channel.
- **Convolutional Layers:** Two consecutive convolutional layers with a kernel size of (5,5) and an activation function of ReLU. Both layers have 32 filters to extract relevant features from the input images.

- **MaxPooling Layer:** A max pooling layer with a pool size of (2,2) is used to down sample the feature maps, reducing their spatial dimensions by half.
- **Dropout Layer:** To prevent overfitting, we have added a dropout layer with a rate of 0.25 after the max pooling layer. This layer randomly drops out 25% of the neurons during training.
- **Additional Convolutional Layers:** Following the dropout layer, we have duplicated the previous convolutional layers. However, in this duplication, we have adjusted the kernel size to (3, 3) and increased the number of filters to 64.

Overall, our modified CNN model effectively handles the small input image size by utilizing a combination of convolutional layers, max pooling, and dropout regularization. By adapting the architecture from AlexNet, we ensure that our model can effectively extract relevant features from the handwritten digit images, leading to accurate recognition and classification results.

Below is the summary table for the given model:

Layer (Type)	Output Shape	Param #
Img_Input (Inputlayer)	(None, 28, 28, 1)	0
Layer_1 (Conv2D)	(None, 28, 28, 32)	832
Layer_3 (Conv2D)	(None, 28, 28, 32)	25632
Layer_2 (Maxpooling2d)	(None, 14, 14, 32)	0
Dropout_1 (Dropout)	(None, 14, 14, 32)	0
Layer_4 (Conv2D)	(None, 14, 14, 64)	18496
Layer_5 (Conv2D)	(None, 14, 14, 64)	36928
Layer_6 (Maxpooling2d)	(None, 7, 7, 64)	0
Dropout_2 (Dropout)	(None, 7, 7, 64)	0
Flatten (Flatten)	(None, 3136)	0
Layer_7 (Dense)	(None, 256)	803072
Dropout_3 (Dropout)	(None, 256)	0
dense (Dense)	(None, 10)	2570
Total params: 887,530	Trainable params: 887,530	Non-trainable params: 0

Table 3: CNN architecture used

6. Dataset used:

6.1.Cvl Dataset

The CVL Digit Strings dataset is a collection of digit strings extracted from the CVL Single Digit dataset. It includes 10 different digit strings written by approximately 120 different writers. This dataset contains a total of 1,262 training images.

The CVL Single Digit dataset, on the other hand, consists of 7,000 single digits, with 700 digits per class. These single digits were extracted from the digit strings in the CVL Digit Strings dataset. The CVL Single Digit dataset provides a focused collection of individual digits for training and recognition purposes.

It is important to note that the validation set, which has the same size as the training set, consists of different writers. The validation set is primarily used for parameter estimation and validation, rather than supervised training. This ensures that the models are evaluated on unseen data, enhancing the reliability of the validation results.

In summary, the CVL Digit Strings dataset contains digit strings written by various writers, while the CVL Single Digit dataset is derived from these strings and consists of individual digits. Both datasets play a crucial role in training and evaluating models for handwritten digit recognition [14].

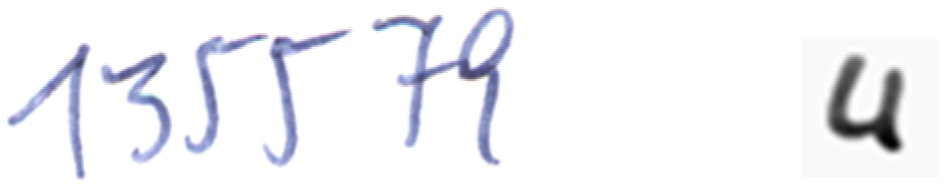


Figure 26: cvl dataset [14]

6.2.Mnist Dataset:

MNIST stands for "Modified National Institute of Standards and Technology." It is a well-known dataset in the field of machine learning and computer vision. The dataset consists of a collection of grayscale digital images representing hand-written numbers from 0 to 9 [12].

MNIST is typically composed of 60,000 images used for training and 10,000 images used for validation and testing. The images are sized at 28x28 pixels and are converted into a digital dataset with values ranging from 0 to 255, where white color represents a value of 0, and black color represents a value of 255.

The MNIST dataset is widely used for developing and testing machine learning models in the field of optical image classification. The main objective is to correctly identify and classify the handwritten numbers in the images.

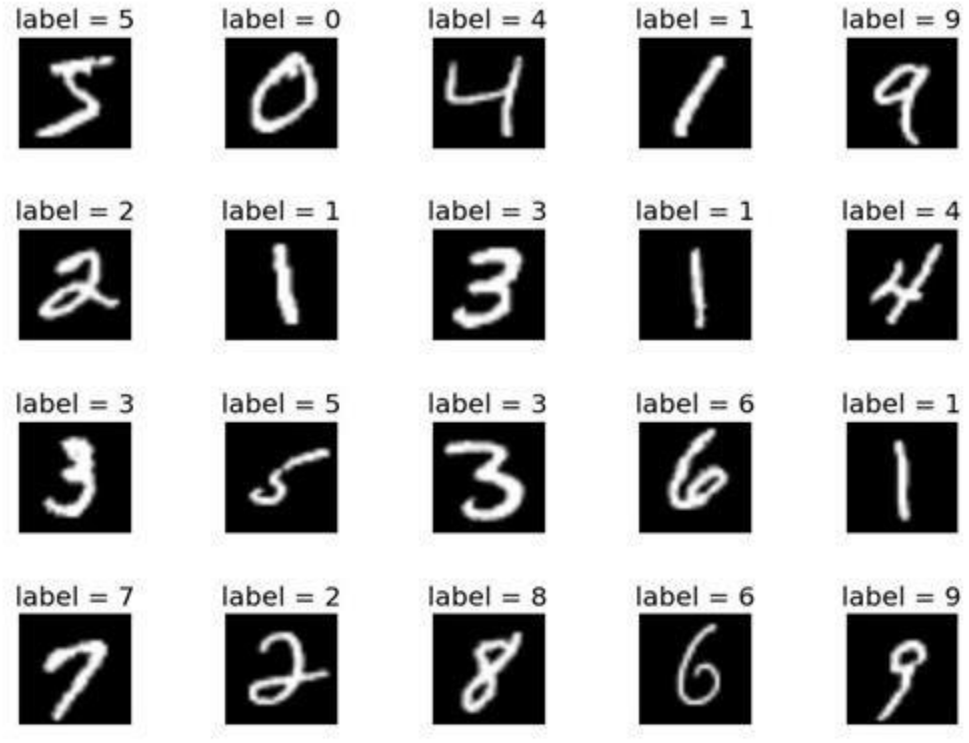


Figure 27: The MNIST dataset [13]

7. Conclusion:

This chapter focuses on the recognition of handwritten numbers. It highlights the significance of preprocessing techniques, such as normalization and noise removal, in improving accuracy. Different segmentation techniques like sliding window and edge detection are discussed, along with their pros and cons in number recognition. An adapted CNN architecture based on AlexNet is proposed for recognizing handwritten digits, with detailed explanations of the modifications made to enhance its performance. The training and testing were conducted using standard numeric datasets. The next chapter will present the experimental results, demonstrating how well the trained CNN model accurately recognizes handwritten digits. These results will be carefully analyzed to evaluate the effectiveness of the proposed techniques and their suitability for the segmentation process.

Chapter 4: The experimental results

1. Introduction

In the preceding chapter, we outlined the overall architecture of the system and provided a comprehensive breakdown of each stage and step involved in the process. In this chapter, we focus on discussing the tools utilized in the development of our work, alongside showcasing the empirical results that have been achieved.

2. Development Tools

2.1.Hardware Tools

This program was created in a type of HP laptop with the following specifications:

• Processor: AMD Ryzen 3 2200U/2.50 GHz

• Installed memory (RAM): 12.0GB.

• System type: 64-bit operating system.

• Operating system: Windows 10 pro.

2.2.Software Tools

Anaconda Distribution

The Anaconda Distribution, an open-source platform, provides the most straightforward approach for conducting Python-based data science and machine learning tasks on Linux, Windows, and Mac OS X. Trusted by more than 11 million users globally, it has become the industry norm for development, testing, and training processes on a single machine. [38].

Spyder IDE

Spyder is a Python-based scientific environment that is freely available and open source. It has been specifically developed by and for scientists, engineers, and data analysts. This software offers a distinctive blend of powerful features, including advanced editing, analysis, debugging, and profiling tools typically found in comprehensive development environments. Additionally, it provides data exploration, interactive execution, deep inspection, and visually appealing visualization capabilities commonly associated with scientific packages [39].

Google colab

Colaboratory, or "Colab" for convenience, is a product developed by Google Research. It offers users the ability to write and execute Python code directly from their web browser. It serves as an excellent platform for individuals interested in machine learning, data analysis, and educational purposes. Colab functions as a hosted Jupyter notebook service, eliminating the need for any setup or configuration. Furthermore, it provides users with free access to computational resources, including GPUs, enhancing the overall computing experience [40].

Python language

The utilized version is 3.9. Python is a programming language known for its simplicity and power, making it easy to learn and use. It offers efficient high-level data structures and employs a straightforward yet effective approach to object-oriented programming. Python's elegant syntax, dynamic typing, and interpreted nature make it a versatile language suitable for scripting and rapid application development across various domains and platforms [41].

Data split:

In this study, we divided the data into two sections for different purposes. The first section was used for training a CNN model, while the second section was utilized to apply the segmentation methods proposed in this research.

For the training phase of the CNN model, we used the cvl and mnist data sets together where we had 10 classes, each class consisting of 5700 samples that were used for training, and 4178 samples for testing, the set's size 24*24. In addition, we introduced a new class, Number 11, which contains parts of number models that represent undesirable patterns.

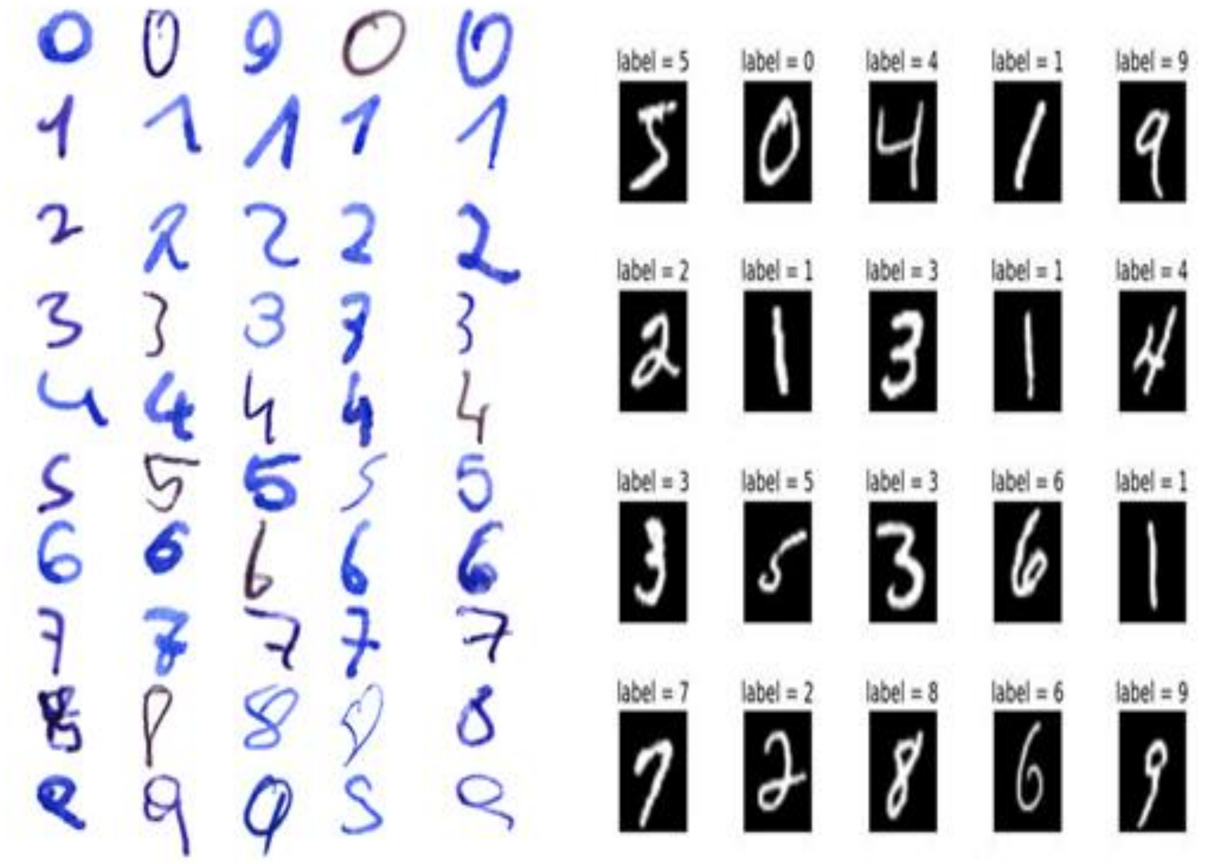


Figure 28: Single digit dataset used

The second section focused on the application of segmentation methods. It comprised the CVL string digits dataset, which consisted of a series of numbers in each image. This section was solely dedicated to the application of segmentation methods and their evaluation.

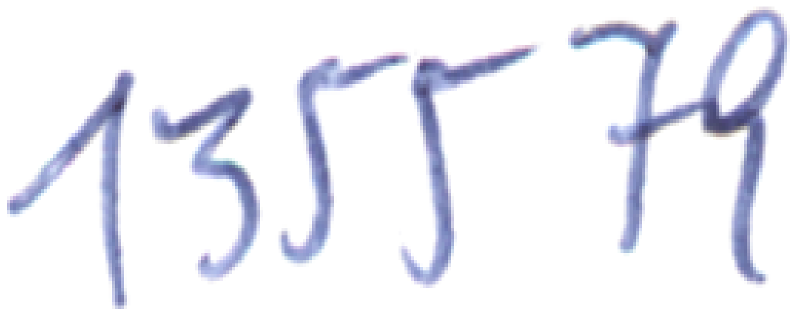


Figure 29: Cvl string digits dataset used [14]

3. Experimentations and Results:

In this section, we will systematically examine and discuss the various findings derived from our methodology, as presented in the previous chapter. Each finding will be presented in a step-by-step manner, accompanied by thorough explanations and justifications.

3.1. Proposed CNN:

As stated earlier, we initially trained the CNN model by combining the CVL and MNIST datasets. The outcomes of this training process yielded favorable results, which are illustrated in the table below:

	The proposed CNN model			
	Accuracy	Loss	Val accuracy	Val loss
CVL+MNIST Datasets	98.42%	05.38%	97.81%	07.74%

Table 4: The proposed CNN model results

According to the table, the model trained on this combined dataset achieved remarkably high performance. It exhibited an accuracy of 98.42% and a low loss of 05.38%. These results were crucial as we relied on this well-performing model for predicting the numbers obtained through the segmentation process.

3.2. Segmentation:

In this section, we will introduce the segmentation methods proposed in this study and demonstrate their efficacy in the segmentation process by accurately predicting segmented numbers. We will present the final results, including the number of instances where the prediction was entirely correct, the number of instances where there was an error in a single digit, and the number of instances where errors occurred in two digits (resulting in a different number being predicted).

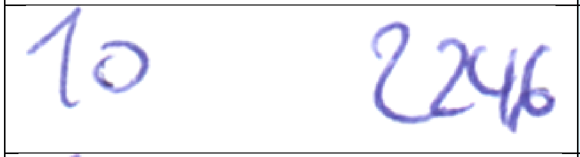
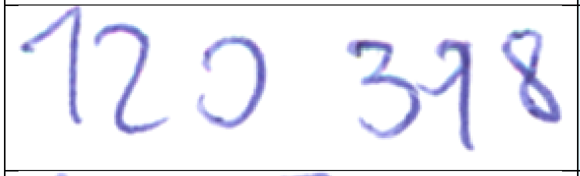
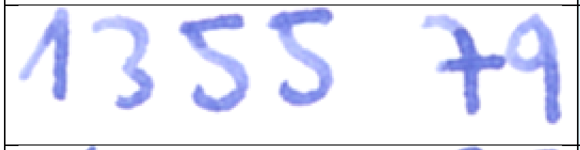
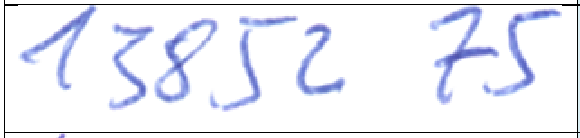
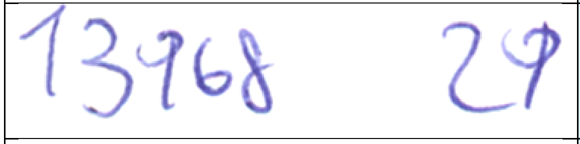
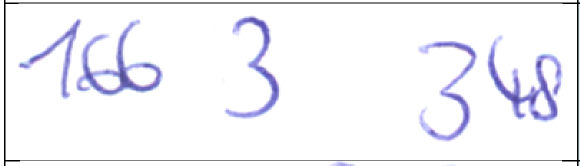
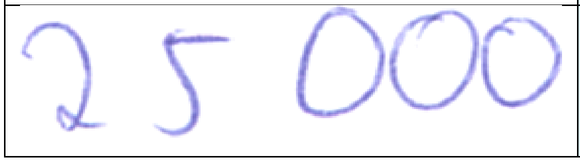
3.2.1. Proposed system using Sliding window:

The segmentation process, utilizing the sliding window technique as explained in Chapter 3, was conducted in three stages.

In the first stage, we applied the sliding window to the original image, employing the pre-processing methods discussed in the third chapter, except for skeletonization.

In the second stage, the sliding window was applied to an image that had undergone pre-processing, including the skeletonization process. Here, the image was further processed with skeletonization, which aids in capturing the essential structure of the digits.

Lastly, in the third stage, we applied the sliding window to the image that had been subjected to an encryption method. The outcomes of this segmentation process are presented in the table, showcasing the results obtained from each stage.

	the first stage				**the second stage**				**the third stage**			
String digits	**CP**	**OM**	**TM**	**T**	**CP**	**OM**	**TM**	**T**	**CP**	**OM**	**TM**	**T**
	0	0	19	24	0	0	1	24	0	0	0	24
	1	2	9	24	0	0	4	24	0	0	0	24
	0	0	0	24	0	0	1	24	0	0	0	24
	0	1	6	24	0	0	2	24	0	0	0	24
	0	1	0	24	0	0	0	24	0	0	0	24
	0	1	2	24	0	0	1	24	0	0	0	24
	1	5	7	24	0	3	6	24	0	0	0	24

2806 34	0	0	2	24	0	0	0	24	0	0	0	24
359 910	0	1	4	24	0	0	0	24	0	0	0	24
3890 30	1	1	5	24	0	0	0	24	0	0	0	24
412 258	0	3	10	24	0	0	4	24	0	0	0	24
41 79248	0	1	3	24	0	0	0	24	0	0	0	24
4300 40	0	2	8	24	0	0	1	24	0	0	0	24
4479 55	20	3	0	24	1	1	1	24	0	0	0	24
529379	0	3	8	24	0	0	2	24	0	0	0	24
584 1077	2	2	6	24	0	0	4	24	0	0	0	24
609 251	0	0	2	24	0	1	0	24	0	0	0	24
627379	3	5	14	24	0	2	5	24	0	0	0	24

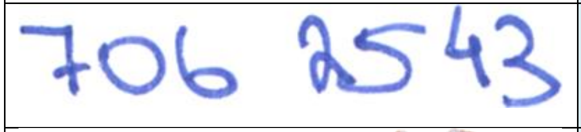
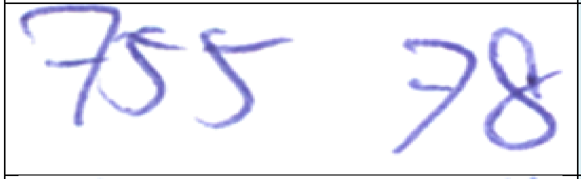
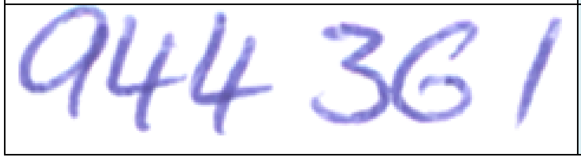
	0	0	8	24	0	1	2	24	0	0	0	24
	0	1	5	24	0	0	1	24	0	0	0	24
	0	1	7	24	0	0	0	24	0	0	0	24
	3	8	12	24	0	1	17	24	0	0	0	24
	0	0	0	24	0	0	0	24	0	0	0	24
	0	0	4	24	0	0	0	24	0	0	0	24

Table 5: Sliding window result

CP: correct prediction.

OM: one mistake.

TM: two mistakes.

Note: The outcome achieved represents the most favorable outcome among a series of conducted experiments. In each trial, we altered the width of the sliding window while maintaining the length equal to that of the original image. The window size that yielded the optimal results was determined to be 8.

The table displays the outcomes of our initial experiment. As previously mentioned, the experiment consisted of three stages. Due to the time and resource requirements, we applied the experiment to 24 images for each number.

During the first stage, we implemented the segmentation process on the original image, resulting in satisfactory to commendable outcomes. Our evaluation of the segmentation process effectiveness was based on three values. It is worth noting that the second and third values, which correspond to cases where one or two elements were predicted incorrectly, yielded average results.

Moving on to the second stage, we applied the segmentation process to an image that underwent skeletalization. Unfortunately, the results of this experiment were poor, with no accurate predictions. However, the cases where there was an error in one or two components were minimal.

In the third stage, which we proposed, we utilized segmentation methods on an image and applied the RSA encryption algorithm. Regrettably, the results were disappointing, with no correct predictions observed in any of the three cases.

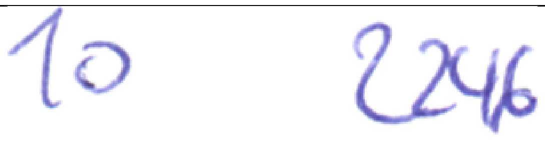
3.2.2. Proposed system using Edge detection:

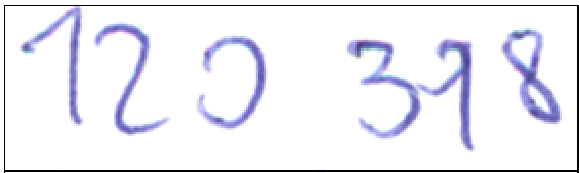
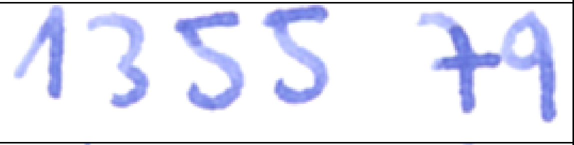
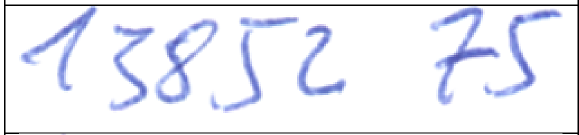
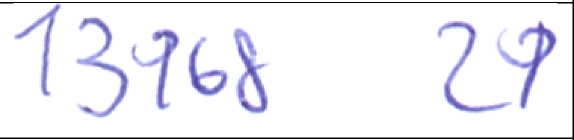
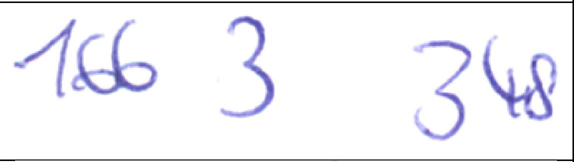
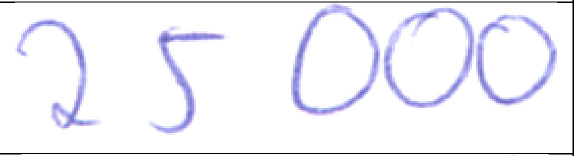
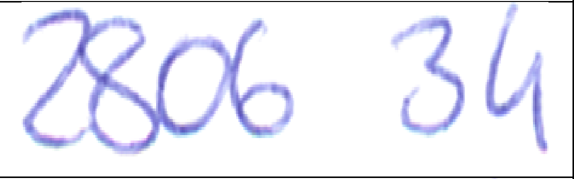
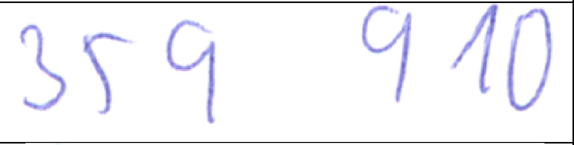
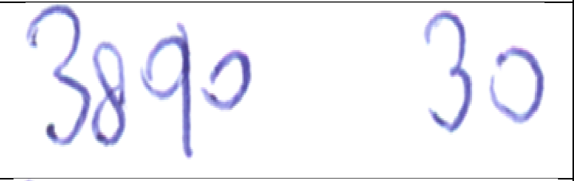
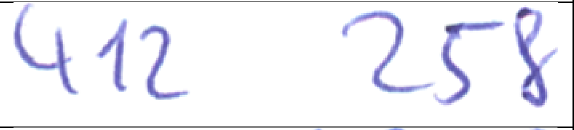
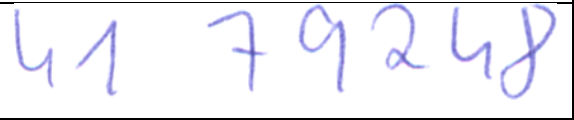
The segmentation process, utilizing the Edge detection technique as explained in Chapter 3, was conducted in three stages.

In the first stage, we applied the Edge detection to the original image, employing the pre-processing methods discussed in the third chapter, except for skeletonization.

In the second stage, the Edge detection was applied to an image that had undergone pre-processing, including the skeletonization process. Here, the image was further processed with skeletonization, which aids in capturing the essential structure of the digits.

Lastly, in the third stage, we applied the Edge detection to the image that had been subjected to an encryption method. The outcomes of this segmentation process are presented in the table, showcasing the results obtained from each stage.

	the first stage				the second stage				the third stage			
String digits	**CP**	**OM**	**TM**	**T**	**CP**	**OM**	**TM**	**T**	**CP**	**OM**	**TM**	**T**
	19	24	28	100	0	0	0	100	0	0	0	100

	16	27	18	100	0	0	0	100	0	0	0	100
	0	0	4	100	0	0	0	100	0	0	0	100
	11	26	21	100	0	0	0	100	0	0	0	100
	13	15	20	100	0	0	0	100	0	0	0	100
	11	13	21	100	0	0	0	100	0	0	0	100
	18	16	19	100	0	0	0	100	0	3	14	100
	11	22	25	100	0	0	0	100	0	0	0	100
	17	17	27	100	0	0	0	100	0	0	0	100
	18	15	17	100	0	0	0	100	0	0	0	100
	17	20	16	100	0	0	0	100	0	0	0	100
	9	27	23	100	0	0	0	100	0	0	0	100

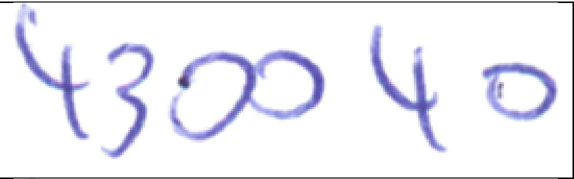
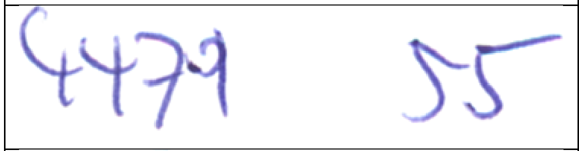
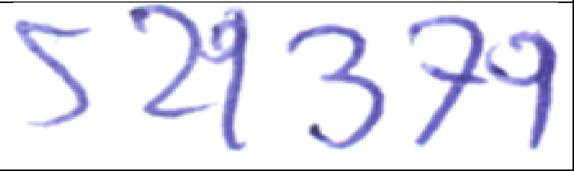
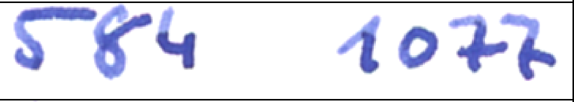
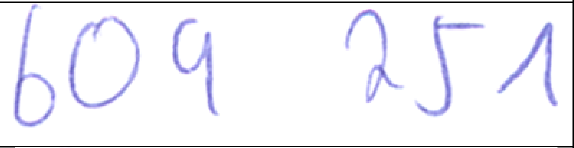
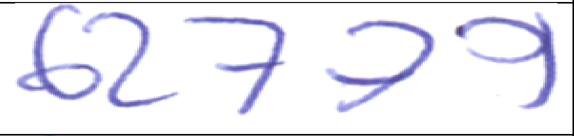
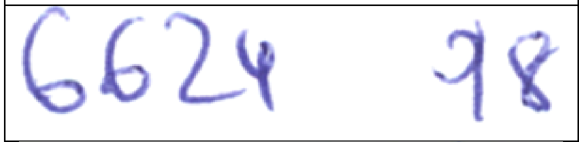
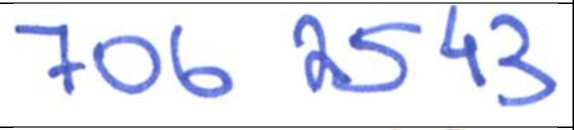
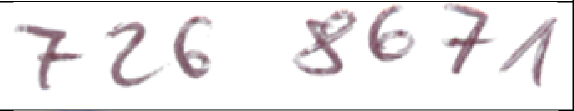
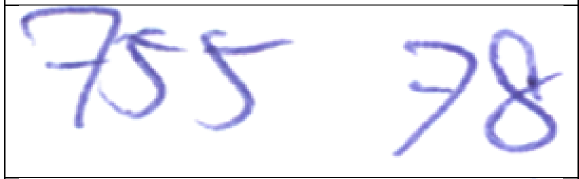
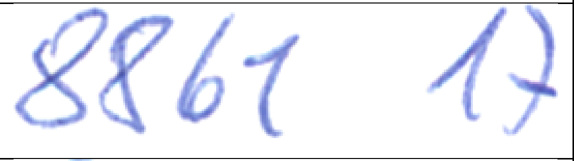
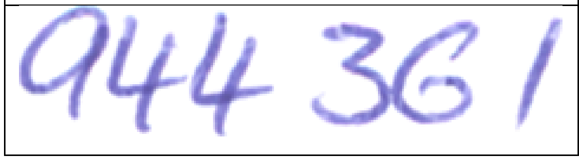
 4300 40	25	18	18	100	0	0	0	100	0	0	0	100
 4479 55	12	26	21	100	0	0	0	100	0	0	0	100
 529379	17	14	30	100	0	0	0	100	0	0	0	100
 584 1077	14	18	26	100	0	0	0	100	0	0	0	100
 609 251	11	20	19	100	0	0	0	100	0	0	0	100
 62779	14	32	17	100	0	0	0	100	0	0	0	100
 6624 98	8	14	23	100	0	0	0	100	0	0	0	100
 706 2543	28	36	37	179	0	0	0	179	0	0	0	179
 726 8671	11	16	14	100	0	0	0	100	0	0	0	100
 755 78	24	19	19	100	0	0	0	100	0	0	0	100
 8861 17	0	0	1	100	0	0	0	100	0	0	0	100
 944 361	17	30	12	100	0	0	0	100	0	0	0	100

Table 6: Edge detection result

CP: correct prediction.

OM: one mistake.

TM: two mistakes.

The table presents the findings of our second experiment. As previously mentioned, the experiment consisted of three stages. Due to the time and resource-intensive nature of the method, we applied the experiment to 100 images for each number.

During the first stage, we implemented the segmentation process on the original image. The results obtained ranged from good to excellent. Our evaluation of the segmentation process relied on three values. Firstly, we measured the accuracy of correctly predicting the image, which yielded predominantly positive and excellent outcomes. Additionally, we considered two values that represented cases where the prediction incorrectly identified one or two elements within each number. In these cases as well, the results were excellent.

Contrary to our expectations, the second and third stages, involving the skeletonization process and the RSA encryption algorithm, respectively, did not yield improved results. Instead, the outcomes were entirely negative. None of the three values showed completely accurate predictions.

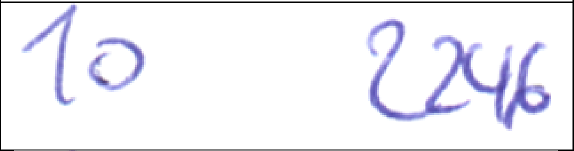
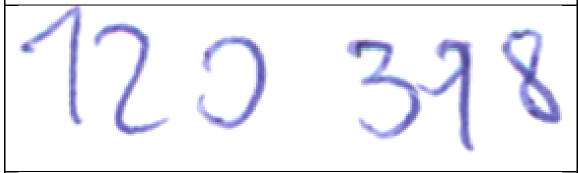
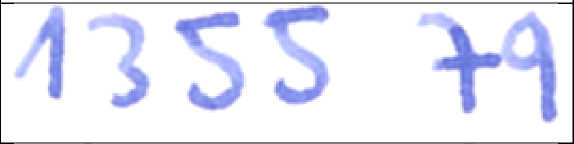
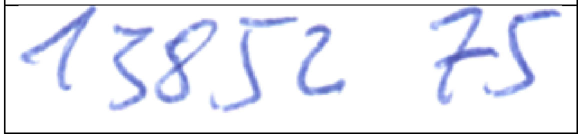
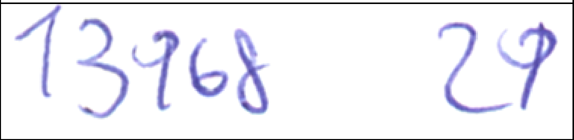
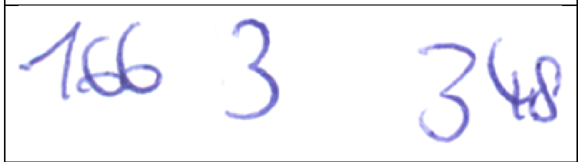
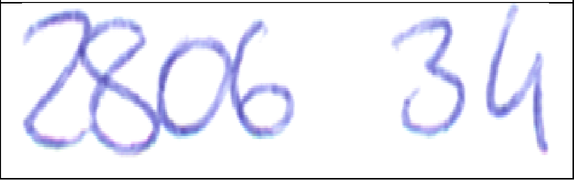
3.2.3. Proposed system using Sliding window and Edge detection:

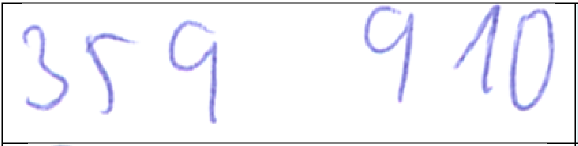
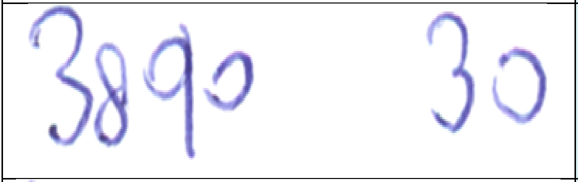
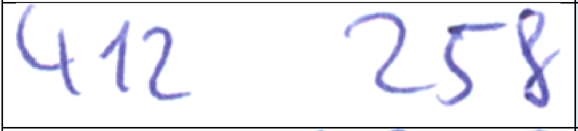
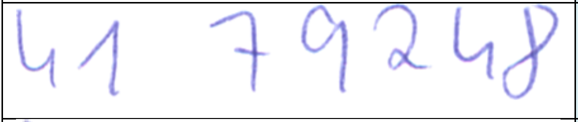
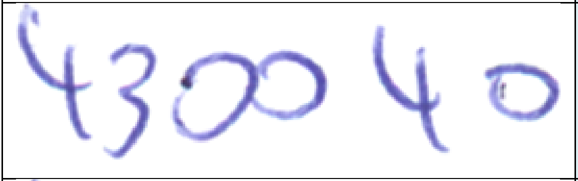
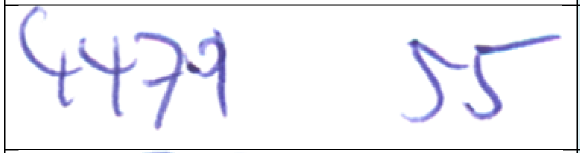
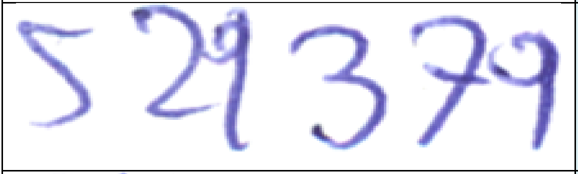
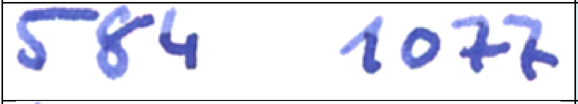
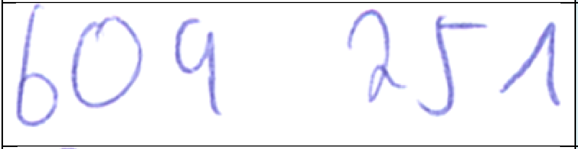
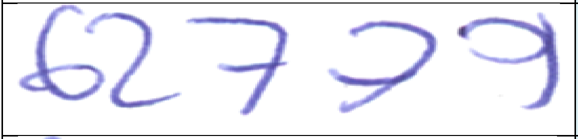
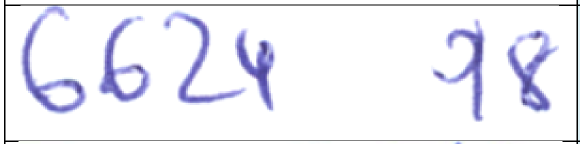
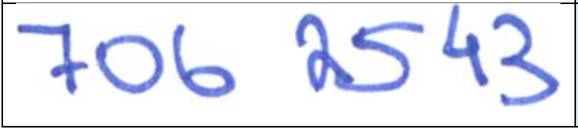
The segmentation process involved the utilization of the edge detection technique along with the proposed sliding window method. This process was conducted in two stages, as the structureization stage was found to be ineffective in a previous study when combined with edge detection.

In the first stage, we applied the edge detection technique and the sliding window to the original image. Pre-processing methods discussed in the third chapter, excluding skeletonization, were employed during this stage.

In the second stage, the edge detection technique and the sliding window were applied to the image that had undergone an encryption method. This stage involved processing the image after applying an encryption technique.

The results of the segmentation process for each stage are presented in the table, illustrating the obtained outcomes.

	the first stage				the second stage			
String digits	**CP**	**OM**	**TM**	**T**	**CP**	**OM**	**TM**	**T**
	19	24	28	100	0	0	0	100
	16	27	19	100	0	0	0	100
	0	0	4	100	0	0	0	100
	11	27	20	100	0	0	0	100
	13	15	20	100	0	0	0	100
	11	13	21	100	0	0	0	100
	18	17	18	100	0	0	0	100
	11	22	26	100	0	0	0	100

	17	17	27	100	0	0	0	100
	18	15	17	100	0	0	0	100
	17	20	18	100	0	0	0	100
	9	27	23	100	0	0	0	100
	25	18	18	100	0	0	0	100
	12	26	24	100	0	0	0	100
	17	14	30	100	0	0	0	100
	14	18	26	100	0	0	0	100
	11	20	19	100	0	0	0	100
	14	32	17	100	0	0	0	100
	8	14	23	100	0	0	0	100
	28	37	37	179	0	0	0	179

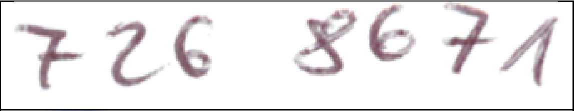
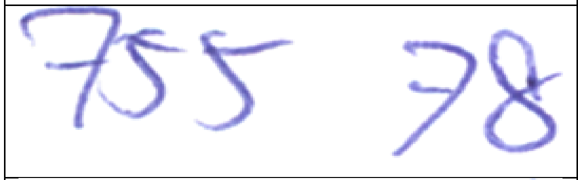
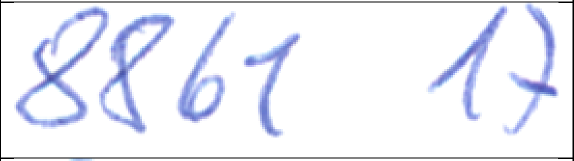
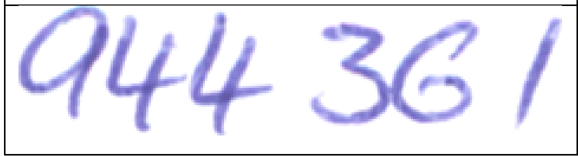
	11	16	14	100	0	0	0	100
	24	20	18	100	0	0	0	100
	0	0	1	100	0	0	0	100
	17	30	12	100	0	0	0	100

Table 7: Sliding window and Edge detection result

CP: correct prediction.

OM: one mistake.

TM: two mistakes.

Note: The outcome achieved represents the most favorable outcome among a series of conducted experiments. In each trial, we altered the width of the sliding window while maintaining the length equal to that of the original image. The window size that yielded the optimal results was determined to be 9.

The table displays the outcomes of our third experiment in this study. As previously stated, the experiment was conducted in two phases. We conducted the experiment on 100 images for each number due to the time and resources required for this method.

During the first phase, when we applied the segmentation process to the original image, we observed an enhancement in the quality of segmentation, and the number of accurately predicted cases increased compared to the previous two experiments.

However, the results of the second phase were extremely disappointing, as none of the cases were predicted correctly.

4. Comparison study:

In this section, we will present a comprehensive table that combines the results obtained from each stage and case. This table will display the total sums of the results previously discussed in detail in the preceding section.

			Total Correct Predictions	Total One Mistake Predictions	Total Two Mistakes Predictions	Total Connected Digits	recognition rate
Sliding Window	Case 1	**Original Picture**	31	41	141	576	36.97%
	Case 2	**Skeletonization**	1	9	52	576	10.76%
	Case 3	**Encryption Algorithm**	0	0	0	576	0%
Edge detection	Case 1	**Original Picture**	341	465	476	2479	51.71%
	Case 2	**Skeletonization**	0	0	0	2479	0%
	Case 3	**Encryption Algorithm**	0	3	14	2479	00.68%
Edge Detection + Sliding Window	Case 1	**Original Picture**	341	469	480	2479	52.03%
	Case 2	**Encryption Algorithm**	0	0	0	2479	0%

Table 8: Comparison of technologies

In the provided table, we present the overall outcomes for each stage and scenario, where some outcomes were satisfactory while others were disappointing. Based on the table, it is evident that utilizing Skeletonization and Encryption Algorithm did not contribute significantly to our work. In fact, their results were not helpful but rather underwhelming. Conversely, employing the original image yielded positive results as it preserved the essential characteristics of the numbers and played a vital role in identifying the numbers within an image.

Moreover, the adoption of the sliding window technique produced weak results. We concluded that the effectiveness of the sliding window heavily relies on the capabilities of the utilized deep learning model and its classification ability to accurately recognize and differentiate between numbers and unwanted elements.

On the other hand, the implementation of Edge detection technology exhibited positive outcomes. However, it faced a limitation when dealing with connected numbers, where handwritten numbers are merged together. Some connected numbers were correctly recognized, while others were identified as individual numbers, disregarding their original composition of interconnected numbers. To address this limitation, we proposed the final experiment.

The combination of the two techniques proved to be a crucial element in this study as it improved the segmentation process. Nonetheless, as mentioned earlier, the sliding window technique necessitates a robust deep learning model and an effective classifier to achieve more precise and accurate outcomes.

Conclusion:

In conclusion, the results achieved from the model for handwritten digit recognition and the explored segmentation methods have been presented in this chapter. A comprehensive analysis of the obtained results and the techniques implemented to improve the segmentation process has been provided. The successful outcomes in certain scenarios have demonstrated the effectiveness of the segmentation techniques utilized, fulfilling the main objective of this thesis. Furthermore, the comparison of different scenarios and approaches has allowed for a deeper understanding of their respective strengths and weaknesses. Overall, this chapter has contributed valuable insights into the performance and limitations of segmentation methods for handwritten digit recognition.

General Conclusion

Handwritten Connected Digits Recognition is a significant and extensive field with notable contributions. In this thesis, our focus was on developing a Segmentation Based Method for Handwritten Connected Digits using Image Encryption Technique. We delved into the domain of handwritten digit recognition and deep learning, discussing well-known CNN models and the datasets commonly employed in this field. We also explored various preprocessing methods and segmentation techniques.

To provide a comprehensive overview of the field until the time of writing this thesis (2023), we conducted an extensive review of recent studies. We examined the datasets, methodologies, and results achieved by each study, summarizing all the relevant papers in a table.

Our research encompassed different aspects of handwritten number recognition. We investigated training methods, preprocessing techniques, and various segmentation approaches such as sliding window and edge detection. We also proposed a modified CNN architecture based on AlexNet specifically designed for this task and utilized relevant datasets.

We divided the data into two parts for different purposes. The first section was used for training the CNN model, consisting of 10 classes with 5700 samples each for training and 4178 samples for testing. Additionally, we introduced a new class, Number 11, containing parts of number models representing undesirable patterns.

The second section focused on applying and evaluating segmentation methods. It comprised a CVL series number dataset consisting of a series of numbers in each image.

We trained the CNN model using the combined CVL and MNIST datasets, achieving exceptional performance. The trained model exhibited an accuracy of 98.42% and a low loss of 05.38%. These results were crucial as we relied on this well-performing model for predicting numbers obtained through the segmentation process.

We then presented and demonstrated the effectiveness of suggested segmentation methods in accurately predicting segmentednumbers.

The experimental study was conducted in three stages for the first two experiments and two stages for the third experiment. The segmentation process was performed using the sliding window technique and edge detection. In the first stage, segmentation methods were applied to the original image, utilizing the discussed preprocessing methods except for the skeleton technique.

In the second stage, segmentation methods were applied to preprocessed images, including the skeletonization process. The skeletonization helped capture the basic structure of the figures.

Lastly, in the third stage, segmentation methods were applied to images that underwent an encoding method.

The obtained results varied, with some being satisfactory and others disappointing. We found that the use of the skeleton algorithm and encryption did not significantly contribute to our work and even produced unsatisfactory results. Conversely, using the original image yielded positive outcomes as it preserved the essential characteristics of the numbers and played a crucial role in their identification.

The sliding window technique produced poor results, highlighting its dependency on the capabilities of the deep learning model and its classification ability to accurately recognize and differentiate between undesirable numbers and items.

On the other hand, the application of edge detection showed positive results, but it faced limitations when dealing with cursive numbers where handwritten numbers were connected. Some connected numbers were correctly recognized, while others were misidentified as odd numbers regardless of their original configuration. To address this limitation, we proposed a final experiment.

We found that the combination of both methods was a critical component of this study as it improved the segmentation process. However, as mentioned earlier, the sliding window technique requires a powerful deep learning model and an efficient classifier to achieve more precise and accurate results.

Given the vast scope of this field, our work serves as a proposal based on the results presented in this thesis. To fully leverage this work, we suggest the use of a powerful deep learning model and an effective classifier to achieve even more accurate results.

Bibliography

[1] "LeCun, Y., Cortes, C. and Burges, C.J.C. (1998) The MNIST Database of Handwritten Digits. New York, USA. http://yann.lecun.com/exdb/mnist/".

[2] A. B. A. H. Y. Shinde, "Handwriting Recognition on Filled-in Forms Using CNN. In: Singh, S.K., Roy, P., Raman, B., Nagabhushan, P. (eds) Computer Vision and Image Processing.," *CVIP 2020. Communications in Computer and Information Science, vol 1376. Springer, Singapore. https://doi.org/10.1007/978-981-16-1086-8_9.*

[3] "LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.".

[4] "Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning.".

[5] "[Online]. https://epynn.net/Pooling.html," [Online].

[6] "Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition.".

[7] "Choi, Hyungeun & Ryu, Seunghyoung & Kim, Hongseok. (2018). Short-Term Load Forecasting based on ResNet and LSTM. 1-6. 10.1109/SmartGridComm.2018.8587554.".

[8] "Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems".

[9] [Online]. Available: https://en.wikipedia.org/wiki/AlexNet#/media/File:Comparison_image_neural_networks.svg.

[10] "He, K., et al. (2016). Deep Residual Learning for Image Recognition.".

[11] "Choi, Hyungeun & Ryu, Seunghyoung & Kim, Hongseok. (2018). Short-Term Load Forecasting based on ResNet and LSTM. 1-6. 10.1109/SmartGridComm.2018.8587554.".

[12] "Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.".

[13] "Aslam, Muhammad. (2020). A robust CNN model for handwritten digits recognition and classification. 10.1109/AEECA49918.2020.9213530.".

[14] "Markus Diem, Stefan Fiel, Angelika Garz, Manuel Keglevic, Florian Kleber and Robert Sablatnig, ICDAR 2013 Competition on Handwritten Digit Recognition (HDRC 2013), In Proc. of the 12th Int. Conference on Document Analysis and Recognition (ICDAR) 2013, pp.".

[15] "Huseyin Kusetogullari, Amir Yavariabdi, Abbas Cheddad, Håkan Grahn and Johan Hall, 2019, "ARDIS: A Swedish Historical Handwritten Digit Dataset," Neural Computing and Applications, Springer. DOI: 10.1007/s00521-019-04163-3".

[16] "Markus Diem, Stefan Fiel, Florian Kleber, Robert Sablatnig, Jose M. Saavedra, David Contreras, Juan Manuel Barrios, Luiz S. Oliveira. ICFHR 2014 Competition on Handwritten Digit String Recognition in Challenging Datasets (HDSRC 2014). In Proceedings of th".

[17] "Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng Reading Digits in Natural Images with Unsupervised Feature Learning NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011.".

[18] ".J, Pradeep & Srinivasan, E. & Himavathi, S.. (2011). Diagonal Based Feature Extraction for Handwritten Alphabets Recognition System Using Neural Network. International Journal of Computer Applications. 8. 364-368. 10.1109/ICECTECH.2011.5941921.".

[19] "V. Shanthi,Noise Reduction and Pre-processing techniques in Handwritten Character Recognition using Neural Networks Magesh Kasthuri, EnathurKanchipuramPublished 2014".

[20] "M. Sezgin, B. Sankur, “Survey over image thresholding techniques and quantitative performance evaluation”, Journal of Electronic Imaging 13 (1) (2004) 146–168.".

[21] "M. Zaiz Faouzi, Les Supports Vecteurs Machines (SVM) pour la reconnaissance des caractères manuscrits arabes, Université Mohamed Khider-BISKRA, pp.15 ,15/07/2010.".

[22] "Chun Lei He, Ping Zhang, Jianxiong Dong, Ching Y. Suen, Tien D. Bui,The Role of Size Normalization on the Recognition Rate of Handwritten Numerals.".

[23] "Nawaz, S., & Kaur, A. (2019). A Survey on Handwritten Digit Recognition Techniques. Journal of Artificial Intelligence and Soft Computing Research, 9(3), 213-228.".

[24] "Amjad Rehman, Dzulkifli Mohamad and Ghazali Sulong,Implicit Vs Explicit based Script Segmentation and Recognition: A Performance Comparison on Benchmark.".

[25] "Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.".

[26] "P. Sermanet, S. Chintala and Y. LeCun, "Convolutional neural networks applied to house numbers digit classification," Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), Tsukuba, Japan, 2012, pp. 3288-3291.".

[27] "A. Gattal, C. Djeddi, Y. Chibani and I. Siddiqi, "Isolated Handwritten Digit Recognition Using oBIFs and Background Features," 2016 12th IAPR Workshop on Document Analysis Systems (DAS), Santorini, Greece, 2016, pp. 305-310, doi: 10.1109/DAS.2016.10.".

[28] "U. R. Babu, Y. Venkateswarlu and A. K. Chintha, "Handwritten Digit Recognition Using K-Nearest Neighbour Classifier," 2014 World Congress on Computing and Communication Technologies, Trichirappalli, India, 2014, pp. 60-65, doi: 10.1109/WCCCT.2014.7.".

[29] "A. Khan, MCS HOG Features and SVM Based Handwritten Digit Recognition System Hamayun, Faculty of Computer Studies, Arab Open University, Industrial Ardiya, Kuwait".

[30] "K. T. Islam, G. Mujtaba, R. G. Raj and H. F. Nweke, "Handwritten digits recognition with artificial neural network," 2017 International Conference on Engineering Technology and Technopreneurship (ICE2T), Kuala Lumpur, Malaysia, 2017, pp. 1-4, doi: 10.1109".

[31] "Gattal, A., Chibani, Y. & Hadjadji, B. Segmentation and recognition system for unknown-length handwritten digit strings. Pattern Anal Applic 20, 307–323 (2017). https://doi.org/10.1007/s10044-017-0607-x".

[32] "Shujing Lyu, Yue Lu, Hongjian Zhan, Handwritten Digit String Recognition using Convolutional Neural Network, 2018.".

[33] "Rabia KARAKAYA, Serap KAZAN; (2021), Handwritten Digit Recognition Using Machine Learning . Sakarya University Journal of Science, 25(1), 65-71, DOI: https://doi.org/10.16984/saufenbilder.801684".

[34] "R. Dey, R. C. Balabantaray and J. Piri, "A Robust Handwritten Digit Recognition System Based on Sliding window with Edit distance," 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), Bangalore, India, 20".

[35] "Amit Choudhary, Anand Nayyar, Saurabh Singh and Byungun Yoon Savita Ahlawat, Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN), Received: 25 May 2020; Accepted: 9 June 2020; Published: 12 June 2020.".

[36] B. K. S. C. Rajib Ghosh, "Handwritten Digit Recognition using Sliding Window Technique".*International Journal of Advanced Research in Computer Science and Software Engineering.*.

[37] "Dutta, Monish & Roy, Pritha & Datta, Sunanda & Nath, Asoke & Student, M. (2015). A New Method for Recognition of Handwritten Characters Using Edge Detection, Segmentation and Pattern Matching.".

[38 [Online]. Available: https://docs.anaconda.com/free/anaconda/index.html.

]

[39 [Online]. Available: https://docs.spyder-ide.org/current/index.html.
]

[40 "https://research.google.com/colaboratory/faq.html," [Online].
]

[41 [Online]. Available: https://docs.python.org/3/.
]

Table of Figure and Table: