August 2024
Master's Degree Thesis

Lightweight Methods for Data Work-Load Reduction in Resource-Constrained Computer Vision Systems through Abstraction, Fuzzy Decision Fusion, and the Statistical Method

Graduate School of Chosun University

Department of Computer Engineering

Mohammadreza Najafi

Lightweight Methods for Data Work-Load Reduction in Resource-Constrained Computer Vision Systems through Abstraction, Fuzzy Decision Fusion, and the Statistical Method

추상화, 퍼지 결정 융합 및 통계적 방법을 통해 리소스가 제한된 컴퓨터 비전 시스템에서 데이터 작업 부하를 줄이기 위한 경량 방법 연구

August 23, 2024

Graduate School of Chosun University

Department of Computer Engineering

Mohammadreza Najafi

Lightweight Methods for Data Work-Load Reduction in Resource-Constrained Computer Vision Systems through Abstraction, Fuzzy Decision Fusion, and the Statistical Method

Advisor: Prof. Jeong-A, Lee

This Thesis is submitted to Graduate School of Chosun University
in partial fulfillment of the requirements for a Master's degree

April 2024

Graduate School of Chosun University

Department of Computer Engineering

Mohammadreza Najafi

나자피 모하마드레자 사학위논문을 인준함

위원장 신석주

위 원 전찬준

위 원 이정아 (인)

2024년 5월

조선대학교 대학원

ACKNOWLEDGEMENTS

The completion of this thesis is attributed to the unwavering support, reassurance of my parents, whose indescribable patience has sustained them in my absence. I am profoundly thankful for their constant encouragement. Additionally, I extend my heartfelt appreciation to my wife, whose invaluable assistance has greatly contributed to my education and life journey.

I extend my deepest appreciation to Professor Lee, my supervisor, for the exceptional guidance and support provided throughout the course of this degree. My sincere gratitude is also extended to Dr. Gorgin, my research advisor, whose mentorship has played a pivotal role in shaping my research journey. His support, akin to that of a father, has been invaluable, contributing significantly to both my personal and academic growth.

I would like to express my gratitude to Dr. Fallah for consistently surprising me with innovative ideas in addressing challenging research points. His approachable and friendly demeanor has made our collaboration enjoyable and intellectually enriching. I consider myself truly fortunate to have worked alongside such dedicated and inspiring mentors.

In heartfelt acknowledgment, I extend my appreciation to all Iranian women worldwide for their outstanding companionship. Additionally, I am grateful for the generous hospitality of Mr. Mohammad Sina Karvandi and Mr. Mohammadhosein Gholamrezaie. To everyone who has contributed to my journey.

Contents

List of Tables

List of Figures

Acronyms

FCN	Fully Connected Neural Networ
MAC	Multiply–Accumulate
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DNN	Deep Neural Networks
CV	Computer Vision
ASL	American Sign Language
ISL	Indian Sign Language
BSL	Bangla Sign Language
OA	Opaque Abstraction
GA	Glass Abstraction
GPU	Graphics Processing Unit
TP	Trainable Parameters
DR	Dimension Reduction
IC	Image Compressio
QC	color quantization
RE	Region-of-Interest Extraction
SS	Superpixel Segmentatio
FO	Filtering Operation
FE	Feature Extraction
IR	Image Resizing
ROI	Region Of Interest
ADFA	Abstraction and Decision Fusion Architecture
ML	Machine Learning
AI	Artificial Intelligence
ANFIS	Adaptive Neuro-Fuzzy Inference System
SVM	Support Vector Machine
FG	FCN with Glass Abstraction
FO	FCN with Opaque Abstraction
FP	FCN with Peak Abstraction
SG	SVM with Glass Abstraction
SO	SVM with Opaque Abstraction
SP	SVM with Peak Abstraction

한 글 요 약

추상화, 퍼지 결정 융합 및 통계적 방법을 통해 리소스가 제한된 컴퓨터 비전 시스템에서 데이터 작업 부하를 줄이기 위한 경량 방법 연구

나자피 모하마드레자
지도교수: 이정아
컴퓨터공학과
조선대학교 대학원

컴퓨터비전 처리를 할 때, 사용할 수 있는 자원이 제한되어 있는 경우, 이미지 처리 시스템의 추론 능력을 향상시키면서도 사용하는 컴퓨팅 자원을 잘 관리하여 미세한 균형을 설정하는 것은 중요한 문제로, 임베디드 시스템, IoT 기기, 비용 효율적인 모바일 기기를 포함한 광범위한 범위에 적용된다. 본 논문은 추상화, 퍼지 의사 결정 융합, 통계적 방법의 세 가지 방법을 활용하여 이러한 문제를 해결한다.

추상화 방법을 통하여 원본 데이터를 다양한 관점에서 처리하여 획득한 이러한 데이터는 처리량이 감소된 경량 모델에 의해 독립적으로 처리될 수 있고, 독립적인 의사 결정 로짓을 생성하게 되고, 의사 결정 융합 과정을 통하여 최종 결과값이 결정된다. 통계적 방법은 이 과정, 특히 추상화 방법을 적용하여 생성되는 데이터로 결과값의 정확도 획득이 어려운 응용 분야에서 이 부분을 보완하기 위하여 활용된다.

본 논문에서 제안된 추상화, 퍼지 의사 결정 융합 및 통계적 방법 활용한 이미지 처리 아키텍처를 수화 분류, EMNIST 분류, 감정 분류와 같은 다양한 응용 분야에 적용하여 이의 효율성을 보였다. 수화 데이터셋에서 두 가지 데이터 추상화를 적용하고 완전 연결 신경망을 훈련하여 수화 분류를 수행했으며, 추상화와 퍼지 의사 결정 융합 의사 결정 기능을 통합하여 EMNIST 분류를 수행한 실험결과는 본 논문에서 제시한 경량방식에 의하여, 데이터 인식의 정확도가 높으면서도, 모델 크기와 MAC(곱셈-누적) 연산의 수가 최신의 계산 모델보다 훨씬 작아짐을 보였다.

Abstract

Lightweight Methods for Data Work-Load Reduction in Resource-Constrained Computer Vision Systems through Abstraction, Fuzzy Decision Fusion, and the Statistical Method

Mohammadreza Najafi
Advisor: Prof. Jeong-A, Lee
Department of Computer Engineering
Graduate School of Chosun University

Resource-aware image understanding seeks to establish a delicate balance by efficiently managing computational constraints while enhancing the inferential capabilities of image processing systems. Its applications span a wide range, encompassing embedded systems, IoT devices, and cost-effective mobile devices. This thesis addresses these challenges through three distinct methods: Abstraction, Fuzzy Decision Fusion, and the statistical method. The Abstraction method generates diverse views to create abstract representations of original data, processed independently by lightweight models producing independent decision logits. The final output is then obtained through a decision fusion tool. The statistical method complements these methods, particularly for applications where the Abstraction method encounters limitations. To evaluate the proposed architecture's effectiveness, we deployed the models in various applications such as Sign Language classification, EMNIST Classification, and Emotion classification.

In this work, we introduced two data abstractions, followed by the training of fully connected neural networks tailored to these abstractions for Sign Language classification. In addition we presented another abstraction for addressing the EMNIST classification problem, incorporating a Fusion Decision Function for improved accuracy. Our experiments on EMNIST and

Sign Language datasets validate the efficiency and high accuracy of the proposed architectures in data recognition. Notably, the model size and the number of MAC operations are significantly smaller than those of state-of-the-art computational models.

Index Terms: Resource-Awareness, Image Abstraction , Neural Network,Adaptive Neuro-Fuzzy Inference System, Sign Language Classification, Emnist Classification, Emotion Classification.

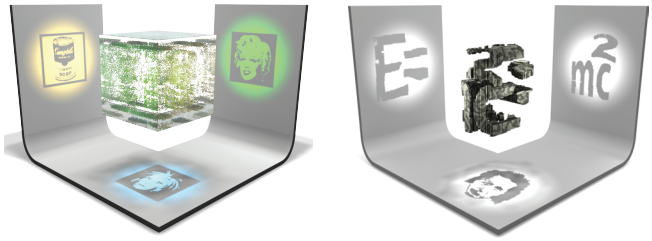
Chapter I.
Introduction

Intelligent computing comprises the development and enhancement of leading-edge computational techniques and algorithms with a primary focus on problem-solving and decision-making within the domains of Artificial Intelligence (AI) and Machine Learning (ML). It reshapes traditional computing and propels a digital revolution in the era of artificial intelligence [1]. This concept is embraced and implemented by numerous applications in diverse fields, including automation and health care [2], solving complex mathematical models [3], Internet of Things [4], smart city applications [5], natural language processing [6]. It empowers the systems to learn from data, reason and make decisions, interact with humans, and continuously improve. However, high intelligence often embraces processing large amounts of information and execution of computationally intensive algorithms to solve problems, make decisions, and learn from experience [7]. Consequently, there is usually a trade-off between the level of intelligence and the limitations of computation resources.

There are different approaches to enhancing the intelligence of systems. As a case in point, boosting productivity through creating diversity in processes increases the possibility of achieving positive results. Therefore, when individual models that process the data from different approaches to training come together and share their information, it augments the intelligence of systems. Following this approach, Ensemble Learning [8, 9] is a well-established machine learning technique that combines multiple models to improve the performance of a predictive model. It is based on the idea that by combining various models [10], each with its strengths and weaknesses, the resulting combined model can be more accurate and robust than any of the combined models. However, in ensemble learning, the main focus is on increasing accuracy by supporting the diversity of algorithms for miscellaneous data analysis [11] while resource-aware computing is not a top point of interest. Reducing the volume of data to be processed is An efficient approach in resource-aware computing [12–14]. It encompasses sampling, dimensionality reduction, data aggregation, data compression, filtering, and

data discretization. These techniques involve selecting representative subsets, reducing input variables, summarizing data, improving data encoding efficiency, eliminating irrelevant or redundant data, and transforming high-dimensional data into a simpler representation.

We have employed data abstraction as a high-potential approach to reducing the amount of data to be processed. It consists of focusing on data from defined points of view and representing them simplified to mitigate their complicatedness. Data abstraction plays a crucial role in the abbreviation of data and can be achieved through various techniques such as filtering [15], summarization [16], compression [17], quantization [18], and transformation and projection [19]. Although data abstraction can reduce the volume of data to be processed, it has several potential disadvantages, including information loss, over-simplification, the introduction of inaccuracies into the data, and the lack of data transparency, making it challenging to understand [20]. The multi-abstraction approach addresses these problems by providing multiple perspectives on data from diverse points of view [15, 21].



(a) Glass cube. **(b)** Opaque object.

Figure 1: Two shadow sculptures that represent multiple viewpoints of objects [22].

As an intuitive example, Fig.1 illustrates two sculptural arts where the $2D$ shadows are cast by two $3D$ sculptures [22]. In the first one, the object is a glass cube, and the light source inside it has caused each part of the data to be projected with a special shape on a dimension (Fig.1a). In the second one, the object is opaque where some light sources are at different angles, each one projects the data as a special shape on a dimension (Fig.1b). Here, by removing each projection, our understanding of the sculpture (data) becomes deficient. This example demonstrates that keeping multiple perspectives can improve the model's understanding of the data and support intelligence.

Preparing multiple abstractions and processing them by different models may lead to distinctive decisions, where integrating them into a single result is called decision fusion [23, 24]. It is an important aspect of a collaborative system that can provide a way to combine the predictions of its subsystems into a single, more robust, and accurate decision.

In this paper, we present a three-layer architecture named ADFA with the aim of resource-aware image understanding. ADFA is based on providing multiple abstractions for the input image, processing each abstracted data by a lightweight computational model, and fusing their decisions. It should be mentioned that ADFA pursues a fundamentally distinct objective and approach when compared to parallel ensemble learning [9]. The main focus of attention in ensemble learning is to find an optimal combination of individual models that results in improved accuracy, reduced variance, and increased generalization. In other words, ensemble learning emphasizes the diversity of data processing tools, whereas the ADFA architecture emphasizes data abstraction and the diversity of perspectives in simplification.

The aim of ADFA is to remove the complexities from data to be suitable for processing with a combination of lightweight models. With the intention of showcasing the potential of the proposed architecture, we have developed several ADFA-based models for handwriting character classification. The experimental findings affirm that the proposed architecture provides an advanced direction for the development of resource-aware models with remarkable accuracy.

A. Research Motivation

The MAC unit stands as a cornerstone in digital signal processing, playing a pivotal role in various applications. Fig.2 offers a comprehensive profiling of six contemporary DNNs, emphasizing the significance of MAC operations in their execution. In this representation, each row constitutes a stacked bar chart delineating the proportion of operations, including MAC, comparison (comp), addition (add), division (div), and exponent (exp), performed during

a single inference run across the respective DNN. All six charts reveal that the preponderance of a DNN's computational workload is attributed to MAC operations. Consequently, a primary strategy for curtailing computations in neural networks revolves around the reduction of MAC operations.

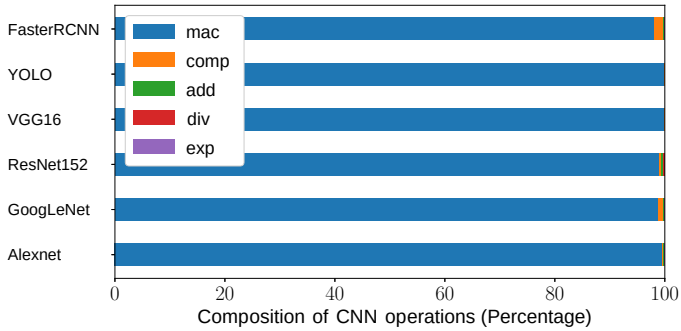


Figure 2: Percentage of MAC, comparison (comp), addition (add), division (div) and exponent (exp) operations for different networks [25].

To enhance operational efficiency, our strategy involves the utilization of compact computing models by optimization of data representation to reduce the data workload for processing. This approach aims to achieve a harmonious balance, ensuring a reduction in the number of operations while preserving and enhancing accuracy. Hereafter, we enumerate several frequently employed techniques for image data reduction.

One fundamental technique for data workload reduction is resizing images [26]. It is crucial for computational models like neural networks, ensuring consistent input, enhancing computational efficiency, and improving model generalization. However, image resizing, typically involving pixel value interpolation, may introduce artifacts, potentially compromising the original image's integrity and hindering classifier proficiency in pattern recognition [27].

Transforming multi-dimension data into a lower-dimension space simplifies data complexity while retaining essential information [28]. Similarly, image downsampling, achieved through techniques like subsampling and averaging, is crucial for reducing computational complexity in convolutional neural networks (CNNs) while maintaining essential details [29]. Common downsampling methods, such as max-pooling and average pooling, con-

tribute to feature extraction. However, striking the right balance between information loss and computational efficiency is challenging. Determining the optimal number of downsampling operations is critical, as an excessive number may result in information loss, while too few may increase computational demands [30, 31].

Compression algorithms reduce the volume of data by efficiently encoding images while preserving visual quality [32]. They are vital for both storage and the efficient transmission of images as well as real-time recognition of dynamic gestures from continuous data streams [33, 34]. Also, encoding transformations can highlight discriminative features in data, improving classification accuracy. Moreover, they have the potential to filter out noise or irrelevant information, leading to more robust models. However, developing techniques that preserve valuable information while eliminating redundant and irrelevant information from images is still a research case [35].

Color quantization efficiently addresses memory and processing constraints by condensing the number of distinct colors in an image [36]. The impact of color quantization on image classification varies based on task requirements. While it simplifies data, there is a potential risk of losing valuable color information.

Region-of-interest extraction [37] enhances data transmission and simplifies the analysis by identifying and isolating specific regions within an image or video frame. This segmentation isolates crucial areas for in-depth examination, streamlining the processing pipeline.

Superpixel segmentation streamlines image analysis by reducing the entities to consider, grouping pixels with similar characteristics into larger regions [38]. However, the segmentation accuracy is constrained by the limited representation of raw spectral signatures, which contain intricate details crucial for precise segmentation. The inherent complexity of unprocessed spectral data makes distinguishing subtle variations challenging, impacting the precision of segmentation algorithms.

Applying filters and smoothing operations can simplify noisy images by reducing high-frequency noise or irrelevant details, resulting in cleaner data for analysis [39, 40]. Enhancing image quality, filters, and smoothing opera-

tions improves the accuracy of subsequent image processing and classification tasks.

Feature extraction techniques identify and extract relevant features from images, such as edges, textures, colors, or key points [41]. They reduce the dimensionality of image data, making it more manageable for subsequent processing to enhance processing speed and reduce memory requirements, aligning with resource-aware principles as well as real-time processing. Additionally, feature extraction techniques emerge as critical facilitators as they extract relevant information from visual data, emphasizing key elements of sign language gestures while suppressing noise and irrelevant details [39,42]. This information accelerates processing speed and improves classification accuracy, facilitating a more precise recognition of signs and gestures. Naturally, to reduce processing requirements, feature extraction can be followed by feature reduction or feature dimensionality reduction operations [43–45].

Certainly, in real-world applications, a holistic approach involves employing a combination of strategies to reduce input data size, thereby elevating the efficiency and precision of the system to new levels.

B. Research Objectives

This research embarks on an in-depth exploration of resource-aware image understanding, with a primary objective to establish a nuanced equilibrium by effectively managing the intricate computational constraints inherent in image processing systems. The scope of applications extends comprehensively across embedded systems, IoT devices, and economically viable mobile devices. This thesis systematically tackles the multifaceted challenges associated with resource-aware image understanding through the meticulous investigation of three distinctive Methods: Abstraction, Fuzzy Decision Fusion, and the statistical Method.

The Abstraction Method, a pivotal aspect of this research, involves the generation of diverse views to craft abstract representations of the original data. These abstract representations undergo independent processing by

lightweight models, resulting in the generation of decision logits. The final output is subsequently derived through the utilization of a sophisticated decision fusion tool. This methodology not only showcases the adaptability of the Abstraction Method but also underlines its potential for facilitating independent decision-making in the computational realm.

Complementing the Abstraction Method, the Fuzzy Decision Fusion method plays a crucial role in refining accuracy, particularly in scenarios where the inherent limitations of the Abstraction Method become apparent. This Method introduces a Fusion Decision Function, contributing to improved accuracy in addressing the EMNIST classification problem. The fusion of decisions derived from independent abstract representations enhances the overall robustness and accuracy of the computational model.

The statistical Method, introduced in Chapter V., emerges as an innovative solution for Emotion Classification in resource-aware devices. This Method is strategically applied in scenarios where the Abstraction Method encounters specific limitations, presenting a nuanced approach to address computational challenges. The deployment of the proposed architectures in practical applications, such as Sign Language classification, EMNIST Classification, and Emotion classification, forms the cornerstone of assessing their efficacy.

Throughout the experimental phase, the research scrutinizes the efficiency and high accuracy of the proposed architectures in data recognition. Notably, a key focus is placed on quantitative measures, such as the model size and the number of Multiply-Accumulate (MAC) operations, providing a meticulous comparative analysis against state-of-the-art computational models. This research, with its comprehensive exploration of resource-aware image understanding and the tailored Methods devised, is poised to significantly contribute to the advancement of computational models in the realm of image processing, with broad applications across diverse domains.

C. Contributions

This research contributes novel Methods for resource-aware image understanding. The proposed methods are demonstrated to be effective and efficient through their application in various image recognition tasks, addressing the challenges posed by limited computational resources in diverse contexts.The contributions of this research work can be summarized as follows:

- Resource-Aware Image Understanding Framework:

  The research introduces a comprehensive framework for resource-aware image understanding. This framework is designed to efficiently manage computational constraints while enhancing the inferential capabilities of image processing systems. This is crucial for applications in embedded systems, IoT devices, and cost-effective mobile devices where computational resources are often limited.

- Abstraction Method:

  The Abstraction Method is proposed as a key component of the framework. It generates diverse views of original data, creating abstract representations processed independently by lightweight models. This approach allows for the production of independent decision logits, which are then fused using a decision fusion tool to obtain the final output. This Method contributes to the efficiency and flexibility of the image understanding process.

- Fuzzy Decision Fusion:

  Fuzzy Decision Fusion is introduced as a Method to enhance accuracy in decision-making. It is particularly applied in addressing the EMNIST classification problem, where the fusion decision function is employed to improve the accuracy of the classification process. This contribution demonstrates the adaptability of the proposed framework to different types of image recognition tasks.

- Applications and Validation:

  The proposed architecture is validated through practical implementations in diverse applications, including Sign Language classification, EMNIST Classification, and Emotion classification. The experiments conducted on EMNIST and Sign Language datasets showcase the efficiency and high accuracy of the proposed architectures in data recognition.

- Efficiency Metrics:

  The research emphasizes the efficiency of the proposed architectures by highlighting significantly smaller model sizes and a reduced number of Multiply-Accumulate (MAC) operations compared to state-of-the-art computational models. This aspect is crucial for resource-aware devices where minimizing computational requirements is a primary consideration.

D. Thesis Organization

This dissertation is organized as follows: In Chapter II., an overview of the requisite subjects is presented. Chapter III. delineates two distinct Abstraction Methods and employs them in the context of Sign Language Classification. Chapter IV. introduces an additional abstraction method tailored for EMNIST classification within the Ensemble Architecture. Within this chapter, a Fusion Decision Function is applied to enhance the classification process within our Ensemble Architecture. Chapter V. introduces a statistical method aimed at reducing data workload, implemented in the application of Emotion classification. Lastly, Chapter VI. summarizes our proposed methodologies, provides an account of the outcomes pertaining to data workload reduction methods, and conducts a comparative analysis with state-of-the-art classification models in terms of accuracy, model size, and power usage. Additionally, potential avenues for future enhancements are delineated.

Chapter II.
Background

This section provides some characteristics of Machine Learning, Vector Machine (SVM), Fully Connected Neural Network (FCN), and Adaptive Neuro-Fuzzy Infer- ence System (ANFIS). These tools are employed to construct lightweight computational models, which serve as components of the our proposed models.

A. Machine Learning

Conceptually speaking, Machine Learning (ML) strives to find all these ways to take advantage of the available data and provide insights out of it, without trying to explicitly define the relationships between the data (i.e., without using explicit instructions). Rather "abstractly", ML tries to reach a conclusion in a similar fashion to the way humans process information, using some prior knowledge or experience. That is, ML algorithms can "learn" from representative examples of input (and output) data.

A more structured definition by [46] denotes that "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E". Applications of ML algorithms include but are not limited to learning how to recognize visual object categories [47], learning how to translate linguistic data [48], learning how to play GO [49] or, even learning how to suggest our favorite music [50] and so on and so forth.

We can map ML algorithms in the following broad categories; Supervised Learning (e.g., Classification, Regression), Unsupervised Learning (e.g., Clustering, Dimensionality Reduction), Semi-Supervised Learning & Reinforcement Learning. In this thesis, we will deal mostly with Differential Privacy within Supervised Learning. Therefore, a succinct introduction will be provided in this area. For an in-depth and comprehensive reading in ML literature, we highly suggest [51–55].

1. A Probabilistic Perspective

In supervised learning, we suppose that there is an "input" space $X \in \mathbb{R}^p$ and an "output" space Y . Depending on the supervised learning task, Y can be either real valued or categorical. The training data S, consists of n samples that we suppose that they are drawn independently and identically distributed (i.i.d.) from a probability distribution $\mu(z)$ on $Z = X \cdot Y$: $(x_1, y_1), ...(x_n, y_n)$, that is, $z_1, ...z_n$. Therefore, we are looking for a function $f(X)$ that is able to predict a target Y given the (new) input values X, that is $y_{pred} = f_S(x_{new})$. We use the conditional probability of y given x, (i.e., $p(y|x)$) to model this relationship as follows:

$$p(x, y) = p(x|y) \cdot p(x). \tag{II..1}$$

If y is real valued, then we say that we have a regression problem and so we are looking for a mapping such as $f : \mathbb{R}^p \mapsto \mathbb{R}$ If the output space is a categorical value of y in range of k, the mapping we are looking for is $f : \mathbb{R}^p \mapsto \{1, ..., k\}$, and hence we call it Classification.

2. Loss Function

Whether the problem is Classification or Regression, it is clear that there exists a hypothesis space H, which is the space of functions that our learning algorithm is allowed to search in. This is also known as function approximation, as we are basically trying to find the function f out of a set of possible functions $F \in H$ that best maps inputs to outputs. Therefore, we need a way to quantify "how good", or "how bad" a function f is for the task.

We can now introduce the Loss function L: Given some particular pair of inputs x and outputs y the function

$$L(f(x), y). \tag{II..2}$$

quantifies the error of the prediction f(x) when the true output is y. The choice for loss function depends on the task, Hinge Loss1 is common for

classification with Support Vector Machines (SVM), while the square loss (or L2 loss) is popular within regression tasks, in Deep Learning classification algorithms, the cross-entropy loss is the most commonly used one (i.e., for Softmax output layer when dealing with multi-class problems).

Given some function f, a loss function L and the true distribution $p(x,y)$ over inputs x and y we can define the expected or (true) Risk as follows:

$$R_{true}(f) \stackrel{\triangle}{=} \mathbb{E}[L(f(x),y)] \int\int p(x,y)L(f(x),y)dxdy. \tag{II..3}$$

The risk measures how much, on average, it costs to use f as our prediction algorithm. The idea is to make R_{true} small, however the main problem here is that we don't know μ we defined earlier (i.e., the true distribution of our data points). We can approximate this by the empirical error; Given some function f, a loss function L and a training set S that is made by n data points, we can now define the empirical risk on that training set as:

$$R_{emp}(f) \stackrel{\triangle}{=} \frac{1}{n}\sum_{i}^{n} L(f(x_i),y_i). \tag{II..4}$$

In the above definition, it is typical to include a regularization parameter G, such as:

$$R_{emp}(f) = \underbrace{\frac{1}{n}\sum_{i}^{n} L(f(x_i),y_i)}_{\text{Risk (training error)}} + \underbrace{\lambda G(f)}_{\text{Regularizer}} \tag{II..5}$$

This parameter is included in order to impose a complexity penalty on the loss function and prevent overfitting (i.e., it tries to implement Occam's Razor, the simpler the model the better), where $G(f)$ measures the complexity of the prediction function f and λ controls the strength of the complexity penalty.

The goal in this case, is, to *minimize* this quantity, that is, we need to FInd that $f^* \in H$ which minimizes the empirical risk:

$$f^* = arg\min_{f \in H} R_{emp}(f). \tag{II..6}$$

Restricting the space of functions H to those parametrized by θ, we can re-write Equation II..6 as follows:

$$\theta^* = arg \min_{\theta \in \mathbb{R}^d} \left\{ \frac{1}{n} \sum_{i}^{n} L(f(x_i), y_i) + \lambda G(\theta) \right\} \tag{II..7}$$

Where θ, accounts for all learnable parameters. Therefore, solving the aforementioned regularized convex optimization problem and finding θ^*, leads to the "learnt model". We note that under certain loss functions, for example, the Negative Log- Likelihood loss (NLL), which is widely used for training Deep Neural Networks, the Empirical Risk Minimization is basically reduced to Maximum Likelihood Estimation (MLE).

More concretely, suppose that our loss function acts like a posterior density Q, with parameterization θ, given some data points (x,y), that is $Q(\theta|x,y)$. Let's re-define the Loss in Equation II..2 as follows:

$$L(f(x;\theta),y) \stackrel{\triangle}{=} -\log(Q(x,y|\theta)) \tag{II..8}$$

which we will refer to it as negative log-likelihood loss. We can show that minimizing the (non) regularized empirical risk (Equation II..4) under this loss function (Equation II..8) is equivalent to estimating the maximum likelihood:

$$arg \min_{\theta \in \mathbb{R}^d} \sum_{i}^{n} (-\log(Q_\theta(x_i,y_i|\theta))) = arg \max_{\theta \in \mathbb{R}^d} \sum_{i}^{n} \log(Q(x_i,y_i|\theta))) \tag{II..9}$$

In the above formulation a scaling factor $\frac{1}{n}$ could be added to make it clearer as this does not affect the *argmin*/*argmax* of the log probability.

Starting from our initial assumption of the posterior and expressing it by Bayes:

$$Q(\theta|x,y) = \frac{Q(x,y|\theta)\,Q(\theta)}{Q(x,y)}. \tag{II..10}$$

then by taking the logarithm of this expression:

$$\log Q(\theta|x,y) = \log Q(x,y|\theta) + \log Q(\theta) - \log Q(x,y). \tag{II..11}$$

we can now omit the last factor since it is a constant (does not depend on θ):

$$\log Q(\theta|x,y) = \log Q(x,y|\theta) + \log Q(\theta). \tag{II..12}$$

Thus, if we would like to maximize this posterior such that we obtain the optimal model parameters θ^* over all training samples (x_i, y_i), we can write:

$$\theta^* = arg \max_{\theta \in \mathbb{R}^d} \left\{ \underbrace{\sum_i^n \log(Q(x_i,y_i|\theta))}_{\text{Likelihood)}} + \underbrace{\log(Q(\theta))}_{\text{Prior)}} \right\} \tag{II..13}$$

In case the last term, the prior, is uninformative (i.e., $Q(\theta) = 1$) we return to the original formulation of MLE (Equation II..9). Otherwise, this is called Maximum-a-Posteriori Estimation (MAP), as the role of the prior is basically to encode some prior knowledge about which models are more or less likely. Under the ERM framework, this prior can be interpreted as the regularization term, while the likelihood can be interpreted as the empirical risk. Finally, we note that depending on the distribution that the error/risk is modeled around, we obtain different estimators; minimizing the log-likelihood under Gaussian distribution results in a Least Squares loss, while a Bernoulli distribution would give the Binary Cross-entropy loss.

3. Optimization

These kinds of optimization problems are usually solved by Gradient Descent (GD) algorithms that use the slope of the surface of the loss function to find some optimal region, where the error is minimized. In the case of Deep Neural Networks (DNN), strictly speaking, we usually end up with highly non-convex loss surfaces [56] (i.e., we can get multiple local minima [57]). Thus we use Stochastic Gradient Descent (SGD) class solvers, which in their simplest form, they just shuffe the samples and update the gradient after each example. Practically, a variation of this method is widely used where the datapoints are basically packed and aggregated into mini-batches to

update the gradient at each iteration (mini-batch SGD) [58]. Empirically speaking, this method considerably speeds up the training procedure (over GD), and usually converges to reasonably good solutions.

Briey, SGD is an iterative algorithm for solving the (regularized) convex optimization problem in Equation II..7. It begins from some initial model parameters θ_0, and, at every step t, it updates the model parameters as follows:

$$\theta_{t+1} = \theta_t - \eta(\lambda \bigtriangledown G(\theta_t) + \bigtriangledown L(f(x_t, \theta_t), y_t)). \tag{II..14}$$

where η is the learning rate, $\bigtriangledown G(\theta_t)$ is the gradient of the regularizer, and $\bigtriangledown L(f(x_t; \theta_t), y_t)$ is the gradient of the loss function, evaluated on a single example (x_t, y_t). In the case of DNN's, this gradient can be calculated using automatic differentiation with respect to each layer's learnable parameter set θ^{layer}.

B. Deep Neural Networks

DNNs in artificial intelligence attempt to mimic this biological system by using nodes (or artificial neurons) and connections (or artificial synapses) to perform computations [59]. These nodes receive input signals, which are then weighted and combined before being passed on to other nodes in the network. This process continues until an output signal is produced.

Neural networks possess a significant benefit in their capacity to acquire knowledge and accommodate fresh information. This is accomplished via a training procedure wherein the weights of node connections are modified according to input-output pairs. Through successive weight adjustments, the network becomes proficient in identifying patterns and making predictions using novel input data [60].

Artificial neurons serve as the fundamental components of artificial neural networks (depicted in Figure 5). They are modeled after the biological neurons in the human brain, which receive input signals from other neurons, process the information, and produce output signals [62].

The artificial neuron receives a set of input signals that are multiplied

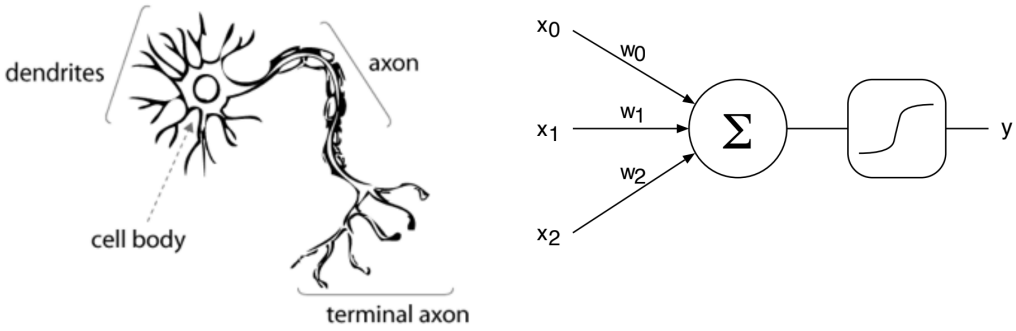


Figure 3: Showing a comparison between a biological neuron and its artificial counterpart (adapted from [61]).

by respective weights. The weights govern the magnitude of the connection between the input signal and the neuron. The weighed input signals are then summed up and passed through a non-linear activation function. The activation function determines whether the neuron should fire or not based on the weighted inputs. If the input signal is strong enough, the neuron will fire, producing an output signal. The weights of the artificial neuron can be adjusted during the training process to optimize the neuron's response to certain inputs. The functionality of an artificial neuron can be represented by Equation (II..15), where the activation function is denoted by F, bias by b, and N represents the number of input activations.

$$y = F(\sum_{i=1}^{N} x_i w_i + b) \tag{II..15}$$

The organization of a neural network typically involves arranging artificial neurons into layers and connecting them in a specific way. Within a feed-forward neural network architecture, the neurons are organized in a directed acyclic graph, permitting connections solely between neurons in contiguous layers. The neurons are organized into layers according to their location within the network. The initial layer, known as the input layer, receives the input data into the network. The output layer is the final layer in the network, where the output signals are produced. Any layers between the input and output layers are referred to as hidden layers, as they do not directly interact with the input or output data.

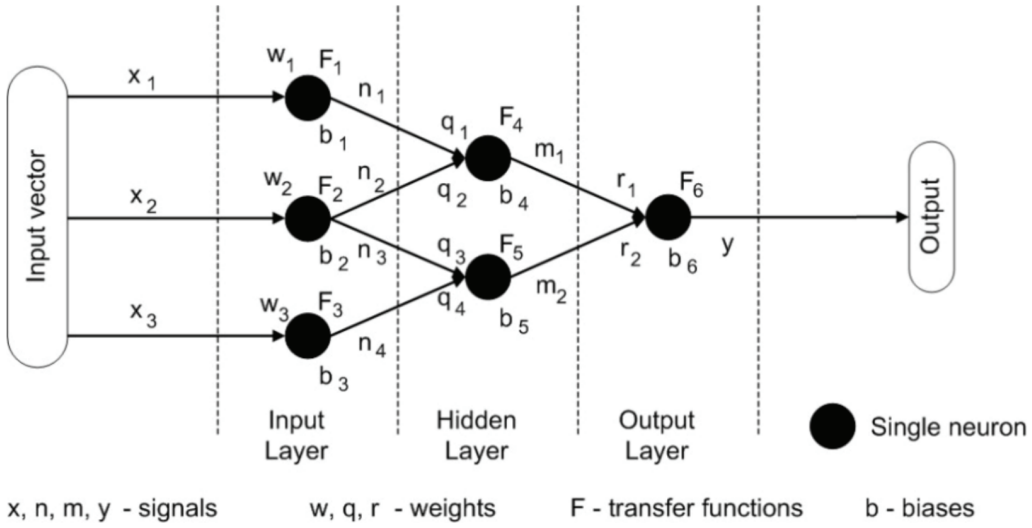


Figure 4: Feed-forward artificial neural network (adapted from [62]).

In a fully connected layer, each neuron in one layer is connected to every neuron in the adjacent layer. This allows for a high degree of connectivity between the neurons and enables the network to process complex data sets. The number of layers and neurons in each layer can vary depending on the specific task the network is designed to perform. In general, deeper neural networks with more layers tend to perform better on complex tasks, but may require more computational resources and longer training times. Figure 4 illustrates a basic feed-forward artificial neural network consisting of multiple layers, which is used for analytical description and described by sets of equations (II..16), and (II..17).

$$\begin{aligned} n_1 &= F_1(w_1 x_1 + b_1) \\ n_2 &= F_2(w_2 x_2 + b_2) \\ n_3 &= F_2(w_2 x_2 + b_2) \\ n_4 &= F_3(w_3 x_3 + b_3) \end{aligned} \tag{II..16}$$

$$
\begin{aligned}
m_1 &= F_4(q_1x_1 + q_2x_2 + b_4) \\
m_2 &= F_5(q_3x_3 + q_4x_4 + b_5) \\
y &= F_6(r_1m_1 + r_2m_2 + b_6)
\end{aligned}
\tag{II..17}
$$

C. Fully Connected Neural Network

FCNs are supervised ML tools highly applicable to solving classification and regression problems. An FCN is a multi-layer network of artificial neurons such that each neuron receives all of the outputs in the previous layer as its input. However, its performance can be influenced by factors such as the size and architecture of the network, the amount of training data, and the quality of the data. Fig.5 illustrates the model of a primary artificial neuron that generates an output y_j according to its input vector x. The training of a neuron includes finding weights w and bias b for Eq.II..18 using a training dataset. These parameters are commonly referred to as trainable parameters. Correspondingly, the input vector x corresponds to the outputs generated by neurons in the preceding layer.

$$
y = \sigma(w.x + b) \tag{II..18}
$$

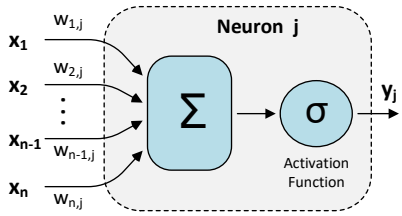


Figure 5: The model of an artificial neuron.

Here, σ is the activation function. Nonlinear activation functions play a crucial role in neural networks by introducing nonlinearities into the network. Eq.II..19 present the ReLU, a low-cost nonlinear activation function

commonly used in different types of neural networks.

$$\sigma(z) = max(0, z) \tag{II..19}$$

To normalize the output of FCN, a Softmax activation function is applied on vector y of the last layer. Eq.II..20 presents the Softmax activation function applied on a single output y_s.

$$\sigma(y_s) = \frac{exp(y_s)}{\sum_{j=1}^{j=n} exp(y_j)} \tag{II..20}$$

The size of an FCN model and the number of required operations to classify input data depend on the number of layers as well as the number of neurons in each layer. On the other hand, the number of neurons in the first layer of FCN is equal to the size of input data, and the number of neurons in the last layer is proportional to the number of possible outputs (i.e., the number of classes in a classification task). Moreover, the number of neurons in hidden layers can vary depending on the complexity of the data and the architecture of the neural network. Therefore, reducing the complexity and the size of input data significantly impacts the size of the FCN model and the number of operations.

D. Support Vector Machine (SVM)

SVM is a supervised ML tool capable of solving various classification and regression problems. The primary concept underlying SVM is identifying a set of hyperplanes as decision boundaries in the feature space that achieves the maximum possible separation between the data classes. The number of hyperplanes significantly affects the computational complexity and memory requirements. A higher hyperplane count can lead to increased resource consumption, potentially compromising efficiency and scalability in resource-constrained environments.

The kernel trick [63] enables SVM to handle problems that are not linearly separable. It maps the input data to a higher-dimensional space, where

nonlinear hyperplanes can be used to classify input data. Suppose that $\{(x_1,y_1),\ldots,(x_m,y_m)\}$ is a training dataset, where each $x_i \in \mathscr{R}^n$ is a data point and $y_i \in \{-1,1\}$ is its label. Eq.II..21 presents a nonlinear hyperplane that classifies the input data x_s binarily, where α_i and b are constant values, and $K(x_s,x_i)$ is a kernel function. The common kernels used in SVM are linear, polynomial, and Radial Basis Functions (RBF). Here, Eq.II..22 is an RBF kernel function that x_s and x_i are its inputs and γ is a constant value.

$$y_s = sign\left(\sum_{i}^{n}(\alpha_i\ y_i\ K(x_s,x_i)) + b\right) \tag{II..21}$$

$$K(x_s,x_i) = exp\left(-\gamma\ \|x_s - x_i\|^2\right) \tag{II..22}$$

According to Eq.II..21 and Eq.II..22, both the size of an SVM model and the number of operations it conducts to classify input data are contingent upon the size of the training dataset.

E. Adaptive Neuro-Fuzzy Inference System

The ANFIS is a hybrid ML model that compounds the benefits of artificial neural networks and fuzzy logic to provide accurate and reliable predictions and classifications. It is a type of fuzzy inference system that uses neural networks to adjust and optimize its parameters for modeling complex relationships between input variables and output variables.

Fuzzy inference systems come in several types, each offering distinct advantages that make them better suited for specific applications. Two popular types of fuzzy inference systems are Mamdani and Sugeno models. The Mamdani model [64] is commonly used in systems where linguistic interpretations and transparency are important, such as expert systems, decision-making, and control systems with human-like reasoning. As well, the Sugeno model [65], also known as the Takagi-Sugeno-Kang model, is suitable for applications where precise calculation is the primary concern, such as system identification, control systems with more mathematical control laws, and

signal processing.

We utilized the Sugeno model [66] in our experiments to handle the image processing tasks with a high degree of accuracy. It is structured with five layers, namely, the Fuzzification layer, Rule layer, Normalization layer, Consequent layer, and Aggregation layer. The Fuzzification layer consists of fuzzy sets that map input values to fuzzy membership degrees. Each fuzzy set is characterized by a membership function. Eq.II..23 presents a Gaussian membership function, where ε_i and c_i are the parameters obtained for each membership function μ_i during the ANFIS training phase.

$$\mu_i(x, \varepsilon_i, c_i) = exp\left(-\frac{(x-c_i)^2}{2\varepsilon_i^2}\right). \tag{II..23}$$

The Rule layer contains a set of IF-THEN rules that generate the vector of weights w called the firing strength of the rules. The rules combine all fuzzy linguistic values of different input variables, while each linguistic value is represented by a fuzzy membership function in the Fuzzification layer. Eq.II..24 presents the firing strength of r^{th} rule of an ANFIS model, where R_r indicates the set of indices of its inputs.

$$w_r = \prod_{j \in R_r} \mu_j \tag{II..24}$$

The Normalization layer normalizes vector w of firing strengths by Eq.II..25.

$$\bar{w}_r = \frac{w_r}{\sum_i w_i}. \tag{II..25}$$

The Consequent layer consists of a set of adaptive nodes. The output of i^{th} node in this layer is computed by Eq.II..26, which is the product of the normalized weights and a linear combination of the inputs. Here, each parameter $\alpha_{i,k}$ is obtained during the training of the ANFIS model.

$$y_i = \bar{w}_i \sum_k \left(\alpha_{i,k} x_k\right) \tag{II..26}$$

The Aggregation layer consists of one single fixed node that generates

the output by performing the summation of all incoming signals according to Eq.II..27.

$$z = \sum_i y_i \tag{II..27}$$

The number of trainable parameters plays a crucial role, as its increment leads to a proportional growth in the model size, ultimately resulting in higher memory requirements and potentially increased computational complexity during both the training and inference stages. According to Eq.II..23 and Eq.II..26, the set of trainable parameters of an ANFIS model includes c_i and σ_i for each membership function μ_i as well as the consequent parameters $\alpha_{i,k}$. Of course, in our study, ANFIS was employed to integrate the decisions of two or three models, thereby limiting the maximum number of inputs to 13.

F. Kullback-Leibler (KL) Divergence

The Kullback-Leibler (KL) divergence, also known as relative entropy and denoted as $D_{KL}(P\|Q)$, is a fundamental concept in information theory and statistics. It quantifies the difference between two probability distributions, P and Q, by measuring how one distribution diverges from another. It is defined in Equation II..28.

$$D_{KL}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{II..28}$$

where $P(i)$ and $Q(i)$ are the probabilities assigned by distributions P and Q to outcome i, respectively.

In the context of continuous distributions, the KL divergence is expressed from Equation II..29.

$$D_{KL}(P\|Q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx \tag{II..29}$$

where $p(x)$ and $q(x)$ are the probability density functions of distributions P and Q, respectively.

Chapter III.

Abstraction of Image Data: Novel Image Data Representation

Given our primary objective of minimizing computational complexity while upholding precision in sign language recognition, we developed a model designed to significantly reduce data volume while maintaining a high level of comprehension. Figure 6 provides a comprehensive overview of our computational model, which is primarily focused on enhancing efficiency in terms of operations and memory usage. To achieve these objectives, we have developed an abstraction method for simplifying and efficient encoding of hand gesture images. This method [67] significantly reduces the image size by shortening data representation while retaining crucial information for accurate classification. Our data abstraction plays a pivotal role in mitigating data complexity and enables us to employ FCNs with a limited number of hidden layers to classify compact hand gesture data. Subsequently, we will concentrate on elucidating the proposed abstraction method.

Figure 6: An overview of the proposed computational model.

A. Methodology

The proposed abstraction comprises four distinct steps, namely Resizing, Grayscal, Filtering, and Encoding, as illustrated in Fig.7. In the first step, the input is resized into a 28×28 image. In the subsequent steps, image grayscale conversion is applied, followed by background elimination and the creation of a filtered binary image through a straightforward processing step. During the Filtering step, a simple formula has been applied, demonstrating efficacy, particularly for images that don't have complex backgrounds. In

this regard, Formula III..1 was initially employed to determine a threshold value.

Figure 7: Steps of the proposed hand gesture abstraction method.

$$\tau = \alpha \ \frac{\sum_{i,j}^{28} D_{ij}}{28 \times 28} = \alpha \ \mathbb{E}_{1 \leq i,j \leq 28} \left[D_{ij} \right]. \tag{III..1}$$

The Eq.III..1 computes the binarization threshold value τ by multiplying the fee parameter α with the average value of the data pixels D. Then, given that the target images typically encompass approximately 70% background and considering the fixed size of the input image at 28×28, the alpha coefficient in the experiments was set at the value of 0.7. In the following, to

eliminate the background and create a binary image using threshold τ. The final step in our hand gesture image abstraction employs a heuristic technique inspired by shadow art, resulting in data dimensionality reduction. This process encodes the binary 28×28 image into a 1×112 vector.

In some image understanding and classification tasks, the intricate patterns and textures within an image may be deemed extraneous noise. Within such contexts, the primary objective typically revolves around discerning the geometry of objects contained in the image, necessitating a comprehensive exploration of the subject from diverse perspectives. As an intuitive example, Fig.8 is a hand shadow illustrating a boy [68]. Here, comprehending the subject by analyzing the shape of the hands is more challenging compared to understanding the resultant shadow displayed on the screen. The human interpretation of Fig.8 serves as a source of inspiration, suggesting the feasibility of extending the analysis of the subject from diverse perspectives to include an examination of the interactions between light radiation and the subject. Hence, if the image is rendered binary, delineating the hand from the background and preserving its comprehensibility to human observers, the imperative lies in encoding the outcomes of the light interaction with the targeted object (in this case, the hand). This encoding is indispensable for the development of a computational model capable of discerning and comprehending visual information. In this context, we have formulated two distinct encodings for the comprehension of hand gestures, namely Opaque and Glass.

Figure 8: A hand-shadow which is the interaction of light radiation with hand [68].

1. Opaque Abstraction

The nomenclature "Opaque" is selected due to the conceptualization of the analyzed shape (in this context, hand gesture) as an opaque object impervious to light penetration. Consequently, the interaction of light sources with this object facilitates the comprehension of its outer boundary. Figure 9 illustrates the Opaque encoding process. Here, the gray box presents four assumed interactions of light with the object. In the first row, the solid yellow line placed on one of the sides of each figure's frame, indicates an array of light sources that shine on the object, and the dashed line arrows are the corresponding light rays. Each atomic light source radiates in a straight line horizontally or vertically, where the abbreviations indicate the direction of light rays and include the letters T (Top), D (Down), B (Bottom), U (Up), R (Right), and L (Left). Also, white pixels ($value = 0$) are considered transparent, and black pixels ($value = 1$) are considered opaque. As a result, light is unable to pass through black pixels, creating a bright area between the light source and the outer boundary of the object. The geometry of the bright area depends on the angle of light radiation and the outer border of the desired object. To encode this geometry, a one-dimensional vector, matching the number of these light sources, is employed. The associated value in the vector element corresponds to the distance of the respective light source to the point where its light intersects with the object.

The Opaque abstraction process transforms a 28×28 image into four separate vectors, each of length 28. Formulas III..2 to III..5 present the mathematical representation of these transformations for every $m \times n$ image, where, in our case, $m = n = 28$. Upon concatenating these vectors, the resulting abstraction is a single vector of length 112, which is displayed in Fig.9.

$$TD[i] = min\big(\{m\} \cup \{j | image[j][i] = 1\}\big). \tag{III..2}$$

$$BU[i] = m - max\big(\{0\} \cup \{j | image[j][i] = 1\}\big). \tag{III..3}$$

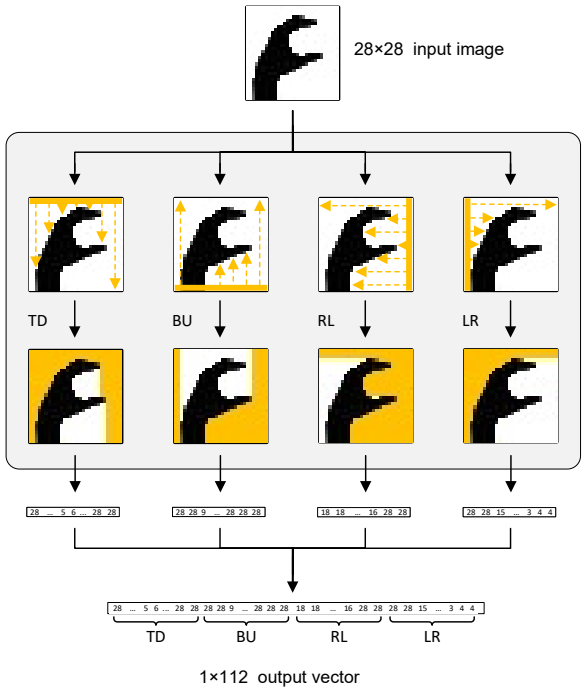


Figure 9: The interaction of light with the object in the Opaque abstraction.

$$RL[i] = min\big(\{n\} \cup \{j | image[i][j] = 1\}\big). \tag{III..4}$$

$$LR[i] = n - min\big(\{n\} \cup \{j | image[i][j] = 1\}\big). \tag{III..5}$$

2. Glass Abstraction

The term "Glass" is designated for the second abstraction due to the nature of the studied shape (in this case, hand gesture) being perceived as a non-uniform semi-transparent object. In this abstraction, the shadow contrasts with the transparency of the light path. Consequently, the interaction of light sources with this object is leveraged to comprehend both the row and column density of pixels. Figure 10 displays the Glass encoding process. In this instance, the gray box presents four assumed light interactions with the object. In the first row, the solid yellow line placed on the center of each figure's frame, indicates an array of light sources that shine on the

object, and the dashed line arrows are the corresponding light rays. Here, the abbreviation M (Middle) is new, where the notation ”MU” designates ”middle-to-up”, indicating that the light sources are centrally positioned within the frame, with their illumination directed upward. In this encoding scheme, the image is treated as an object through which light passes, where white pixels are considered fully transparent and black pixels are considered partially transparent. Therefore, as light traverses the figure, a grayscale shadow manifests on the frame's border. The intensity of darkness in each shadow pixel correlates with the degree of blurring in the path it traverses, proportionate to the number of black pixels encountered. Thus, the value assigned to each dimension of the resultant vector is established as the count of black pixels along its light trajectory.

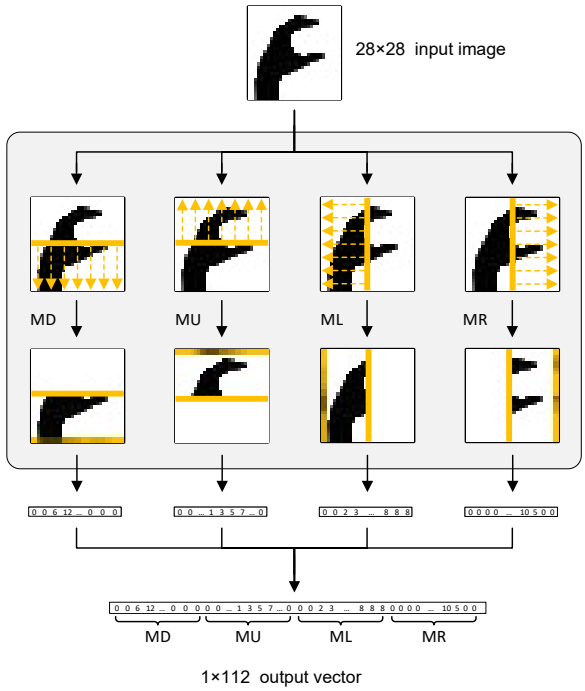


Figure 10: The interaction of light with the object in the Glass abstraction.

This abstraction process transforms a 28×28 image into four separate vectors, each of length 28. Formulas III..6 to III..9 present the mathematical representation of these transformations for every $m \times n$ image, where, in our case, $m = n = 28$. Upon concatenating these vectors, the resulting

abstraction is a single vector of length 112.

$$MD[i] = \sum_{m/2}^{m} image[j][i]. \tag{III..6}$$

$$MU[i] = \sum_{1}^{m/2} image[j][i]. \tag{III..7}$$

$$ML[i] = \sum_{n/2}^{n} image[i][j]. \tag{III..8}$$

$$MR[i] = \sum_{1}^{n/2} image[i][j]. \tag{III..9}$$

In the chapter VI., we explicitly term the abstraction encompassing Opaque coding as Opaque Abstraction (OA) and the one involving Glass coding as Glass Abstraction (GA). Analogously, the neural networks trained under each of these abstractions are specifically denominated as OA and GA models, respectively.

Chapter IV.
Information Fusion of Lightweight Computation Models

The overall structure of the proposed architecture in this chapter is depicted in Fig.11, which consists of three layers Abstraction, Computation, and Fusion [69]. The abstraction layer consists of a set of transformations (abstractions) $\{A_1, A_2, \ldots, A_n\}$ for organizing and presenting the input data from several views. In this context, A_i represents the i^{th} perspective which is an interpretation of the data, facilitating processing and problem-solving through a lightweight computation model. Within the Computation layer, a set of lightweight models $\{M_1, M_2, \ldots, M_m\}$ process their corresponding views individually. Here, each model M_j processes the output of an abstraction A_i according to its perspective on the input data and generates an independent output called its decision. Ultimately, in the Fusion layer, all the separate decisions made in the preceding layer are combined to arrive at the final decision. The rest of this section elaborates on the characteristics of each layer.

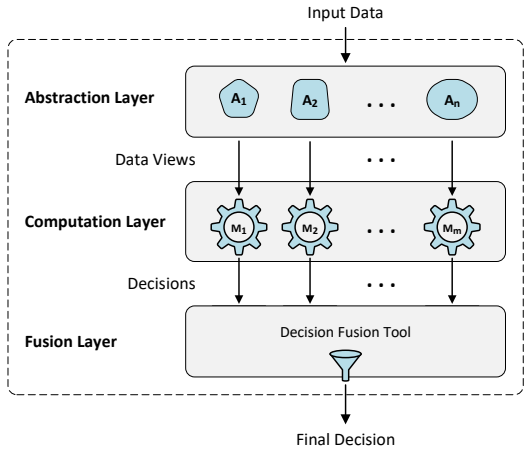


Figure 11: The proposed three-layer architecture ADFA.

A. Abstraction Layer

An abstract view is a well-defined in previous chapter III. that abbreviates the entire data from a particular perspective. When defining an abstract view, there is no need to focus on extracting specific features from data; rather, the focus is on a perspective that makes data simpler by removing unnecessary information. Certainly, it is advantageous for each view to yield simple outputs, thereby obviating the need for a complex computational model during processing. Also, the information conveyed by the entirety of viewpoints should exhibit sufficient diversity to facilitate well-informed decisions regarding the data. The information provided by the set of all viewpoints should be diverse enough to support well-informed decisions about the data.

B. Computation Layer

Each computational model M_j is constructed (trained) according to an abstraction view A_i. Therefore, it only processes the corresponding abstraction. It should be noted that there is a possibility for two models with distinct structures, such as an SVM and an FCN, to operate on the same abstraction. Due to the variance in the structures and input data utilized by these computational models, it is highly probable that they will produce distinct outputs.

C. Fusion Layer

Within this layer, a decision fusion tool is integrated. It is capable of combining multiple predictions stemming from various models to arrive at a conclusive decision. It improves the accuracy and robustness of decision-making by combining the strengths of multiple models and reducing the impact of individual model limitations. The main characteristics of this tool are the capability to integrate multiple sources of information, robustness

in individual model failures, and accuracy improvement by combining the strengths of multiple models.

D. Methodology

This chapter focuses on the handwritten data recognition problem to outline the methodology of constructing ADFA-based models. In this context, the method of test data preparation, the experiment design, and the approach for developing different layers of ADFA-based models are sequentially described.

1. Data

We have utilized the EMNIST [70] handwritten data in our experiments. It is a collection of 28×28 pixel gray-scale images of handwritten digits and letters. The pixel value of each image is an integer number ranging from 0 to 255. Table 1 presents the three EMNIST datasets employed in our experiments. For example, EMNIST Balanced includes 47 classes of English letters and digits, where $75,000$ images are utilized for training ($60,000$ for the models in layer 2 and $15,000$ for the decision tool in layer 3) and $37,500$ images are used for training ($30,000$ for the models in layer 2 and $7,500$ for the decision tool in layer 3). The table also contains additional statistics and information on utilizing the Letters and MNIST datasets.

2. Experimental Design

In this section, the experimental design is described in four steps. The initial three steps present the systematic development of the fundamental compo-

Table 1: The EMNIST datasets [70] employed for the experiments.

Name	CLS	No. Training		No. Testing		Total
		Layer2	Layer3	Layer2	Layer3	
MNIST	10	10,000	10,000	3,000	3,000	26,000
Letters	37	60,000	15,000	30,000	7,500	112,500
Balanced	47	60,000	15,000	30,000	7,500	112,500

nents of proposed (ADFA-based) models, following a layer-by-layer approach. Furthermore, the fourth step involves characterizing the structure of the various ADFA-based models that are generated using these components, along with describing the method of conducting the experiments.

2..1 Step 1 (Defining the abstractions)

In the first layer of the ADFA-based models, three data abstractions called Glass, Opaque, and Peak are utilized. Glass and Opaque have explained in chapter III. and Peak is defined based on different views inspired by two sculpting techniques in Shadow Art [22] (see Fig.1)

Fig.12 presents ten views that form Glass, Opaque, and Peak abstractions. The solid yellow line inside the figures indicates the light sources that shine on the object, and the dashed line arrows are the corresponding light rays. The abbreviations used for some views indicate the direction of light rays and include the letters M (Middle), B (Bottom), D (Down), T (Top), U (Up), L (Left), and R (Right). For example, MU means middle-to-up.

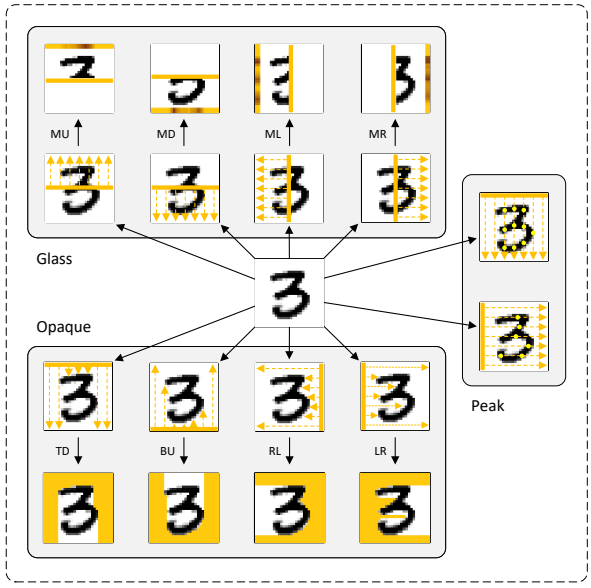


Figure 12: Three abstractions of a handwritten digit 3.

As defined in previous chapter the Glass abstraction treats the image as an object through which light passes, with the transparency of its pixels

being proportional to their values. In this abstraction, when light passes through the figure, a gray-scale shadow is formed on the opposite side. The darkness intensity of each shadow pixel corresponds to the amount of blurriness in the path it passes. Furthermore, in the Opaque abstraction, the image pixels are considered opaque. As a result, light is unable to pass through them, creating shadows behind these pixels. The shape of these shadows depends on the angle of light radiation and the outer border's shape of the desired object.

In the Peak abstraction, the image is treated as a digital elevation model, where each pixel's value represents the elevation of the corresponding point. Afterward, for every horizontal or vertical light ray, the number of peaks it passes through is tallied.

The Glass and Opaque abstractions have identical representations and map a 28×28 input image to a vector of length 112. Here, each element of the vector corresponds to a light source pixel, and the range of its values is the Real interval $[0, 1]$. Also, the Peak abstraction maps a 28×28 input image to a vector of length 56, where the value associated with each light ray is the number of passing peaks starting from it.

2..2 Step 2 (Preparing computational models)

To provide computational models of ADFA, three SVMs and three FCNs have been trained according to Glass, Opaque, and Peak abstractions on each dataset. Table 2 delivers a brief description of these models associated with the abbreviation that we use in the rest of the paper.

Table 2: The abbreviated name of the six basic computational models employed in the second layer of ADFA-based models.

		Abstraction		
		Glass	**Opaque**	**Peak**
Model	**FCN**	FG	FO	FP
	SVM	SG	SO	SP

In the SVM models (SG, SO, and SP), we adopted the RBF presented in Eq.II..22 as the kernel function, as it has proven to be highly effective in

classifying non-linear data. Regarding the FCN models (FG, FO, and FP), the input and output layers are tailored to the size of the corresponding abstraction and the number of output classes, respectively. Additionally, the layout of hidden layers is adjusted based on the complexity of the classification task. Specifically, all the basic models classifying MNIST data have one hidden layer with 128 neurons. On the other hand, all the models classifying Letters or Balanced data consist of three hidden layers with 128, 256, and 128 neurons, respectively. Moreover, ReLU is utilized as the activation function in all hidden layers, while Softmax is employed in all output layers.

2..3 Step 3 (Preparing the decision fusion tool)

We have employed an ANFIS in the third layer to take advantage of fuzzy logic in the fusion of decisions. The training of the ANFIS models is done according to the decision result of its input models on the same training data set. We gently note that our developed ADFA-based models undergo a dual-phase training process. The initial stage involves training fundamental models within the second layer of the architecture, followed by a second stage of ANFIS training utilizing the established second-layer models. Fig.13 illustrates the ANFIS model for fusing the decisions made by m models. In fact, the output decision of each basic model M_i in the Computation layer that is sent to ANFIS is the four tuples $(L_{i1}, T_{i1}, L_{i2}, T_{i2})$, which are the two output values with the highest degree of possibility, including their degree of trust. Therefore, ANFIS receives $4m$ inputs from m models; $2m$ output values, and $2m$ trust levels of the output values. Also, the first layer of ANFIS includes $8m$ fuzzification functions, each L_{ij} value, and each T_{ij} value are mapped respectively by one and three Gaussian membership functions (Eq.II..23) to fuzzy membership degrees.

The number of inference rules in the second layer of the ANFIS is 3^{2m}, where each rule uses a minimization operator to combine the fuzzy degrees obtained by applying the membership functions on the disjoint inputs and producing a fuzzy output. In the Normalization layer, the i^{th} node calculates the ratio of the i^{th} rule's strength to the sum of all rules' strengths.

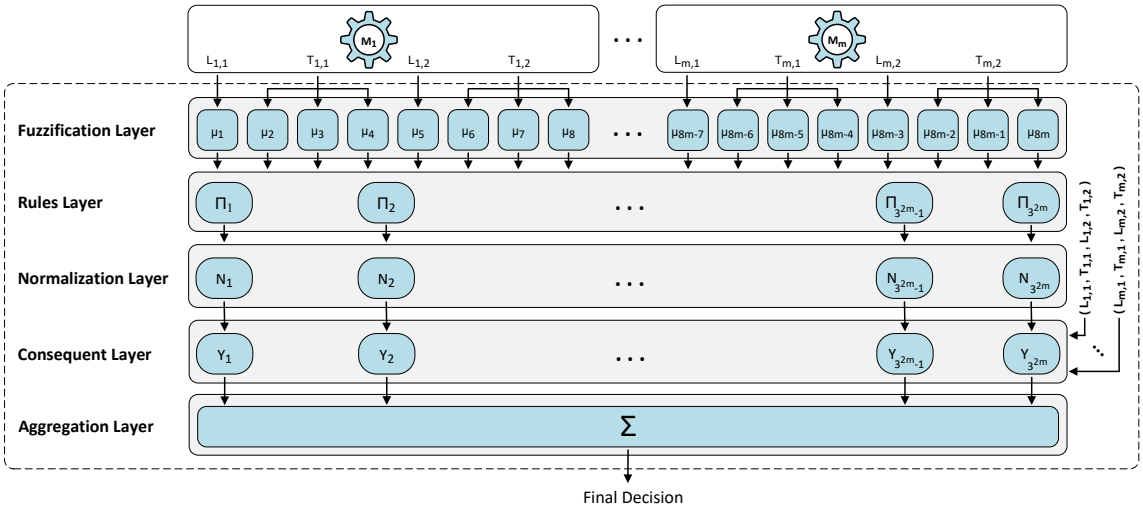


Figure 13: The ANFIS model used for fusing the decisions of m models.

2..4 Step 4 (Performing the experiments)

As depicted in Fig.13, the main distinction between the developed ADFA-based models lies in the set of basic computational models utilized in the second layer. On the other hand, ANFIS served in the third layer to fuse their decisions. Hence, when referring to these models, we specify their names as ANFIS inputs. For instance, ANFIS(FO,SG) is a model in which two glass and opaque abstractions are in the first layer, FO and SG basic models are in the second layer, and ANFIS is in the third layer. The experiments are conducted independently on the datasets but using a consistent methodology. First, we trained six basic models as shown in Table 2. Subsequently, we selected appropriate ones to construct various ADFA-based models by combining them and training the embedded ANFIS model. In line with the research objectives, the appropriateness of the basic models is evaluated based on three criteria: the number of MAC operations, the model size, and their compatibility with the ADFA structure, while aiming for high accuracy. In the next section, the experimental result of evaluating these seven models on the MNIST dataset is discussed according to the four measurements; accuracy, number of MAC (Multiply-Accumulate) operations, model size, and number of trainable parameters.

Chapter V.

Statistical Method for Image Data Work-Load Reduction in Emotion Recognition

In this section, we aim to enhance existing methodologies by incorporating statistical techniques for the classification of emotional datasets ([71]). The dataset under consideration comprises 28,709 training images and 7,178 test images distributed across seven distinct classes: anger, disgust, fear, happiness, neutrality, sadness, and surprise. As depicted in Figure 15, a selection of sample images from this dataset is presented alongside their corresponding labels. It is evident from the illustration that the dataset primarily encompasses facial expressions portraying various emotional states across different individuals.As illustrated in Figure 14, the proposed method's overview is divided into four steps: Pre-processing, Abstraction, Classification, and Fusion which are elaborated upon below.

A. Pre-processing

The contours of the eyes and lips play a pivotal role in conveying emotional expressions on the face. The subtle nuances of happiness or sadness can often be discerned by closely examining these facial features. In the realm of classifying images within emotion datasets, our attention is drawn to these critical attributes as they serve as key indicators of emotional states. In the process of image classification within the dataset, each image m is initially transformed into grayscale format and resized to 100×100 pixels. Subsequently, the openCV library [72] is employed to approximately determine

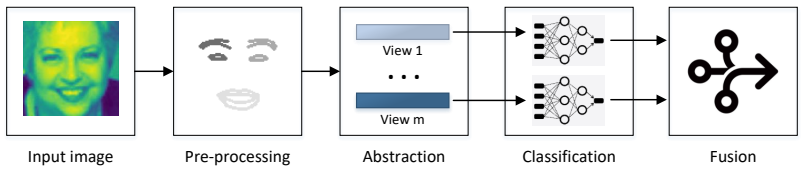


Figure 14: An overview of the proposed computational model.

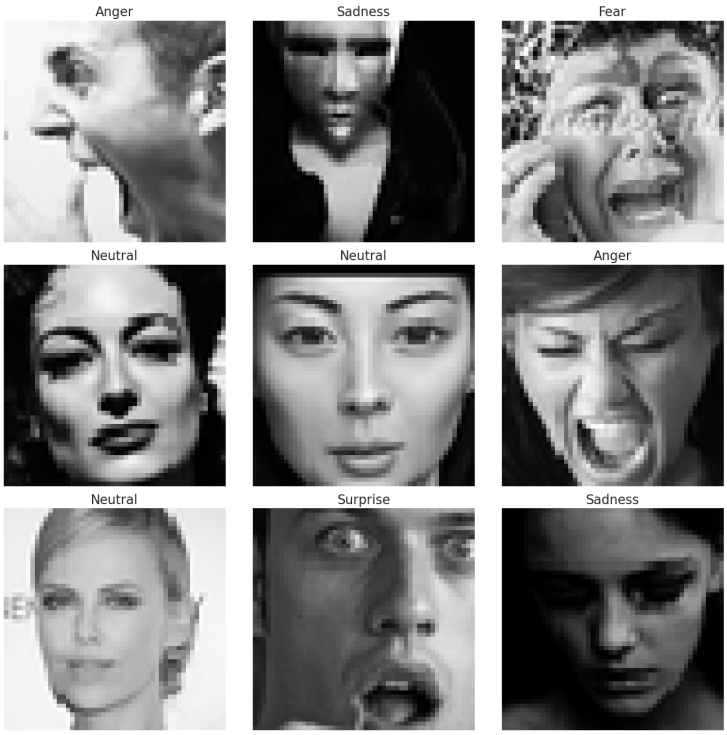
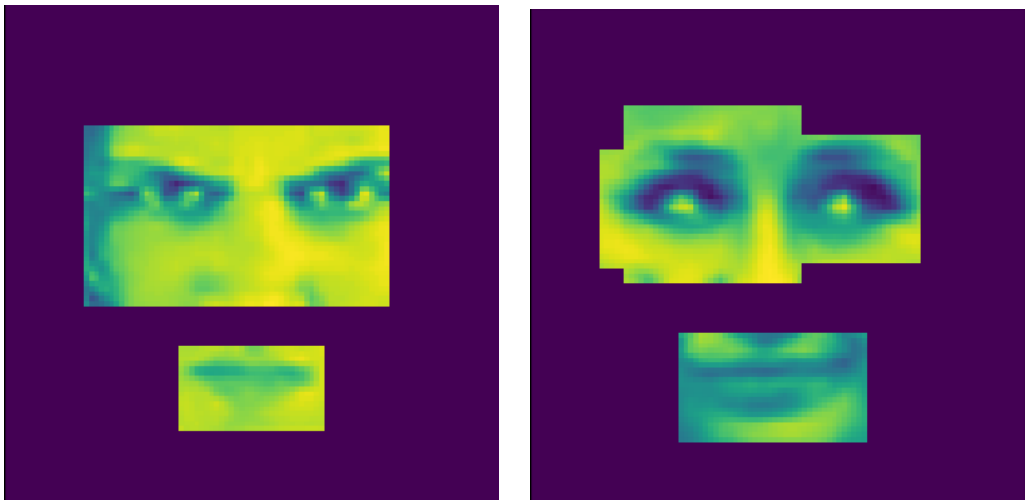


Figure 15: Images extracted from the emotion dataset alongside their corresponding labels.

the regions corresponding to the eyes and lips in all images. These regions are then cropped from each image, as depicted in Figure 16. As a result of this step, a dataset of grayscale images denoted as M is obtained, with dimensions $100 \times 100 \times 1$, wherein all pixel values outside the eyes and lips regions are set to 0. In the subsequent phase, a 100×100 array denoted as W is generated, initialized with zeros according to Equation V..1. Here, M_k represents the k^{th} image, while i and j denote the column and row indices of the image, respectively. Finally, all values of W are normalized within the range of $[0,1]$, and a masked matrix is generated as depicted in Figure 17.

$$W[i,j]_{new} = \begin{cases} W[i,j] + 0 & if \quad M_k[i,j] = 0 \\ W[i,j] + M_k[i,j] & if \quad M_k[i,j] \neq 0 \end{cases} \tag{V..1}$$

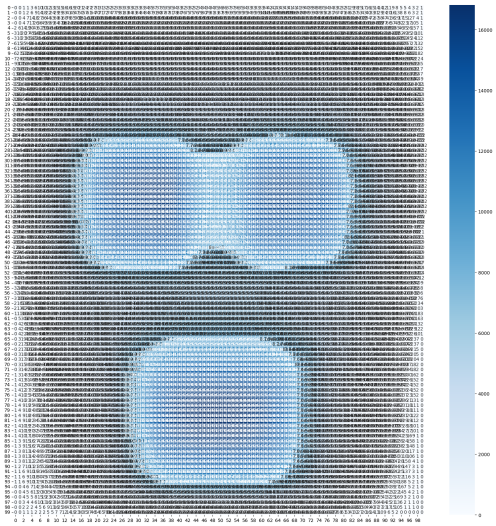


(a) (b)

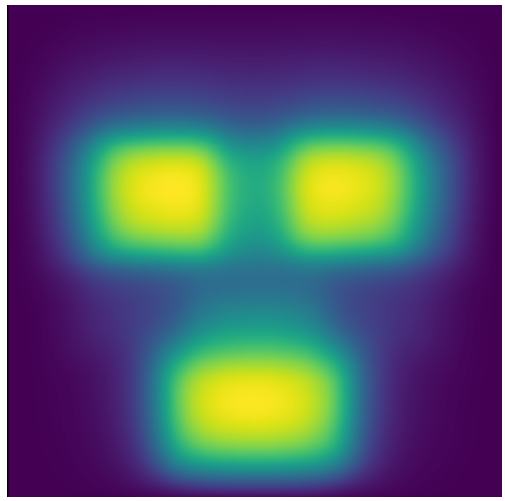
Figure 16: Cropping the regions corresponding to the eyes and lips in every image.

Utilizing the mask matrix W illustrated in Fig 17(b), we are able to produce a weighted image, M^*, of m through element-wise multiplication with W, Equation V..2. This operation assigns zero values to less significant pixels, thereby emphasizing the crucial regions corresponding to the eyes and lips. As depicted in Fig 18(a) for the image of a smiling face, and Fig 18(b) showcasing the resultant output, this multiplication process effectively highlights the salient features indicative of emotions.

$$M^*[i,j] = m[i,j] \cdot W[i,j] \tag{V..2}$$



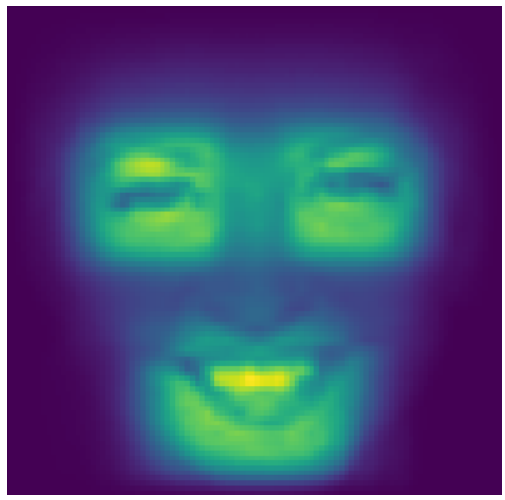
(a) The heatmap matrix of W which represent its values befor normalization .



(b) The matrix W after normalization and creates mask matrix.

Figure 17: Mask matrix, W, generated with M images.

(a) The Happy face image, m.



(b) Result of multiplication of Happy face image with the mask matrix, and generating M^*.

Figure 18: Generating M^* by multiplying W with m.

In the domain of image processing and computer vision, the task of locating points within an image with the maximum gradient holds significant importance. Such points often represent edges, corners, or other distinct features critical for various applications such as object detection, image segmentation, and scene understanding. The process of identifying these points

involves analyzing the gradient magnitude of the image, which signifies the rate of change in intensity across different regions. In this study we investigate methodologies for efficiently detecting points with the maximum gradient in the M^* image. By leveraging techniques from signal processing, mathematical morphology, and machine learning, we developed robust algorithms capable of accurately pinpointing these salient locations. at first Horizontal Gradient (G_x) and Vertical Gradient (G_y)) calculated with Equation V..3, V..4. after that Gradient Magnitude (∇f) Calculated from Equation V..5, in all these equation i and j denote the column and row indices of the M^*, respectively.

$$G_x[i,j] = M^*[i,j+1] - M^*[i,j-1] \tag{V..3}$$

$$G_y[i,j] = M^*[i+1,j] - M^*[i-1,j] \tag{V..4}$$

$$|\nabla f|[i,j] = \sqrt{(G_x[i,j])^2 + (G_y[i,j])^2} \tag{V..5}$$

At the end, as illustrated in Equation V..6, the local gradient maximum pixels attain a value of 255, while all other pixels in the image M^* are assigned a value of 0. Equation V..6 identifies all edge points in Figure 18(b), enabling the detection of eye and lip contours, as demonstrated in Figure 19. By applying a threshold to the formula, we achieve enhanced detection of eyes and lips, as depicted in Figure 20, across four different emotional expressions.

$$M^*[i,j] = \begin{cases} 255 & \text{if } |\nabla f|[i,j] \geq |\nabla f|[i',j'] \text{ for all } (i',j') \text{ in a local neighborhood}(i,j) \\ 0 & \text{otherwise} \end{cases} \tag{V..6}$$

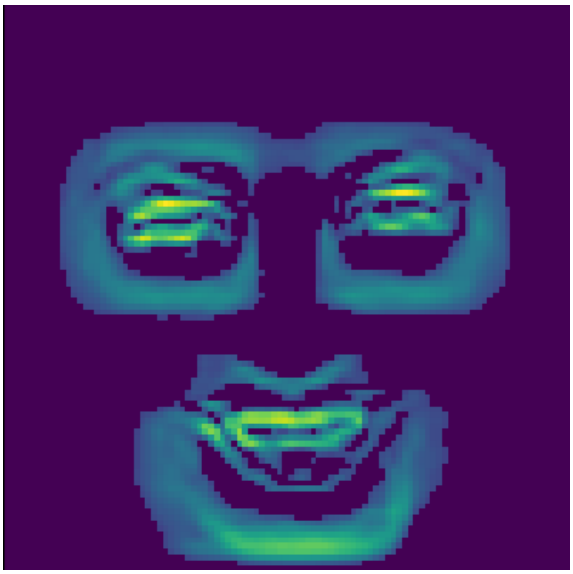


Figure 19: Detecting edge points by finding local gradient maximum of pixels

B. Abstraction

In this specific application, we implement two abstraction techniques in conjunction with two fully connected models serving as the classification model, shown in Fig 24, accompanied by an ANFIS model acting as the decision model, shown in Fig 25. As depicted in Fig 21 one of the abstraction techniques, referred to as Opaque abstraction and introduced in Chapter III., is utilized to generate a 4×100 array, denoted as $\mathrm{OA}(M^*)$, from the input image M^*.

Another abstraction method, denoted as "BA", is derived from the array of Face Blendshapes features, which calculated with [73] library, in addition to the Kullback-Leibler divergence (D_{KL}) between the eyes region and lips region, resulting in a 1×53 array for each M^*. The computation of $\mathrm{BA}(M^*)$ for the image M^*, depicted in Fig 22, has been performed and visualized in Fig 23.

Figure 20: Detection of eyes and lips contours clearly across four different emotional expressions.

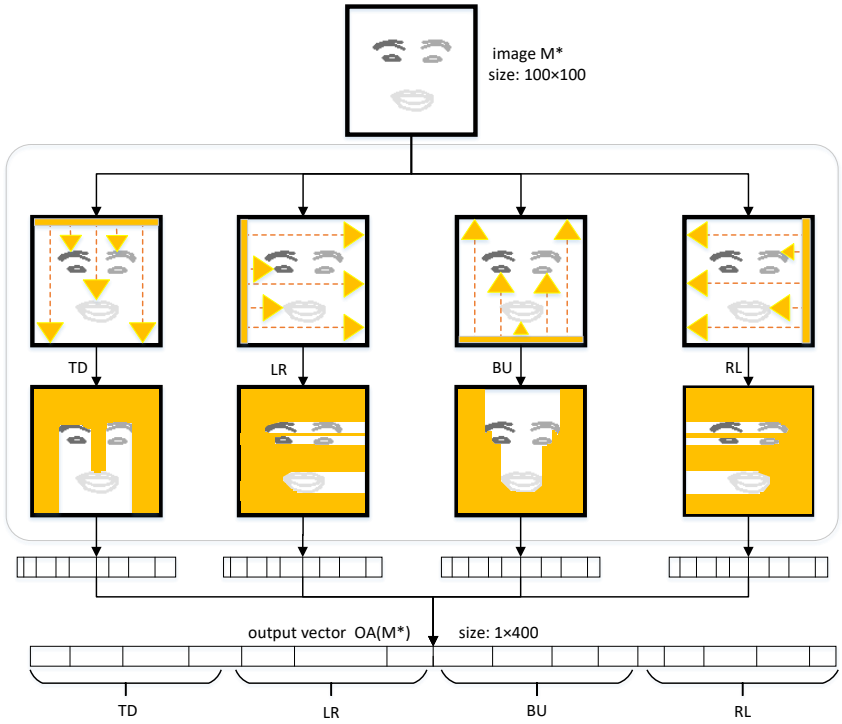


Figure 21: The interaction of light with the object in the Opaque abstraction.

Figure 22: The M^* sample image

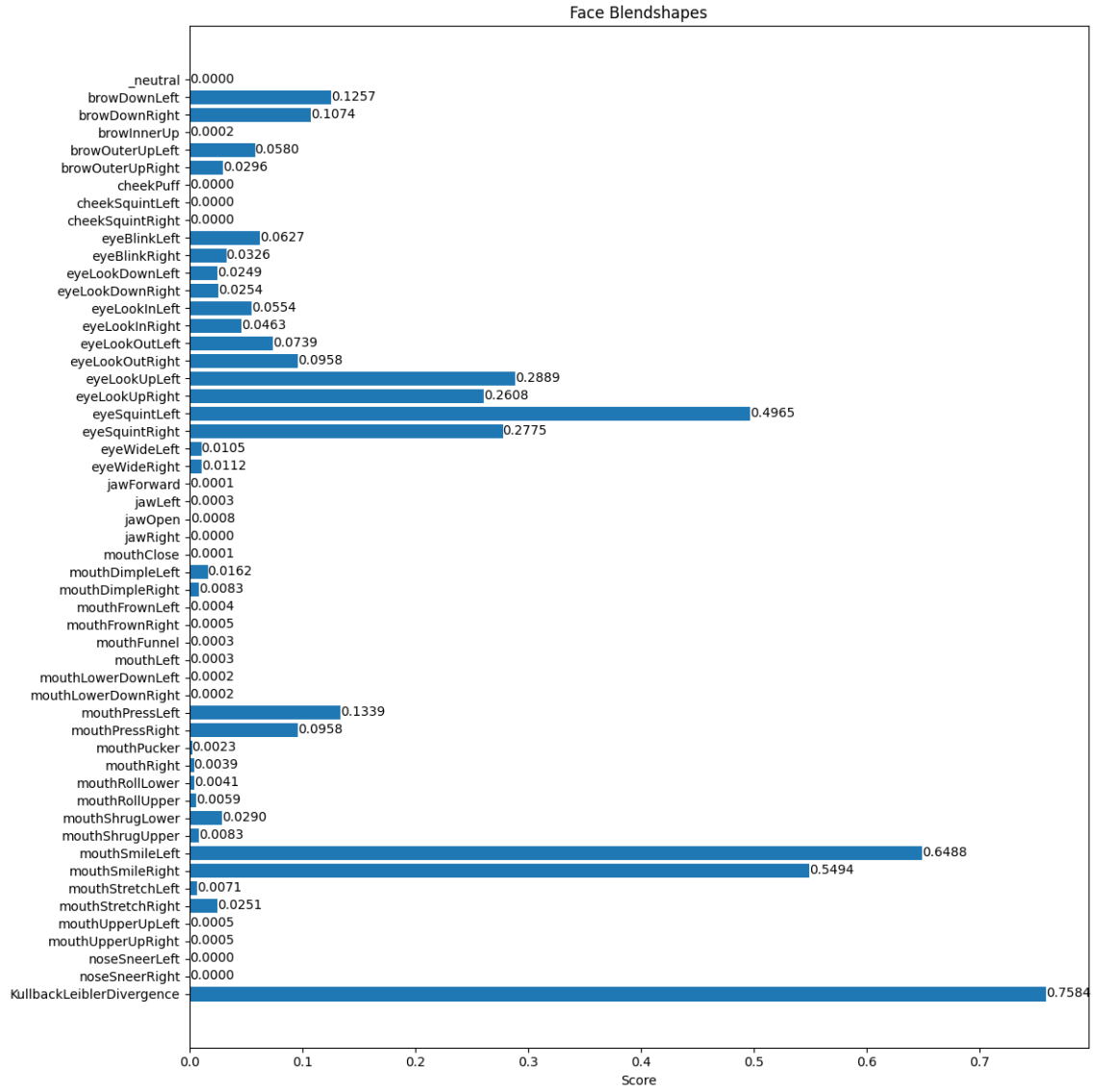


Figure 23: Results of BL(M^*)

C. Classification

Following the generation of two abstract data representation arrays,demonstrated in Fig 24, we employ two identical Fully Connected (FC) models for classification purposes. The architectures of these models are detailed in Table 3. To train and Test these models, 60% of the training data and 60 % of the test data of the dataset were utilized. Both of these models are fed with $ABS(M^*)$ as input which $ABS()$ is relevant abstraction function(OA or BA), and their outputs represent the emotion class with the highest likelihood, determined through a softmax layer. As described in Equation V..7, FC denotes the fully connected model along with its trained parameters(weights and biases) denoted by θ. The inputs to Fc consist of $ABS(M^*)$ and θ, resulting in logits as outputs. Subsequently, σ represents the softmax layer, which outputs the emotion class name.

$$\text{emotion class name} = \sigma(FC(ABS(M^*), \theta)) \tag{V..7}$$

Table 3: Architecture of FC model for classifying OP and BL abstraction datas.

Index	Layer	Size	Activation Function
1	Dense	(None, 256)	relu
2	Dropout	(None, 256)	-
3	Dense	(None, 1024)	relu
4	Dropout	(None, 1024)	-
5	Dense	(None, 512)	relu
6	Dropout	(None, 512)	-
7	Dense	(None, 128)	relu
8	Dropout	(None, 128)	-
9	Classification	(None, 7)	softmax

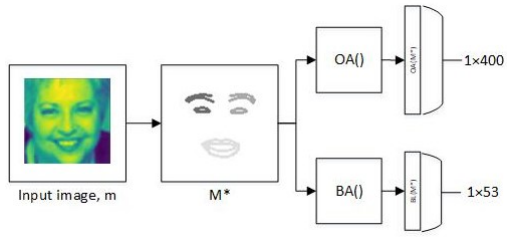


Figure 24: Architecture of Creating BA(M^*) and OA(M^*) as input for classifying in FC model

D. Fusion

In the fusion steps, as outlined in Equation V..8, the ANFIS model takes the logits from two FC models into account. Here, FO corresponds to the FC model for $OA(M^*)$, while FB pertains to the FC model for $BA(M^*)$. These outputs are then employed for the conclusive decision-making and classification of the input image, m. To train and test the ANFIS model, the remaining 30% of the dataset is utilized. This methodology is illustrated in Fig 25.

$$\text{emotion class name} = ANFIS(FO(OA(M^*), \theta^O), FB(BA(M^*), \theta^B)) \quad \text{(V..8)}$$

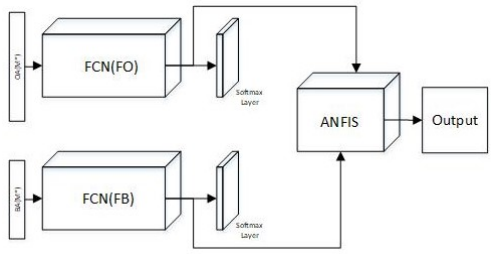


Figure 25: Architecture of ensemble's two FC models with ANFIS model for classification

Chapter VI.
Evaluation and Conclusion

A. Experimental Results of III.

We evaluated the proposed method of chapter III. by focusing on American, Indian, and Bangla sign language recognition. In each case, a comparative analysis was conducted by assessing the number of MAC operations, model size, accuracy, and power in direct comparison with state-of-the-art models. Table 4 presents the properties of our test data, along with the respective counts of images employed for training and testing the models. The ASL, ISL, and BSL datasets contains $87K$, $42k$, and $11k$ sign gesture images arranged in 28, 35, and 38 distinct classes, respectively.

Table 4: Sign language datasets utilized in our experiments.

Dataset	# Class	# Image	# Train	# Test
American Sign Language (ASL) [74]	28	87000	69600	17400
Indian Sign Language (ISL) [75]	35	42000	33600	8400
Bangla Sign Language (BSL) [76]	38	11060	8848	2212

1. Hyperparameter Tuning

Hyperparameters constitute critical configuration settings influencing both the computational load and learning dynamics of the models. Employing a thorough exploration of the hyperparameter space, our objective was to pinpoint an optimal combination that simultaneously minimizes computational requirements while ensuring model accuracy surpasses the 99% threshold. This meticulous tuning process seeks to strike an ideal balance, enhancing the efficiency and effectiveness of our computational models.

Table 5 presents a comprehensive view of hyperparameter configurations applied to our classifiers. Notably, the OA model features a single hidden layer with 128 neurons, whereas the GA model incorporates two hidden layers. This distinction arises from the varied information and features preserved by OA and GA abstractions, leading to differences in model accuracy.

To align the GA model's accuracy with OA, an additional hidden layer was introduced, despite the ensuing increase in MAC operations and memory footprint. These meticulous hyperparameter choices are instrumental in ensuring the resilience and efficacy of our classifiers.

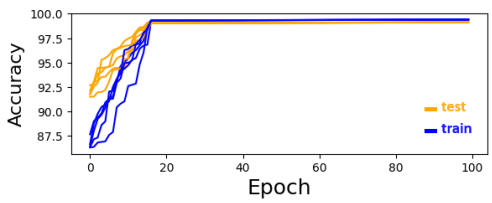
Table 5: The model parameters.

Parameter Name	**OA**	**GA**
Learning Rate	0.01	0.01
Batch Size	100	100
Number of Hidden Layers	1	2
Neurons per Hidden Layer	128	128
Loss Function	Cross Entropy	Cross Entropy
Optimization Algorithm	Adam	Adam
Activation Function	ReLU	ReLU
Number of Epochs	100	100

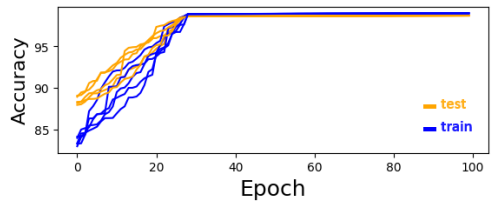
2. Cross-Validation

We employed a five-fold cross-validation process on each dataset ASL, ISL, and BSL, utilizing the proposed computational models OA and GA. Figure 26 depicts the accuracy of training and testing of each fold for 100 epochs. The figure comprises six distinct charts arranged in three rows and two columns. Each row corresponds to a dataset, and each column corresponds to a model. This information uncovers that the performance of the proposed models underscores the progressive refinement of accuracy over epochs, ultimately resulting in stable and consistent accuracy values for both training and test datasets. This observation emphasizes the robust learning capabilities of the proposed approach and its ability to generalize effectively across diverse applications and abstraction levels. Certainly, a key distinction is apparent in the convergence patterns of the OA and GA models across all datasets. The OA model exhibits a faster convergence in both training and testing phases, stabilizing before the 15th epoch, while the GA model stabilizes after the 20th epoch. Considering that the OA abstraction extracts features related to the outer boundaries of gestures, while the GA abstraction emphasizes characteristics of both the boundaries and internal cavities, we infer that, in the context of analyzing sign language

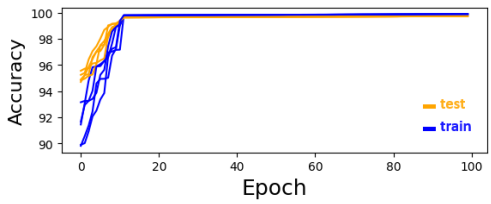
through static images, features associated with the external borders play a more decisive role. Moreover, the accuracy fluctuations across different folds within each dataset, the OA and GA models consistently demonstrate nearly identical patterns. Furthermore, in the initial epochs of the five-fold cross-validation, training accuracy tends to be lower than test accuracy. As the model approaches convergence, training accuracy surpasses test accuracy until reaching the convergence point. However, this discrepancy between test and training accuracy is not substantial enough to suggest overfitting.



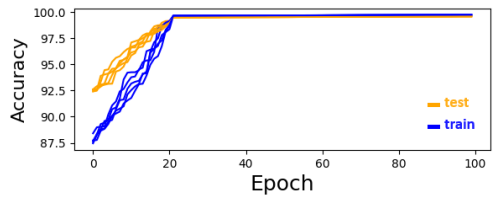
(a) ASL / OA Mean: 99.40% STD: 0.0214



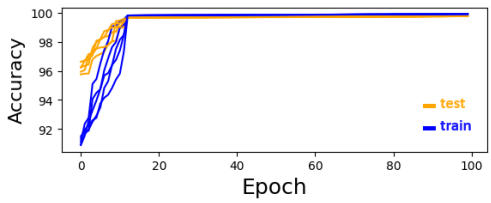
(b) ASL / GA Mean: 99% STD: 0.0048



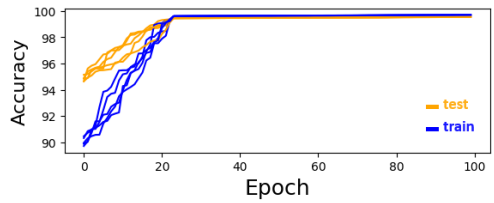
(c) ISL / OA Mean: 99.95% STD: 0.0328



(d) ISL / GA Mean: 99.71% STD: 0.0387



(e) BSL / OA Mean: 99.96% STD: 0.0428



(f) BSL / GA Mean: 99.75% STD: 0.0280

Figure 26: Training vs testing accuracy for 100 epochs during five-fold cross-validation.

Formulas VI..1 toVI..4 provide measures of precision and recall at both macro and micro levels, considering the performance across all folds in the cross-validation process. The precision and recall measurements offer complementary insights into a model's performance. Precision emphasizes the accuracy of positive predictions, while recall focuses on capturing all actual positive instances. Macro metrics deliver a per-class evaluation, assigning equal importance to each class. On the other hand, Micro metrics assess

overall performance across all classes, giving more weight to larger classes.

$$\text{Macro Precision} = \frac{1}{k}\sum_{i=1}^{k}\frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i} \tag{VI..1}$$

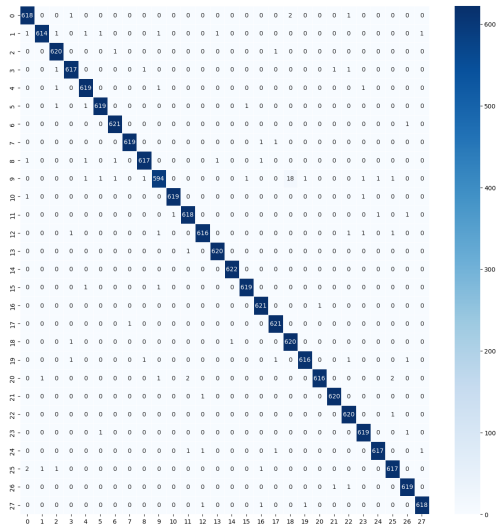
$$\text{Macro Recall} = \frac{1}{k}\sum_{i=1}^{k}\frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i} \tag{VI..2}$$

$$\text{Micro Precision} = \frac{\sum_{i=1}^{k}\text{TP}_i}{\sum_{i=1}^{k}\text{TP}_i + \sum_{i=1}^{k}\text{FP}_i} \tag{VI..3}$$

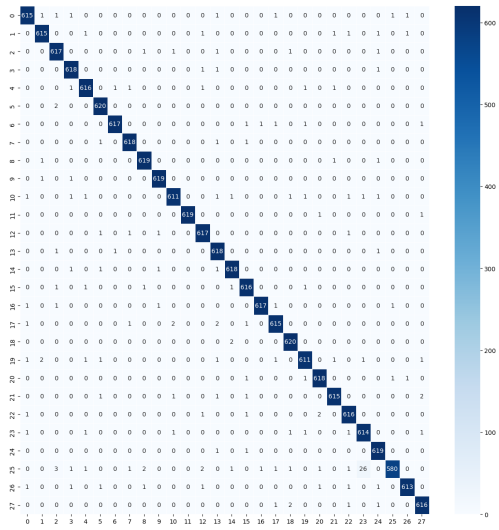
$$\text{Micro Recall} = \frac{\sum_{i=1}^{k}\text{TP}_i}{\sum_{i=1}^{k}\text{TP}_i + \sum_{i=1}^{k}\text{FN}_i} \tag{VI..4}$$

In these formulas, $k = 5$ represents the number of folds in the cross-validation, and TP_i, FP_i, and FN_i are the counts of true positives, false positives, and false negatives in the i-th fold, respectively.

Figure 27 demonstrates the confusion matrices of the OA and GA models for the ASL dataset. Notably, the OA model exhibits its notable confusion in class 9, misclassifying 27 out of 621 items. Conversely, the GA model demonstrates its most confusion in class 25, misidentifying 42 out of 622 items. This highlights that the choice of abstraction influences the level and nature of confusion, impacting the specific classes that are often misclassified. We can leverage this feature as an opportunity, allowing for the requisite flexibility to select and amalgamate various abstractions. This facilitates the creation of a harmonious equilibrium between the model's accuracy and the associated processing demands. Additionally, the information provided below each confusion matrix indicates that the overall accuracy of the OA model (99.4%) is generally higher than that of the GA model (99%).



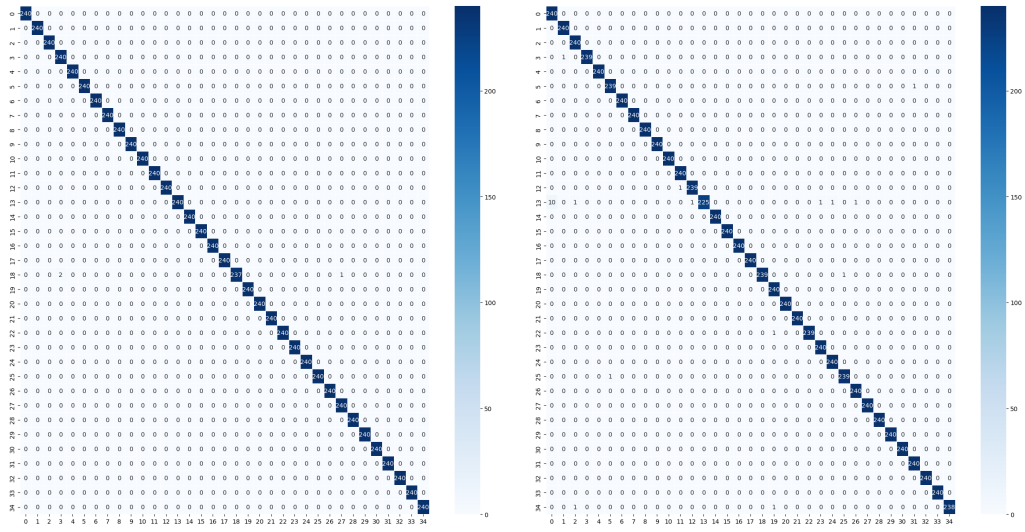
(a) ASL / OA
Accuracy: 99.4023 %
Macro Precision: 99.4052 %
Macro Recall: 99.4022 %
Micro Precision: 99.4023 %
Micro Recall: 99.4023 %



(b) ASL / GA
Accuracy: 99.0000 %
Macro Precision: 99.0073 %
Macro Recall: 98.9999 %
Micro Precision: 99.0000 %
Micro Recall: 99.0000 %

Figure 27: Confusion matrix of the ASL dataset. Numerical values in the X and Y axes mean the sequential letters as well as nothing and space.

Figure 28 presents the confusion matrices of the OA and GA models for Indian sign language detection. In this context, both the OA and ga models show high accuracy of 99.96% and 99.73%, respectively. A noteworthy observation in these two matrices is that the cases of confusion exhibit no common elements. Consequently, the potential to enhance accuracy through the amalgamation of insights from both models becomes evident.

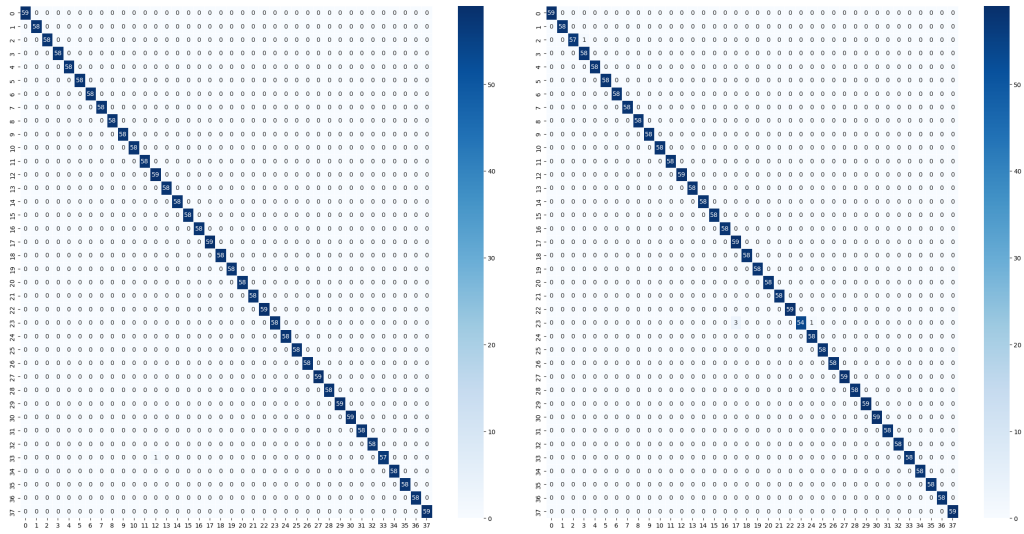


(a) ISL / OA
Accuracy: 99.9643 %
Macro Precision: 99.9643 %
Macro Recall: 99.9643 %
Micro Precision: 99.9643 %
Micro Recall: 99.9643 %

(b) ISL / GA
Accuracy: 99.7262 %
Macro Precision: 99.7316 %
Macro Recall: 99.7262 %
Micro Precision: 99.7262 %
Micro Recall: 99.7262 %

Figure 28: Confusion matrix of the ISL dataset. Numerical values in the X and Y axes mean the sequential digits and letters.

Figure 29 illustrates the confusion matrices of validating the OA and GA models on the BSL dataset. In this case, both the OA and GA models show high accuracy of 99.95% and 99.77%, respectively. Similar to the experiment on ISL, these two matrices also share the observation that cases of confusion exhibit no common elements. Once again, it can be deduced that there exists potential for accuracy enhancement through the amalgamation of insights from both models, albeit with the implication of increased computational demands.



(a) BSL / OA
Accuracy: 99.9548 %
Macro Precision: 99.9561 %
Macro Recall: 99.9546 %
Micro Precision: 99.9548 %
Micro Recall: 99.9548 %

(b) BSL / GA
Accuracy: 99.7740 %
Macro Precision: 99.7835 %
Macro Recall: 99.7731 %
Micro Precision: 99.7740 %
Micro Recall: 99.7740 %

Figure 29: Confusion matrix of the BSL dataset. Numerical values in the X and Y axes are the BSL letters. Certain gestures correspond to representing two or three letters.

3. Comparison with State-of-the-art

Table 6 offers a detailed comparative analysis that sets the performance of OA and GA models against four distinguished state-of-the-art classifiers in the domain of American Sign Language recognition. Here, the OA, GA, CNN [39], MobileNet-V2 [77] and AlexNet [78] models exhibit an accuracy rate exceeding 99%. Specifically, MobileNet-V2 achieves the highest accuracy at 99.90%, narrowly surpassing the performance of our proposed OA model by a margin of 0.5%. Certainly, CNN [39] demonstrates a slightly higher accuracy than the OA and GA models while executing less than 8 times fewer MAC operations than MobileNet-V2. However, it is imperative to recognize that CNN [39] incurs more than 200 times the number of MAC operations and boasts a model size exceeding 7.5 times that of our proposed OA model. This significant contrast underscores the favorable balance of computational efficiency and performance achieved by the OA model. In addition to better use of resources, fewer calculations and a smaller mem-

ory footprint reduce the system's power consumption. Furthermore, it is important to affirm that the count of trainable parameters in OA is notably smaller, approximately 99 times less than that of CNN [39]. This characteristic imparts a distinct advantage, allowing our model to train effectively with smaller datasets, which is a critical consideration in various practical applications.

Table 6: A comparison of the proposed models trained on American Sign Language with the state-of-the-art classifiers.

Model Name	MAC (M)	Size (MB)	TP (M)	% ACC
OA	**0.018**	**1.20**	**0.018**	99.40
GA	0.035	2.25	0.035	99.00
CNN [39]	3.740	9.12	1.781	99.80
MobileNet [79]	30.489	11.96	2.335	95.41
MobileNet-V2 [77]	31.481	18.44	5.378	**99.90**
AlexNet [78]	48.177	19.45	5.128	99.87
ResNet [78]	60.207	26.10	6.221	98.90
VGG16 [78]	69.281	36.90	13.8	98.20

Table 7 provides a comprehensive comparative analysis, facilitating the proximity of our proposed OA and GA models against five distinguished state-of-the-art classifiers specializing in Indian Sign Language recognition. Based on this data, both the CNN [80] and the proposed OA model have achieved high accuracy in tests, 100% and 99.96%, respectively. Subsequently, the FG model exhibits the next highest accuracy, recording an impressive 99.73%. Additionally, CNN [42] has a good accuracy of 99.64%, where its number of MAC operations is 26 times less than CNN [80]. It is important to highlight that the computational efficiency of the OA model is particularly notable, with the number of MAC operations being 130 times fewer than that of CNN [42], and the model size is also substantially smaller, approximately 20 times. These results underscore the impressive performance and computational efficiency balance achieved by the trained neural networks developed based on the proposed computational model.

Table 8 presents the empirical findings concerning the comparative evaluation of our proposed models against five state-of-the-art counterparts in Bangla Sign Language recognition. Based on this information, the pinnacle of accuracy, standing at an impressive 99.95%, is attributed to our proposed

Table 7: A comparison of the proposed models trained on Indian Sign Language with the state-of-the-art classifiers.

Model Name	MAC (M)	Size (MB)	TP (M)	% ACC
OA	**0.019**	**1.26**	**0.019**	99.96
GA	0.036	2.30	0.036	99.73
CNN [42]	2.517	25.78	6.145	99.64
MobileNet-V2 [77]	31.481	18.44	5.378	99.30
CNN [81]	40.978	13.98	4.077	98.70
CNN [82]	54.182	31.5	10.480	99.10
CNN [80]	65.565	9.80	0.414	**100**

OA model, closely aligned with the accuracy achieved by MobileNet-V2, which registers at 99.91%. This proximity in accuracy is particularly remarkable, given that the number of MAC operations executed by MobileNet-V2 exceeds that of OA by more than 1600 times, and its model size is also substantially larger approximately 14 times. Moreover, it is essential to highlight that the size and the count of trainable parameters in the OA model are notably smaller. These results emphasize the competitive advantage of our model, providing a convincing equilibrium between performance and computational efficiency within the realm of Bangla Sign Language recognition.

Table 8: A comparison of the proposed models trained on Bangla Sign Language with the state-of-the-art classifiers.

Model Name	MAC (M)	Size (MB)	TP (M)	% ACC
OA	**0.019**	**1.27**	**0.019**	**99.95**
GA	0.036	2.31	0.036	99.77
CNN+PEN [83]	15.492	8.21	1.603	91.52
MobileNet-V2 [84]	31.481	11.96	2.335	99.91
CNN [85]	38.579	10.48	2.045	99.60
EfficientNet-B1 [84]	39.740	25.70	6.624	98.61
ResNet [84]	60.849	26.20	6.971	99.88

In the assessment of power consumption in our work, the cProfile Python library is integrated. The script execution occurs on an AMD Ryzen 5 2600X six-core processor, with a clock frequency of 3.6 GHz and an applied voltage of 1.4 V. Figure 30 illustrates a comparative analysis of power consumption between the models developed through the proposed method and state-of-the-art models. The reported power consumption considers all operations conducted on the image, such as loading, preprocessing, and classification. The findings indicate a significant reduction in power consumption for OA

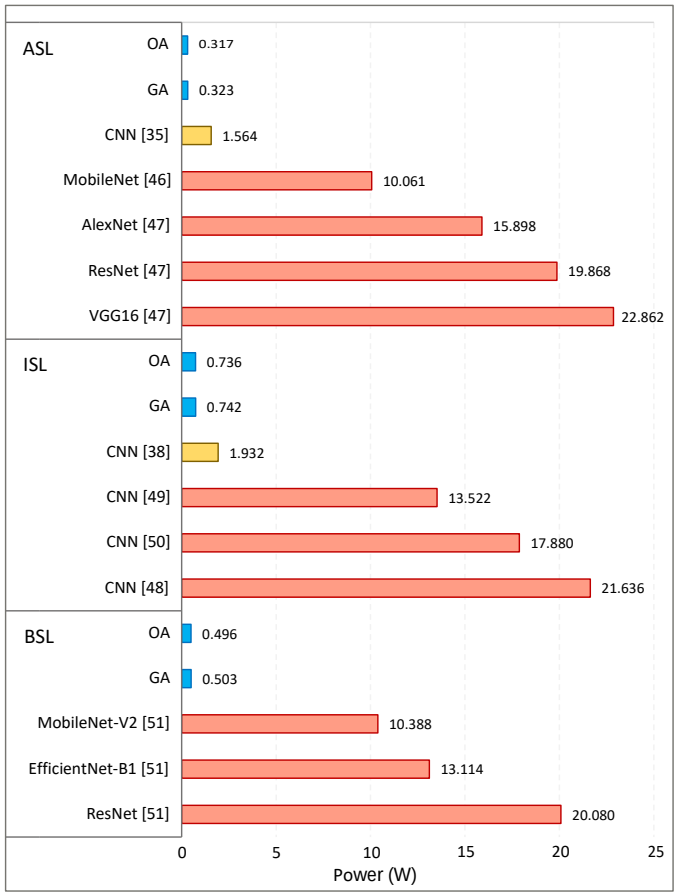


Figure 30: The power consumption of sign language recognizers.

and GA models compared to alternative models. It is imperative to acknowledge that the diminished number of MAC operations in these models contributes substantially to the observed power efficiency. However, it is essential to note that the power consumption associated with data loading and preprocessing remains non-negligible. Furthermore, it is crucial to consider the contextual dimensionality, where each ISL image is approximately four times larger than each PSL image. Consequently, despite the lower number of MAC operations in the FCN model associated with ISL, the overall power consumption of OA and GA models is comparatively higher in the context of ISL due to the increased demands stemming from data size.

In real-world applications, a comprehensive approach to image understanding, particularly in the context of sign language recognition, neces-

sitates combining multiple techniques to diminish input data size. This approach aims to enhance the efficiency and precision of the system to new levels. In this section, we explore new studies about hand gesture recognition and sign language understanding, with a particular focus on data reduction during the preprocessing stage.

Table 9 presents a comparative view between 17 hand gesture classifiers with a focus on image resizing (IR), dimension reduction (DR), image compression (IC), color quantization (QC), region-of-interest extraction (RE), superpixel segmentation (SS), filtering operation (FO), and feature extraction (FE) techniques as well as detecting sign language, number of MAC operations, model size, and the accuracy of classification.

Table 9: using data reduction techniques by sign language detection models.

No	Classifier	IR	DR	IC	CQ	RE	FO	FE	Language	MAC (10^6)	Size (MB)
1	LSTM [33]	✓	✓	✓	-	-	✓	✓	American	0.138	4.61
2	CNN [86]	✓	✓	-	✓	✓	✓	✓	American	0.517	10.51
3	CNN [42]	✓	✓	-	✓	-	✓	✓	Indian	2.517	25.78
4	CNN [39]	✓	-	-	✓	-	✓	✓	American	3.74	9.12
5	LSTM [87]	✓	✓	-	-	-	✓	✓	American	13.622	5.10
6	CNN+PEN [83]	✓	-	-	-	✓	✓	✓	Bangla	15.492	8.21
7	MobileNet [79]	✓	-	-	-	-	✓	✓	American	30.489	15.96
8	MobileNet-V2 [84]	✓	-	-	✓	-	✓	✓	Bangla	31.481	18.96
9	MobileNet-V2 [77]	✓	-	-	-	✓	✓	✓	Indian	31.481	18.44
10	MobileNet-V2 [77]	✓	-	-	-	✓	✓	✓	American	31.481	18.44
11	CNN [85]	✓	-	-	✓	-	✓	✓	Bangla	38.579	10.48
12	EfficientNet-B1 [84]	✓	-	-	✓	-	✓	✓	Bangla	39.740	25.70
13	CNN [81]	✓	-	-	✓	✓	✓	✓	Indian	40.978	13.98
14	CNN [88]	✓	-	✓	✓	✓	✓	✓	Brazilian	44.781	36.17
15	CNN [88]	✓	-	✓	✓	✓	✓	✓	Indian	44.781	36.17
16	FS-EFCNN [?]	✓	-	✓	-	✓	✓	✓	American	46.837	36.49
17	FS-EFCNN [?]	✓	-	✓	-	✓	✓	✓	British	46.837	36.49
18	FS-EFCNN [?]	✓	-	✓	-	✓	✓	✓	Greek	46.837	36.49
19	AlexNet [78]	✓	-	-	-	-	✓	✓	American	48.177	19.45
20	Transformer [89]	✓	-	-	✓	✓	✓	✓	Arabic	50.25	64.33
21	CNN [82]	✓	✓	-	✓	-	✓	✓	Indian	54.182	31.50
22	ResNet [78]	✓	-	-	-	-	✓	✓	American	60.207	26.10
23	ResNet [80]	✓	✓	-	✓	-	✓	✓	Bangla	60.849	26.20
24	CNN [80]	✓	✓	-	✓	-	✓	✓	Indian	65.565	9.80
25	VGG16 [78]	✓	-	-	-	-	✓	✓	American	69.281	36.90
26	VGG19+BiLSTM [90]	✓	✓	-	✓	✓	✓	✓	Indian	71.527	38.42
27	Inception-V1+BiLSTM [91]	✓	-	-	-	✓	✓	✓	Indian	131	93.87
28	EfficientNet [92]	✓	-	-	✓	-	✓	✓	Arabic	132	12.53
29	GLR [93]	✓	-	✓	-	✓	✓	✓	American	559	51.11
30	FSTA-Net [34]	✓	-	✓	✓	-	✓	✓	Gesture	1003	32.50
31	CNN [30]	✓	-	-	-	✓	✓	✓	Gesture	5075	108.47

Based on the information presented in 9, color quantization is one of the most popular techniques applied to hand gesture analysis, in addition to image resizing, filtering, and feature extraction. In addition, we have taken into account the dimension reduction achieved by converting grayscale images, wherein the RGB vector of each pixel is transformed into a numerical

value. This technique, executed during the preprocessing stage, aims to mitigate the processing overhead of subsequent operations and finds widespread application in the domains of sign language recognition and hand gesture recognition. It is important to note that this approach is not applied in scenarios where hand skin color analysis is employed for ROI extraction and background removal [94], or it is implemented subsequent to these operations [95]. The utilization of image compression techniques is not customary within the realm of static sign language detection. Conversely, in real-time scenarios, the compression of multiple frames into a singular frame can be implemented to enhance system performance and expedite response times [33, 34].

For hand gesture recognition, the accurate and efficient identification of the frame including the hand, commonly referred to as the ROI, is pivotal for system performance. In practical applications, two distinct methodologies are employed for ROI detection. The first approach involves dynamic detection of the hand's location, followed by the extraction of the corresponding ROI [30, 81, 86]. This typically entails the utilization of a dedicated processing model during the pre-processing stage. In contrast, the second method is more straightforward, utilizing a static frame encompassing the entire image, requiring the hand to be positioned within the frame to articulate the gesture. Consequently, the majority of the investigations within this research have adopted the latter method to obviate the need for developing an additional model for ROI extraction. Also, in the context of low-computation sign language recognition, the best models in Table 9 are LSTM [33] and CNN [86] which perform 0.138×10^6 and 0.517×10^6 MAC operations for classifying one hand gesture image, respectively. However, the accuracy of non of them is more than 94%. In this ranking, subsequent to the two best low-computation models, CNN [39] and CNN [42] both achieving an accuracy surpassing 99%, their corresponding MAC operation counts stand at 2.517×10^6 and 3.740×10^6, respectively. These two models are included for evaluation of the proposed method in the experimental results section. Notably, other models in this context exhibit elevated MAC operation counts, underscoring a predominant emphasis on optimizing model accuracy during

their development.

Availability

The test data is available on the kaggle.com website, and the corresponding links are provided in references [74–76]. Furthermore, we have implemented the proposed method using the Python programming language, and the source code is accessible via the following link:

https://github.com/cslab-chosun/Sign-language-ABS

B. Experimental Results of IV.

The ADFA-based models which defined in chapter IV. are analyzed in the same manner with previous section A. to assess their effectiveness. Specific models are then selected based on the research's target criteria for comparison with state-of-the-art approaches.

1. The Basic Models

The basic models are evaluated separately in their ability to recognize handwriting digits, alphabets, and a combination of both using MNIST, Letter, and Balanced datasets, respectively. The results are presented in Table 10, with the rows arranged in descending order based on the number of MAC operations. The information indicates that the three FCN models outperformed the SVM models, demonstrating a higher level of resource awareness and achieving superior accuracy. Also in all cases, FP shows the lowest number of MAC operations and the smallest size, and FO achieves the highest accuracy. For example, in MNIST tests, FP performs about 9000 MAC operations for recognizing a handwritten digit, and its model size is 140 KB, while FO achieves an accuracy of 96%. Also, It is important to note that the SP model features fewer trainable parameters than other models; however, the calculations and size of SVM models, as determined by Eq.II..21

and Eq.II..22, are influenced by both the attributes and the quantity of the training data.

The Letter and Balanced datasets include a greater number of classes compared to the MNIST dataset, thus introducing increased complexity to the classification task. As a result, the FCNs are equipped with additional hidden layers to improve accuracy, leading to an escalation in both the count of MAC operations and the overall model size. Moreover, the expansion in the number of output classes and the utilization of a more extensive training dataset have contributed significantly to the enlargement of SVM model dimensions.

Table 10: Performance of the basic models on EMNIST.

Dataset	Model	MAC (10^6)	Size (MB)	TP(10^6)	%ACC
MNIST	FP	**0.009**	**0.14**	0.009	82.60
	FO	0.016	0.28	0.016	**96.10**
	FG	0.016	0.28	0.016	94.80
	SP	0.840	0.98	**0.007**	69.47
	SO	1.250	1.80	0.010	93.60
	SG	1.250	1.80	0.010	91.20
Letter	FP	**0.077**	**2.53**	0.077	72.09
	FO	0.085	2.72	0.085	**86.78**
	FG	0.085	2.72	0.085	85.05
	SP	1.070	15.79	**0.013**	47.31
	SO	1.940	25.61	0.021	63.05
	SG	1.940	25.61	0.021	60.55
Balanced	FP	**0.079**	**2.36**	0.079	72.57
	FO	0.086	2.76	0.086	**87.11**
	FG	0.086	2.76	0.086	85.92
	SP	1.150	15.96	**0.014**	47.99
	SO	1.980	25.84	0.023	63.22
	SG	1.980	25.84	0.023	60.79

2. The ADFA-based Models

Table 11 reports the experimental results of all ADFA-based models with a maximum of three basic models of Table 2 in the second layer and one ANFIS model in the third layer. Here, ANFIS(FO,FP) and ANFIS(FG,FP) have preferable resource awareness, achieving accuracies of 97.49 and 95.82 respectively. However, ANFIS(FG,FO,SO) achieves the pinnacle of accuracy at 99.03. The results verify that when two or three of the basic models are combined based on the proposed ADFA, the decision fusion by ANFIS

improves the accuracy so that the obtained model is more accurate than all its basic models. This observation highlights the dual nature of abstraction application. It not only mitigates the demand for processing resources but also facilitates the concurrent utilization of a spectrum of abstractions it effectively approximates latent information present within the data.

In addition, considering the information related to the number of MAC operations and model size, it uncovers a trade-off between the accuracy and the required amount of processing and memory resources. In other words, while the inclusion of each individual basic model into an ADFA model leads to enhanced accuracy, it also results in an escalation of both MAC operations and memory requirements. For example, attaching SO to ANFIS(FO,FP) yields the ANFIS(FO,FP,SO) model. This augmentation results in an accuracy boost from 97.49% to 98.96%, with a significant increase in MAC operations by 51 times and a doubling of the model size.

We have chosen the top-performing ADFA-based models on each dataset to compare them with state-of-the-art classifiers. Table 12 provides a comprehensive overview of their statistical information across various metrics. The metrics in the table offer a comprehensive evaluation of the models' performance. Macro precision provides an average precision across all classes, without considering class imbalances, while micro precision gives more weight to classes with larger numbers of instances. Macro recall assesses the models' ability to capture all positive instances for each class, while micro recall emphasizes classes with larger numbers of instances. The F1-score combines precision and recall into a balanced measure of overall effectiveness. Accuracy measures the proportion of correctly classified instances out of the total number of instances, offering a general overview of model performance. The provided data indicates that across all performance measures on the MNIST dataset, the models consistently achieve approximately 99% score. This high level of precision, recall, F1-score, and accuracy underscores their efficacy in accurately identifying handwritten digits. Specifically, when assessing the Letter and Balanced datasets, the ANFIS(FG,FO,FP) model emerges as the top performer, achieving F1-scores of 93% and 94% respectively. These robust performance metrics underscore

Table 11: Performance of the ADFA-based models on EMNIST-MNIST.

Model	**MAC** (10^6)	**Size (MB)**	**TP**(10^6)	**%ACC**
ANFIS(FO,FP)	**0.025**	**2.07**	0.025	97.49
ANFIS(FG,FP)	**0.025**	**2.07**	0.025	95.82
ANFIS(FG,FO)	0.032	2.26	0.032	98.87
ANFIS(FG,FO,FP)	0.042	2.29	0.042	98.91
ANFIS(FP,SP)	0.850	2.77	**0.017**	84.40
ANFIS(FO,SP)	0.857	2.98	0.023	96.16
ANFIS(FG,SP)	0.857	2.98	0.023	94.87
ANFIS(FG,FP,SP)	0.866	3.94	0.032	95.91
ANFIS(FO,FP,SP)	0.866	3.94	0.032	97.54
ANFIS(FG,FO,SP)	0.874	4.07	0.039	98.89
ANFIS(FP,SO)	1.260	3.58	0.020	93.90
ANFIS(FP,SG)	1.260	3.58	0.020	92.02
ANFIS(FO,SO)	1.267	3.78	0.026	98.81
ANFIS(FO,SG)	1.267	3.78	0.026	97.29
ANFIS(FG,SO)	1.267	3.78	0.026	95.90
ANFIS(FG,SG)	1.267	3.78	0.026	95.10
ANFIS(FO,FP,SO)	1.275	4.04	0.035	98.96
ANFIS(FO,FP,SG)	1.275	4.04	0.035	98.02
ANFIS(FG,FP,SO)	1.275	4.04	0.035	96.22
ANFIS(FG,FP,SG)	1.275	4.04	0.035	96.02
ANFIS(FG,FO,SO)	1.283	4.26	0.042	**99.03**
ANFIS(FG,FO,SG)	1.283	4.26	0.042	98.94
ANFIS(SO,SP)	2.091	4.97	0.018	93.72
ANFIS(SG,SP)	2.091	4.97	0.018	91.60
ANFIS(FP,SO,SP)	2.102	5.19	0.027	93.96
ANFIS(FP,SG,SP)	2.102	5.19	0.027	92.11
ANFIS(FO,SO,SP)	2.108	5.26	0.034	98.81
ANFIS(FO,SG,SP)	2.108	5.26	0.034	97.30
ANFIS(FG,SG,SP)	2.108	5.26	0.034	95.11
ANFIS(FG,SO,SP)	2.108	5.68	0.034	95.91
ANFIS(SG,SO)	2.501	5.31	0.021	94.69
ANFIS(FP,SG,SO)	2.510	5.47	0.030	94.81
ANFIS(FG,SG,SO)	2.518	5.68	0.037	95.46
ANFIS(FO,SG,SO)	2.518	5.68	0.037	98.83
ANFIS(SG,SO,SP)	3.342	7.84	0.029	94.78

the model's capacity to effectively discern handwritten characters, encompassing both letters and digits, even among the inherent complexities of these datasets.

Table 12: Statistical information of the ADFA-based models selected for comparison with the stat-of-the-art classifiers.

Model Name	Dataset	Precision		Recall		F1-Score	Accuracy
		Macro	Micro	Macro	Micro		
ANFIS(FG,FO)	MNIST	0.9888	0.9887	0.9887	0.9887	0.99	0.9887
ANFIS(FG,FO,SO)	MNIST	0.9906	0.9903	0.9903	0.9903	0.99	0.9903
ANFIS(FG,FO)	Letter	0.9042	0.8994	0.8993	0.8995	0.90	0.8995
ANFIS(FG,FP)	Letter	0.8608	0.8591	0.8591	0.8592	0.86	0.8591
ANFIS(FO,FP)	Letter	0.8730	0.8715	0.8715	0.8715	0.87	0.8714
ANFIS(FG,FO,FP)	Letter	0.9291	0.9285	0.9283	0.9285	0.93	0.9285
ANFIS(FG,FO)	Balanced	0.9061	0.9044	0.9044	0.9044	0.90	0.9044
ANFIS(FG,FP)	Balanced	0.8724	0.8706	0.8705	0.8706	0.87	0.8706
ANFIS(FO,FP)	Balanced	0.8794	0.8788	0.8787	0.8788	0.88	0.8787
ANFIS(FG,FO,FP)	Balanced	0.9385	0.9376	0.9375	0.9376	0.94	0.9376

Table 13: Comparison of the developed ADFA-based models trained on EMNIST-MNIST with the state-of-the-art classifiers.

Model Name	MAC (10^6)	Size (MB)	TP(10^6)	%ACC
ANFIS(FG,FO)	**0.032**	**2.26**	**0.032**	98.87
LeNet300 [96]	0.124	29.59	0.328	98.73
OptConv+Log+Perc [97]	0.546	45.60	0.551	99.43
ANFIS(FG,FO,SO)	1.283	4.26	0.042	99.03
Ensemble [98]	2.867	10.51	1.128	**99.91**
EnsNet [11]	3.452	14.34	1.587	99.84
OPIUM Classifier [70]	5.034	23.82	0.264	95.90
CapsNet [99]	24.397	4.79	0.949	99.75
ResNet-9 [100]	58.177	26.35	4.088	99.68
MobileNet [101]	29.224	15.03	2.259	99.67
VGG-5 [102]	599.763	25.05	5.263	99.72

Table 13 presents the comparison between two ADFA-based models ANFIS(FG,FO) and ANFIS(FG,FO,SO), and nine state-of-the-art classifiers on the EMNIST-MNIST dataset. In this test, ANFIS (FG,FO) stands out as the optimal choice from a resource-aware perspective, accounting for 26% of MAC operations and boasting a compact size of just 8% compared to LeNet300, the leading state-of-the-art in this regard. It also achieves superior accuracy. On the other hand, the ensemble learning-based models Ensemble [98] and EnsNet [11] have the highest accuracy but, their number of MAC operations and model size are considerably more than the ADFA-based models. The number of MAC operations for Ensemble and EnsNet

is 42 and 50 times greater than that of ANFIS(FG,FO), respectively. Also, the four deep learning models CapsNet [99], VGG-5 [102], ResNet-9 [100], and MobileNet [101] have an accuracy of more than 99%, but they are not resource aware in comparison with ANFIS(FG,FO,SO) which its accuracy is more than 99% too. Furthermore, the accuracy of the four deep learning models CapsNet [99], VGG-5 [102], ResNet-9 [100], and MobileNet [101] surpasses the 99% mark. However, the models do not exhibit the same level of resource awareness as ANFIS(FG,FO,SO), achieving an accuracy exceeding 99%. Additionally, it is pertinent to observe that the ADFA-based models exhibit a substantially reduced number of trainable parameters (TP) compared to their counterparts. This attribute facilitates accelerated training procedures, mitigates the potential for overfitting by promoting improved generalization, and culminates heightened performance across validation and tests.

To evaluate the models for handwriting alphabet classification, we exclusively employed FCN models in the second layer of ADFA, as the SVM base models lacked the requisite accuracy and efficiency for this particular scenario. Table 14 illustrates a comparison between the four resultant models and eight state-of-the-art classifiers. The ADFA-based models clearly exhibit a predominance over other models with respect to the metrics of resource awareness. The ADFA model that attains the highest accuracy is ANFIS(FG,FO,FP) with an accuracy rate of 92.85%. However, the most optimal accuracy across all classifiers, reaching 95.96%, is achieved by WaveMixLite [103]. Conversely, in terms of MAC operations, ANFIS(FG,FO,FP) requires only around 6% of the MAC operations needed by WaveMixLite. Also, among the classifiers surpassing ANFIS(FG,FO,FP) in accuracy, µ2Net [104], µ2Net+ [105], TextCaps [106], and VGG-5 [102] exhibit an extensive and incomparable count of MAC operations. However, OptConv+Log+Perc [97] features only twice the number of MAC operations and a volume approximately four and a half times larger.

To assess the models' performance in recognizing both handwriting alphabets and digits in the EMNIST-Balanced dataset, we exclusively utilized FCN models in the second layer of ADFA. This decision was motivated by

Table 14: Comparison of the developed ADFA-based models trained on EMNIST-Letter with the state-of-the-art classifiers.

Model Name	MAC (10^6)	Size (MB)	TP(10^6)	%ACC
ANFIS(FO,FP)	**0.163**	**6.94**	**0.163**	87.14
ANFIS(FG,FP)	**0.163**	**6.94**	**0.163**	85.91
ANFIS(FG,FO)	0.171	7.14	0.171	89.95
ANFIS(FG,FO,FP)	0.250	10.01	0.250	92.85
OptConv+Log+Perc [97]	0.551	46.33	0.551	93.65
DWT-DCT+SVM [107]	2.415	75.14	13.456	89.51
WaveMixLite [103]	4.018	16.00	4.000	**95.96**
OPIUM Classifier [70]	5.034	23.82	0.264	85.27
µ2Net [104]	49.123	26.85	6.714	93.68
µ2Net+ [105]	51.897	26.39	6.599	95.03
TextCaps [106]	253.480	24.58	5.870	95.36
VGG-5 [102]	798.413	38.14	9.536	95.88

the limited accuracy and efficiency of SVM base models in handling this relatively high-dimensional classification scenario. The four resultant models and four state-of-the-art classifiers are compared according to the results illustrated in Table 15. The ADFA-based models demonstrate a predominance over other models concerning resource awareness metrics, and they also outperform other models in terms of accuracy. For example, ANFIS(FO,FP,FG) attains the highest accuracy at 93.76%, surpassing the state-of-the-art models in terms of resource awareness.

Table 15: Comparison of the developed ADFA-based models trained on EMNIST-Balanced with the state-of-the-art classifiers.

Model Name	MAC (10^6)	Size (MB)	TP(10^6)	%ACC
ANFIS(FO,FP)	**0.166**	**7.26**	**0.166**	87.87
ANFIS(FP,FG)	**0.166**	**7.26**	**0.166**	87.06
ANFIS(FO,FG)	0.173	7.41	0.173	90.44
ANFIS(FO,FP,FG)	0.253	10.16	0.253	**93.76**
OptConv+Log+Perc [97]	0.554	46.33	0.554	87.69
OPIUM Classifier [70]	1.034	23.82	0.264	78.02
CNN(6Conv,2Dense) [108]	131.310	115	1.360	90.59
TextCaps [106]	253.480	210	5.870	90.46

The results verify the potential of ADFA-based models in simplifying data through diverse abstraction perspectives for processing reduction, as well as the capacity to opt for lightweight models for decision fusion. This combination facilitates the creation of tools possessing diverse decision-making capabilities and resource demands, rendering each tool accordingly tailored to distinct applications.

3. Availability

The Python source code of the developed abstractions and ADFA-based models is available at:

https://github.com/cslab-chosun/ADFA

C. Experimental Results of V.

In this section we evalute the proposed method which defiend in chapter V. and compare the final fusion model with state-of-the-art model.

1. The FC Models

The FC models are evaluated separately in their ability to recognize emotion state. The results are presented in Table 16, with the rows arranged in descending order based on the number of MAC operations. The information indicates that FB shows the lowest number of MAC operations and the smallest size, and FO achieves the highest accuracy.Furthermore, fig 31 illustrates the ROC curve corresponding to the FO model, showcasing its discriminative performance across various threshold values. The AUC score for this model is reported to be 0.88, indicating its ability to accurately distinguish between the positive and negative classes. A higher AUC score suggests better model performance in terms of classification accuracy and robustness.

Table 16: Performance of the basic models on Emotion.

Dataset	Model	MAC (10^6)	Size (MB)	TP(10^6)	%ACC
Emotion	FO	0.918	10.6	1.217	86.34
	FB	0.891	9.84	1.217	81.69

2. The ANFIS-based Model

Table 17 presents the experimental outcomes of the ANFIS-based model integrated with two fundamental models outlined in Table 16. Specifically,

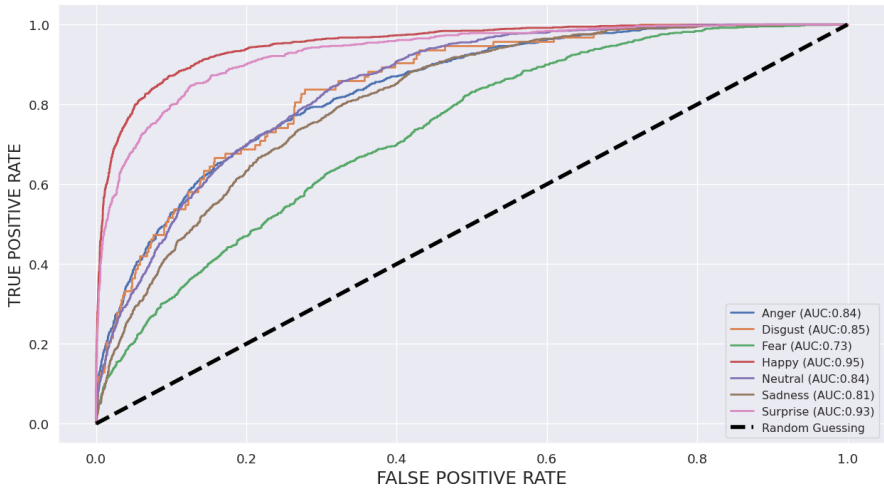


Figure 31: ROC curve corresponding to the FO model.

ANFIS(FO,FB) demonstrates accuracies of 97.49% respectively. These findings corroborate that the amalgamation of two basic models, as per the proposed approach outlined in Chapter V., results in improved accuracy through decision fusion by ANFIS. Consequently, the combined model surpasses the individual accuracies of its constituent basic models. This observation underscores the dual functionality of abstraction application, which not only reduces the computational load but also facilitates the simultaneous exploitation of diverse abstractions to effectively capture latent information within the dataset. Furthermore, an examination of the data pertaining to the number of MAC operations and model size reveals a trade-off between accuracy and the necessary processing and memory resources. Essentially, the integration of each basic model into an ANFIS model contributes to improved accuracy but concurrently leads to an increase in both MAC operations and memory demands. Figure 32 illustrates the confusion matrix corresponding to the ANFIS(FO,FB) model. Utilizing Equations VI..1, VI..2, VI..3, and VI..4, along with the information derived from the confusion matrix, the Macro Precision for the ANFIS(FO,FB) model is 89.85%, the Macro Recall is 89.77%, the Micro Precision is 89.7%, and the Micro Recall is 89.77%.

Table 18 presents the comparison between ANFIS-based model, AN-

Table 17: Performance of the ANFIS-based model.

Model	MAC (10^6)	Size (MB)	TP(10^6)	%ACC
ANFIS(FO,FB)	1.809	26.39	2.434	89.77

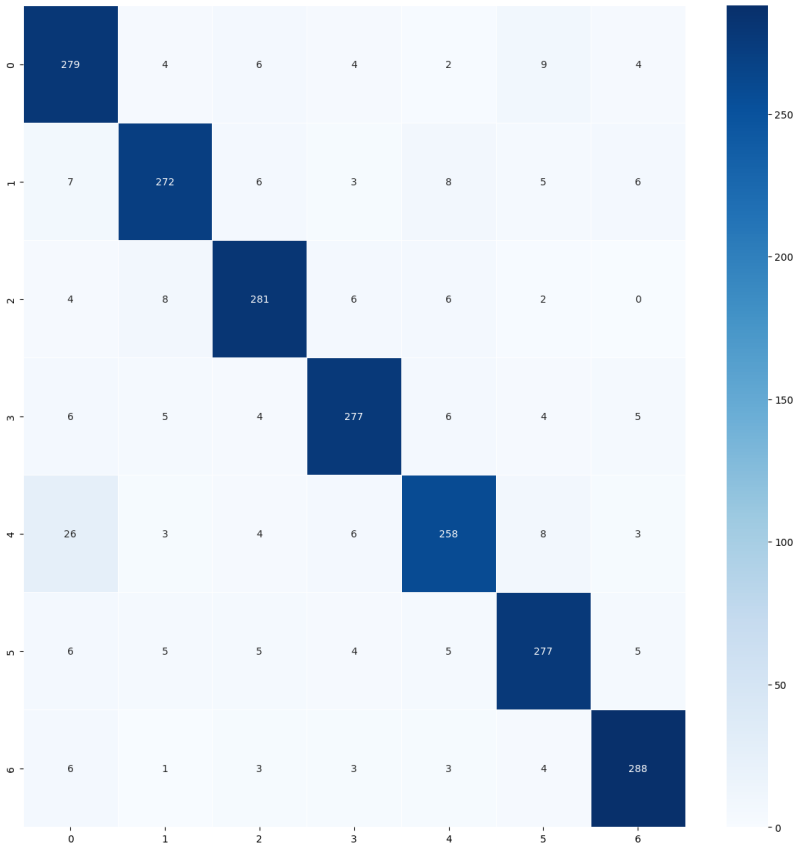


Figure 32: Confusion matrix for ANFIS(FO,FB) model

Table 18: Comparison of the developed ANFIS-based model with the state-of-the-art classifiers.

Model Name	MAC (10^6)	Size (MB)	TP(10^6)	%ACC
ANFIS(FO,FB)	1.809	26.39	2.434	89.77
CNN [109]	40.971	13.91	4.474	65.22
Fully Connected [110]	0.614	2.13	0.614	53
CAGRNN [111]	55.923	50.16	16.12	94.74
Densenet [112]	50.119	43.12	13.86	89.41

FIS(FO,FB), and four state-of-the-art classifiers on the Emotion dataset.

In this experiment, the CAGRNN model [111] achieved the highest accuracy, surpassing our proposed ANFIS-based model by 5%. However, it's worth noting that the CAGRNN model exhibits significantly higher numbers of MAC operations and model size compared to the ANFIS-based model. Specifically, the CAGRNN model's MAC operations are approximately 31 times greater than those of ANFIS(FO,FB), representing a substantial difference in computational complexity. Also, the Densenet model [112] has close accuracu to the ANFIS(FO,FB) model, but it is not resource aware in comparison with ANFIS(FO,FB) which its accuracy is more than 89% too.

D. Conclusion and Future Work

In the era of data-driven decision-making, machine learning models have become widespread in various applications. The performance of these models heavily relies on computational resources. Therefore, resource-aware intelligent computing, finding a proper answer for the trade-off between the restrictions of computational resources and the inference power of the systems, has become a crucial issue in the field of artificial intelligence. In this thesis, to address this challenge, we proposed a data work load reduction method named *Abstraction*. We also used ANFIS as a powerful tool for decision fusion and combining the results of various models. Taking advantage of this architecture enables the reduction of computational resources along with maintaining high inference power. The experiments conducted on the Sign language, EMNIST, and Emotion datasets validate the effectiveness of our proposed architecture, achieving accuracies either surpassing or closely approaching those of state-of-the-art models. Notably, our model exhibits a compact size and reduced number of MAC operations, outperforming existing state-of-the-art counterparts. These findings underscore the practical feasibility of our architecture, given its reasonable computational resource demands alongside high accuracy. Consequently, our architecture emerges as a promising candidate for deployment on edge devices, thereby constituting a focal point for our future research endeavors.

Bibliography

[1] S. Zhu, T. Yu, T. Xu, H. Chen, S. Dustdar, S. Gigan, D. Gunduz, E. Hossain, Y. Jin, F. Lin *et al.*, "Intelligent computing: The latest advances, challenges, and future," *Intelligent Computing*, vol. 2, pp. 1–45, 2023.

[2] W. Doghri, A. Saddoud, and L. Chaari Fourati, "Cyber-physical systems for structural health monitoring: sensing technologies and intelligent computing," *The Journal of Supercomputing*, vol. 78, no. 1, pp. 766–809, 2022.

[3] A. H. Bukhari, M. A. Z. Raja, N. Rafiq, M. Shoaib, A. K. Kiani, and C.-M. Shu, "Design of intelligent computing networks for nonlinear chaotic fractional rossler system," *Chaos, Solitons & Fractals*, vol. 157, p. 111985, 2022.

[4] S. Mao, L. Liu, N. Zhang, M. Dong, J. Zhao, J. Wu, and V. C. Leung, "Reconfigurable intelligent surface-assisted secure mobile edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 6, pp. 6647–6660, 2022.

[5] Z. Tong, F. Ye, M. Yan, H. Liu, and S. Basodi, "A survey on algorithms for intelligent computing and smart city applications," *Big Data Mining and Analytics*, vol. 4, no. 3, pp. 155–172, 2021.

[6] M. Treviso, J.-U. Lee, T. Ji, B. v. Aken, Q. Cao, M. R. Ciosici, M. Hassid, K. Heafield, S. Hooker, C. Raffel, P. H. Martins, A. F. T. Martins, J. Z. Forde, P. Milder, E. Simpson, N. Slonim, J. Dodge, E. Strubell, N. Balasubramanian, L. Derczynski, I. Gurevych, and R. Schwartz, "Efficient methods for natural language processing: A survey," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 826–860, 2023.

[7] S. J. Russell, *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[8] R. Shaw, E. Howley, and E. Barrett, "An intelligent ensemble learning approach for energy efficient and interference aware dynamic virtual machine consolidation," *Simulation Modelling Practice and Theory*, vol. 102, p. 101992, 2020.

[9] Y. Yang, H. Lv, and N. Chen, "A survey on ensemble learning under the era of deep learning," *Artificial Intelligence Review*, pp. 1–45, 2022.

[10] Y. Guo, X. Wang, P. Xiao, and X. Xu, "An ensemble learning framework for convolutional neural network based on multiple classifiers," *Soft Computing*, vol. 24, pp. 3727–3735, 2020.

[11] D. Hirata and N. Takahashi, "Ensemble learning in cnn augmented with fully connected subnetworks," *arXiv preprint arXiv:2003.08562*, 2020.

[12] I. Lahmar, A. Zaier, M. Yahia, and R. Boaullegue, "A multiple fuzzy c-means ensemble cluster forest for big data," in *Hybrid Intelligent Systems: 21st International Conference on Hybrid Intelligent Systems (HIS 2021), December 14–16, 2021*. Springer, 2022, pp. 442–451.

[13] H. Tran-Dang and D.-S. Kim, "Fog resource aware framework for task offloading in iot systems," in *Cooperative and Distributed Intelligent Computation in Fog Computing: Concepts, Architectures, and Frameworks*. Springer, 2023, pp. 47–82.

[14] Y. Yang, S. Mandt, L. Theis *et al.*, "An introduction to neural data compression," *Foundations and Trends® in Computer Graphics and Vision*, vol. 15, no. 2, pp. 113–200, 2023.

[15] P. Garcia, "Effectiveness of multi-abstraction computing tools on promoting exploratory self-learning in engineering: a case study using a custom real-time operating system for remote learning," in *2021 IEEE 8th International Conference on e-Learning in Industrial Electronics (ICELIE)*. IEEE, 2021, pp. 1–6.

[16] M. Ahmed, "Data summarization: a survey," *Knowledge and Information Systems*, vol. 58, no. 2, pp. 249–273, 2019.

[17] S. Kim, B. Jang, J. Lee, H. Bae, H. Jang, and I.-C. Park, "A cnn inference accelerator on fpga with compression and layer-chaining techniques for style transfer applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2023.

[18] S. Cong and Y. Zhou, "A review of convolutional neural network architectures and their optimizations," *Artificial Intelligence Review*, vol. 56, no. 3, pp. 1905–1969, 2023.

[19] M. Shaeri and A. M. Sodagar, "Data transformation in the processing of neuronal signals: A powerful tool to illuminate informative contents," *IEEE Reviews in Biomedical Engineering*, 2022.

[20] J. Chen, H. Huang, J. Peng, J. Zhu, L. Chen, C. Tao, and H. Li, "Contextual information-preserved architecture learning for remote-sensing scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–14, 2021.

[21] R. Chakwate, A. Subramaniam, and A. Mittal, "Marnet: Multi-abstraction refinement network for 3d point cloud analysis," *arXiv preprint arXiv:2011.00923*, 2020.

[22] N. J. Mitra and M. Pauly, "Shadow art," *ACM Transactions on Graphics*, vol. 28, no. CONF, pp. 156–1, 2009.

[23] K. Tang, Y. Ma, D. Miao, P. Song, Z. Gu, Z. Tian, and W. Wang, "Decision fusion networks for image classification," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[24] C. Yin, S. Zhang, and Q. Zeng, "Hybrid representation and decision fusion towards visual-textual sentiment," *ACM Transactions on Intelligent Systems and Technology*, 2023.

[25] S. H. Shahrokhi, M. Hosseinzadeh, M. Reshadi, and S. Gorgin, "A novel high-speed and low-pdp approximate full adder cell for image blending," *Mathematics*, vol. 11, no. 12, p. 2649, 2023.

[26] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," in *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, 2023, pp. 609–617.

[27] M. I. U. Haque, A. K. Dubey, I. Danciu, A. C. Justice, O. S. Ovchinnikova, and J. D. Hinkle, "Effect of image resolution on automated classification of chest x-rays," *Journal of Medical Imaging*, vol. 10, no. 4, pp. 044 503–044 503, 2023.

[28] S. Ayesha, M. K. Hanif, and R. Talib, "Overview and comparative study of dimensionality reduction techniques for high dimensional data," *Information Fusion*, vol. 59, pp. 44–58, 2020.

[29] B. Sun, Y. Zhang, S. Jiang, and Y. Fu, "Hybrid pixel-unshuffled network for lightweight image super-resolution," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 2, 2023, pp. 2375–2383.

[30] H. Liang, L. Fei, S. Zhao, J. Wen, S. Teng, and Y. Xu, "Mask-guided multiscale feature aggregation network for hand gesture recognition," *Pattern Recognition*, vol. 145, p. 109901, 2024.

[31] F.-H. Tseng, K.-H. Yeh, F.-Y. Kao, and C.-Y. Chen, "Mininet: Dense squeeze with depthwise separable convolutions for image classification in resource-constrained autonomous systems," *ISA transactions*, vol. 132, pp. 120–130, 2023.

[32] D. Mishra, S. K. Singh, and R. K. Singh, "Deep architectures for image compression: a critical review," *Signal Processing*, vol. 191, p. 108346, 2022.

[33] M. Lee and J. Bae, "Real-time gesture recognition in the view of repeating characteristics of sign languages," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 12, pp. 8818–8828, 2022.

[34] W. Song, W. Kang, and L. Lin, "Hand gesture authentication by discovering fine-grained spatiotemporal identity characteristics," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[35] S. Jamil, M. J. Piran, M. Rahman, and O.-J. Kwon, "Learning-driven lossy image compression: A comprehensive survey," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106361, 2023.

[36] M. E. Celebi, "Forty years of color quantization: a modern, algorithmic survey," *Artificial Intelligence Review*, pp. 1–82, 2023.

[37] A. Aliouat, N. Kouadria, M. Maimour, S. Harize, and N. Doghmane, "Region-of-interest based video coding strategy for rate/energy-constrained smart surveillance systems using wmsns," *Ad Hoc Networks*, vol. 140, p. 103076, 2023.

[38] P. Zhou, X. Kang, and A. Ming, "Vine spread for superpixel segmentation," *IEEE Transactions on Image Processing*, vol. 32, pp. 878–891, 2023.

[39] M. M. Damaneh, F. Mohanna, and P. Jafari, "Static hand gesture recognition in sign language based on convolutional neural network with feature extraction method using orb descriptor and gabor filter," *Expert Systems with Applications*, vol. 211, p. 118559, 2023.

[40] C. Yang, C. Zhang, H. Shen, T. Peng, C. Wang, L. Deng, H. Chen, and L. He, "Hfan: High-frequency attention network for hyperspectral image denoising," *International Journal of Machine Learning and Cybernetics*, pp. 1–15, 2023.

[41] V. K. Vishnoi, K. Kumar, and B. Kumar, "A comprehensive study of feature extraction techniques for plant leaf disease detection," *Multimedia Tools and Applications*, pp. 1–53, 2022.

[42] S. Katoch, V. Singh, and U. S. Tiwary, "Indian sign language recognition system using surf with svm and cnn," *Array*, vol. 14, p. 100141, 2022.

[43] T. Petković, L. Petrović, I. Marković, and I. Petrović, "Human action prediction in collaborative environments based on shared-weight lstms with feature dimensionality reduction," *Applied Soft Computing*, vol. 126, p. 109245, 2022.

[44] S. M. Qaisar, A. López, D. Dallet, and F. J. Ferrero, "semg signal based hand gesture recognition by using selective subbands coefficients and machine learning," in *2022 IEEE International Instrumentation and Measurement Technology Conference* (*I2MTC*). IEEE, 2022, pp. 1–6.

[45] J. Samkunta, P. Ketthong, K. Hashikura, M. A. S. Kamal, I. Murakami, and K. Yamada, "Feature reduction for hand gesture classification: Sparse coding approach," in *2023 20th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology* (*ECTI-CON*). IEEE, 2023, pp. 1–4.

[46] T. M. Mitchell and M. Learning, "The mcgraw-hill companies," *Inc., New York*, 1997.

[47] A. Krizhevsky, "Advances in neural information processing systems," (*No Title*), p. 1097, 2012.

[48] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[49] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[50] A. Van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," *Advances in neural information processing systems*, vol. 26, 2013.

[51] C. Bishop, "Pattern recognition and machine learning," *Springer google schola*, vol. 2, pp. 5–43, 2006.

[52] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[53] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[54] J. Friedman, T. Hastie, and R. Tibshirani, "The elements of statistical learning. vol. 1 springer series in statistics," *New York*, 2001.

[55] R. Sutton and A. Barto, "Reinforcement learning: An introduction. mit press; 2018," *Google Scholar*, pp. 329–331.

[56] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks," in *Artificial intelligence and statistics*. PMLR, 2015, pp. 192–204.

[57] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[58] M. Li, T. Zhang, Y. Chen, and A. J. Smola, "Efficient mini-batch training for stochastic optimization," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 661–670.

[59] S. Herculano-Houzel, "The human brain in numbers: a linearly scaled-up primate brain," *Frontiers in human neuroscience*, p. 31, 2009.

[60] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," *Advances in neural information processing systems*, vol. 28, 2015.

[61] V. G. Maltarollo, K. M. Honório, and A. B. F. da Silva, "Applications of artificial neural networks in chemical problems," *Artificial neural networks-architectures and applications*, pp. 203–223, 2013.

[62] A. Krenker, J. Bešter, and A. Kos, "Introduction to the artificial neural networks," *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech*, pp. 1–18, 2011.

[63] K. Roushangar, R. Ghasempour, and S. Shahnazi, "Kernel-based modeling," in *Handbook of Hydroinformatics*. Elsevier, 2023, pp. 267–281.

[64] Y. Chai, L. Jia, and Z. Zhang, "Mamdani model based adaptive neural fuzzy inference system and its application," *International Journal of Computer and Information Engineering*, vol. 3, no. 3, pp. 663–670, 2009.

[65] A. Lohani, N. K. Goel, and K. Bhatia, "Takagi–sugeno fuzzy inference system for modeling stage–discharge relationship," *Journal of Hydrology*, vol. 331, no. 1-2, pp. 146–160, 2006.

[66] A. Al-Hmouz, J. Shen, R. Al-Hmouz, and J. Yan, "Modeling and simulation of an adaptive neuro-fuzzy inference system (anfis) for mobile learning," *IEEE Transactions on Learning Technologies*, vol. 5, no. 3, pp. 226–237, 2011.

[67] M. Fallah, M. Najafi, S. Gorgin, and J.-A. Lee, *An ultra-low-computation model for understanding sign languages*, 2023.

[68] H. B. Frank Jacobs, *Hand Shadow Fun*. Dover Publications, 2015.

[69] M. Fallah, M. Najafi, S. Gorgin, and J.-A. Lee, *Abstraction and decision fusion architecture for resource-aware image understanding with application on handwriting character classification*, 2023.

[70] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *2017 international joint conference on neural networks* (*IJCNN*). IEEE, 2017, pp. 2921–2926.

[71] Ares, "Emotion detection," Dec 2020. [Online]. Available: https://www.kaggle.com/datasets/ananthu017/emotion-detection-fer

[72] *The OpenCV Reference Manual*, 2nd ed., Itseez, April 2014.

[73] G. Google, "Google/mediapipe: Cross-platform, customizable ml solutions for live and streaming media." [Online]. Available: https://github.com/google/mediapipe/tree/master

[74] Akash, "Asl alphabet," Apr 2018. [Online]. Available: https://www.kaggle.com/dsv/29550

[75] "Indian Sign Language Dataset — kaggle.com," https://www.kaggle.com/datasets/vaishnaviasonawane/indian-sign-language-dataset, [Accessed 15-11-2023].

[76] "Bengali Sign Language Dataset — kaggle.com," https://www.kaggle.com/datasets/muntakimrafi/bengali-sign-language-dataset, [Accessed 15-11-2023].

[77] S. Sharma and S. Singh, "Isl recognition system using integrated mobile-net and transfer learning method," *Expert Systems with Applications*, vol. 221, p. 119772, 2023.

[78] V. Lakshmi, K. Sivachandra, H. Parthasaradhi, S. Abhishek, and T. Anjali, "An empirical analysis of cnn for american sign language recognition," in *2023 5th International Conference on Inventive Research in Computing Applications* (*ICIRCA*). IEEE, 2023, pp. 421–428.

[79] T. N. Abu-Jamie and S. S. Abu-Naser, "Classification of sign-language using mobilenet-deep learning," 2022.

[80] R. Patil, V. Patil, A. Bahuguna, and G. Datkhile, "Indian sign language recognition using convolutional neural network," in *ITM web of conferences*, vol. 40. EDP Sciences, 2021, p. 03004.

[81] A. Wadhawan and P. Kumar, "Deep learning-based sign language recognition system for static signs," *Neural computing and applications*, vol. 32, pp. 7957–7968, 2020.

[82] R. Dhiman, G. Joshi, and C. R. Krishna, "A deep learning approach for indian sign language gestures classification with different backgrounds," in *Journal of Physics: Conference Series*, vol. 1950, no. 1. IOP Publishing, 2021, p. 012020.

[83] T. Abedin, K. S. Prottoy, A. Moshruba, and S. B. Hakim, "Bangla sign language recognition using a concatenated bdsl network," in *Computer Vision and Image Analysis for Industry 4.0*. Chapman and Hall/CRC, 2023, pp. 76–86.

[84] K. K. Podder, M. E. Chowdhury, A. M. Tahir, Z. B. Mahbub, A. Khandakar, M. S. Hossain, and M. A. Kadir, "Bangla sign language (bdsl) alphabets and numerals classification using a deep learning model," *Sensors*, vol. 22, no. 2, p. 574, 2022.

[85] A. S. M. Miah, J. Shin, M. A. M. Hasan, and M. A. Rahim, "Bensignnet: Bengali sign language alphabet recognition using concatenated segmentation and convolutional neural network," *Applied Sciences*, vol. 12, no. 8, p. 3933, 2022.

[86] D. Bala, B. Sarkar, M. I. Abdullah, and M. A. Hossain, "American sign language alphabets recognition using convolutional neural network," *International Journal of Knowledge Based Computer Systems*, vol. 9, no. 1, 2021.

[87] C. K. Lee, K. K. Ng, C.-H. Chen, H. C. Lau, S. Chung, and T. Tsoi, "American sign language recognition and training method with recurrent neural network," *Expert Systems with Applications*, vol. 167, p. 114403, 2021.

[88] G. Z. De Castro, R. R. Guerra, and F. G. Guimarães, "Automatic translation of sign language with multi-stream 3d cnn and generation

of artificial depth maps," *Expert Systems with Applications*, vol. 215, p. 119394, 2023.

[89] S. Alyami, H. Luqman, and M. Hammoudeh, "Isolated arabic sign language recognition using a transformer-based model and landmark keypoints," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 23, no. 1, pp. 1–19, 2024.

[90] S. Das, S. K. Biswas, and B. Purkayastha, "Automated indian sign language recognition system by fusing deep and handcrafted feature," *Multimedia Tools and Applications*, vol. 82, no. 11, pp. 16 905–16 927, 2023.

[91] A. Venugopalan and R. Reghunadhan, "Applying deep neural networks for the automatic recognition of sign language words: A communication aid to deaf agriculturists," *Expert Systems with Applications*, vol. 185, p. 115601, 2021.

[92] B. Y. AlKhuraym, M. M. B. Ismail, and O. Bchir, "Arabic sign language recognition using lightweight cnn-based architecture," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 4, 2022.

[93] Z. Guo, Y. Hou, and W. Li, "Sign language recognition via dimensional global–local shift and cross-scale aggregation," *Neural Computing and Applications*, vol. 35, no. 17, pp. 12 481–12 493, 2023.

[94] J. Espejel-Cabrera, J. Cervantes, F. García-Lamont, J. S. R. Castilla, and L. D. Jalili, "Mexican sign language segmentation using color based neuronal networks to detect the individual skin color," *Expert Systems with Applications*, vol. 183, p. 115295, 2021.

[95] J. J. Raval and R. Gajjar, "Real-time sign language recognition using computer vision," in *2021 3rd International Conference on Signal Processing and Communication* (*ICPSC*). IEEE, 2021, pp. 542–546.

[96] T. Dettmers and L. Zettlemoyer, "Sparse networks from scratch: Faster training without losing performance," *arXiv preprint arXiv:1907.04840*, 2019.

[97] P. Pad, S. Narduzzi, C. Kundig, E. Turetken, S. A. Bigdeli, and L. A. Dunbar, "Efficient neural vision systems based on convolutional image acquisition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 285–12 294.

[98] S. An, M. J. Lee, S. Park, H. S. Yang, and J. So, "An ensemble of simple convolutional neural network models for mnist digit recognition," *ArXiv*, vol. abs/2008.10400, 2020.

[99] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *Advances in neural information processing systems*, vol. 30, 2017.

[100] P. Gavrikov and J. Keuper, "Cnn filter db: An empirical investigation of trained convolutional filters," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19 066–19 076.

[101] J. Sun, A. P. Fard, and M. H. Mahoor, "Xnodr and xnidr: Two accurate and fast fully connected layers for convolutional neural networks," *arXiv preprint arXiv:2111.10854*, 2021.

[102] H. D. Kabir, M. Abdar, A. Khosravi, S. M. J. Jalali, A. F. Atiya, S. Nahavandi, and D. Srinivasan, "Spinalnet: Deep neural network with gradual input," *IEEE Transactions on Artificial Intelligence*, 2022.

[103] P. Jeevan, K. Viswanathan, and A. Sethi, "Wavemix-lite: A resource-efficient neural network for image analysis," *arXiv preprint arXiv:2205.14375*, 2022.

[104] A. Gesmundo, "A continual development methodology for large-scale multitask dynamic ml systems," *arXiv preprint arXiv:2209.07326*, 2022.

[105] A. Gesmundo and J. Dean, "An evolutionary approach to dynamic introduction of tasks in large-scale multitask learning systems, 2022," *URL https://arxiv. org/abs/2205*, vol. 12755.

[106] V. Jayasundara, S. Jayasekara, H. Jayasekara, J. Rajasegaran, S. Seneviratne, and R. Rodrigo, "Textcaps: Handwritten character recognition with very small datasets," in *2019 IEEE winter conference on applications of computer vision* (*WACV*). IEEE, 2019, pp. 254–262.

[107] P. Ghadekar, S. Ingole, and D. Sonone, "Handwritten digit and letter recognition using hybrid dwt-dct with knn and svm classifier," in *2018 Fourth International Conference on Computing Communication Control and Automation* (*ICCUBEA*). IEEE, 2018, pp. 1–6.

[108] A. Shawon, M. Jamil-Ur Rahman, F. Mahmud, and M. Arefin Zaman, "Bangla handwritten digit recognition using deep cnn for large and unbiased dataset," in *2018 International Conference on Bangla Speech and Language Processing* (*ICBSLP*), 2018, pp. 1–6.

[109] B. G. K. Reddy, P. Yashwanthsaai, A. R. Raja, A. Jagarlamudi, N. Leeladhar, and T. T. Kumar, "Emotion recognition based on convolutional neural network (cnn)," in *2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation* (*ICAECA*). IEEE, 2021, pp. 1–5.

[110] P. K. Sidhu, A. Kapoor, Y. Solanki, P. Singh, and D. Sehgal, "Deep learning based emotion detection in an online class," in *2022 IEEE Delhi Section Conference* (*DELCON*). IEEE, 2022, pp. 1–6.

[111] G. Ambika and Y. Suresh, "Optimal deep convolutional neural network based face detection and emotion recognition model," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 11, no. 3, pp. 841–849, 2023.

[112] odins0n, "emotion detection," Jan 2022. [Online]. Available: https://www.kaggle.com/code/odins0n/emotion-detection