People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

Akli Mohand Oulhadj University of Bouira

Faculty of sciences and Applied Sciences

Department of computer sciences

Master's thesis

in Computer Science

Specialty : Computer Systems Engineering

Theme

Deep learning for agricultural image classification

---

supervised by

- WAHIBA ALIOUAT

Directed by

- BOUCHIBA KENZA
- BADIS LAMIS

2022/2023

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Akli Mohand Oulhadj de Bouira

Faculté des Sciences et des Sciences Appliquées

Département d'Informatique

Mémoire de Master

en Informatique

Spécialité: *Génie système d'informatique*

Thème

L'apprentissage profond pour la classification d'images Agricoles

---

Encadré par

- WAHIBA ALIOUAT

Réalisé par

- BOUCHIBA KENZA
- BADIS LAMIS

2022/2023

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Akli Mohand Oulhadj - Bouira
Tasdawit Akli Muhend Ulhağ - Tubirett

وزارة التعليم العالي و البحث العلمي
جامعة أكلي محند أولحاج - البويرة

التصريح الشرفي الخاص بالالتزام بقواعد النزاهة العلمية لإنجاز بحث

انا الممضي اسفله،

السيد(ة) Bouchiba Kenza الصفة: طالب (ماستر / دكتوراه)

الحامل(ة) لبطاقة التعريف الوطنية: 404072262 والصادرة بتاريخ 2023/12/25

المسجل(ة) بكلية / معهد Des Sciences et Sciences Appliquées قسم informatique

تخصص: Génie Système d'informatique

والمكلف(ة) بإنجاز اعمال بحث(مذكرة، التخرج، مذكرة ماستر، مذكرة ماجستير، اطروحة دكتوراه).

عنوانها: Deep learning for Agricultural image classification
Apprentissage profond pour la classification d'images Agricoles

أصرح بشرفي اني ألتزم بمراعاة المعايير العلمية والمنهجية الاخلاقيات المهنية والنزاهة الاكاديمية المطلوبة في انجاز البحث المذكور أعلاه.

التاريخ: 2023/06/23

توقيع المعني (ة)

هيئة مراقبة السرقة العلمية:

البويرة في: 2023/07/01

النسبة: % 19

الامضاء

جامعة اكلي محند أولحاج - البويرة
كلية العلوم والعلوم التطبيقية

رئيس مصلحة الاعلام الآلي بالنيابة
جوبري عبد الرزاق

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire

وزارة التعليم العالي و البحث العلمي
جامعة أكلي محند أولحاج - البويرة

Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Akli Mohand Oulhadj - Bouira
Tasdawit Akli Muhend Ulhağ - Tubirett

التصريح الشرفي الخاص بالالتزام بقواعد النزاهة العلمية لإنجاز بحث

انا الممضي اسفله،

السيد(ة)........Badis Lamus..........الصفة: طالب (ماستر / دكتوراه)

الحامل(ة) لبطاقة التعريف الوطنية: ......H011850975.......والصادرة بتاريخ 2022-03-25

المسجل(ة) بكلية / معهد Des Sciences et Sciences Appliquées قسم ....informatique.......

تخصص: ....génie Système informatique..........

والمكلف(ة) بإنجاز اعمال بحث(مذكرة، التخرج، مذكرة ماستر، مذكرة ماجستير، اطروحة دكتوراه).

عنوانها: ....Deep learning for Agricultural image classification....

....Apprentissage profond pour la classification d'images Agricoles....

أصرح بشرفي اني ألتزم بمراعاة المعايير العلمية والمنهجية الاخلاقيات المهنية والنزاهة الاكاديمية المطلوبة في انجاز البحث المذكور أعلاه.

التاريخ: ....2023/06/23....

توقيع المعني(ة)

هيئة مراقبة السرقة العلمية:

النسبة: % 19

البويرة في: 2023/07/01

الامضاء

قسم الاعلام الآلي
بالنيابة
جابري عبد الرزاق

Acknowledgements

First of all, we thank our God for giving us the strength and courage to carry out this work.

First and foremost, we extend our deepest thanks to our thesis advisor **WAHIBA ALIOUAT** for their invaluable guidance, expertise, and unwavering support. Their mentorship and constructive feedback have played a pivotal role in shaping the direction and quality of this research work.

We extend our heartfelt appreciation to our esteemed professors of the department of computer science for their invaluable guidance and unwavering support throughout our academic journey. Their expertise and passion for teaching have inspired us to excel in the field of computer science. We are grateful for their dedication to our learning .

We are also grateful to the members of our thesis committee for their time, expertise, and valuable insights. Their thoughtful comments and suggestions have significantly enhanced the depth and rigor of our study.

We extend our appreciation to our colleagues and friends for their support, collaboration, and stimulating discussions throughout this endeavor. Their contributions and shared knowledge have enriched our understanding and broadened our perspectives.

Dedication

I dedicate this thesis to:

My dear father AISSA and my dear mother AMEL, for their unwavering love, boundless support, and endless sacrifices. Your guidance and encouragement have shaped me into the person I am today.
To my dear brothers ISSAM and SAID , your unwavering support and camaraderie have been a source of strength and inspiration. Lastly, to my cherished friends, your presence and unwavering belief in me have lifted my spirits and fueled my determination. I am forever grateful for the love, support, and friendships that have enriched my life.

Kenza Bouchiba.

Dedication

This thesis is a tribute to everyone who has encouraged and helped me in my academic career. It acknowledges my family, mentors, and friends for their unwavering support, love, and faith in me. Their guidance and insights have helped me overcome obstacles and achieve my goals. It also expresses gratitude for their companionship, providing support and laughter during the highs and lows of this thesis. Overall, this experience has been enhanced and made more unforgettable by their companionship.

Lamis Badis

ملخص

أصبحت الشبكات العصبية التكرارية شائعة جدًا في مهام تصنيف الصور، ولكن تصميمها الأمثل يمكن أن يكون تحديًا بسبب الفضاء الهائل من التركيبات الممكنة لعدد الطبقات وقيم المعلمات الفرعية المرتبطة بها. قد يحتاج اختيار نموذج الشبكات العصبية التكرارية المثلى لمهمة معينة للوقت الكثير (لتدريب العديد من النماذج المختلفة). من أجل حل هذه المشكلة، اقترحنا نهج تصميم تلقائي للـ الشبكات العصبية التكرارية يستند على عدة تقنيات. الأولى تستخدم نماذج مدربة مسبقًا، والثانية تستخدم نظام بلوك كأساس وتعتمد على خوارزميات تطورية مختلفة، بما في ذلك خوارزمية الذئاب الرمادية للتحسين و الخوارزمية الجينية. في هذا العمل، نهدف إلى إنشاء وتقييم تلقائي لتصاميم الشبكات العصبية التكرارية المرشحة لتصنيف شتلات النباتات، مع التمييز بين نباتات الأعشاب الضارة ونباتات المحاصيل. سنقدم أيضًا نظام تشفير للـالشبكات العصبية التكرارية وقيم المعلمات الفرعية المقابلة لها. تجمع طرقنا المقترحة بين قوة الخوارزميات التطورية مع التعلم المنقول ونظام البلوك لاستخلاص ملامح معنوية من الصورة. من خلال استغلال هذه التقنيات، تحققت طريقتنا من دقة استثنائية، متفوقة على أحدث الطرق المتاحة بدقة التحقق تصل إلى ٩٧٫٨٣%. هذا النهج يوفر أداة ثورية لتحسين دقة نماذج الشبكات العصبية التكرارية، مصممة خصيصًا للمجموعة البيانات المستخدمة.

كلمات مفتاحية : الشبكات العصبية التكراريه ، تصنيف الشتلات النباتية ، الخوارزميات التطورية ، النماذج المدربة مسبقًا

Abstract

Convolutional neural networks (CNNs) have gained significant popularity in image classification tasks. However, designing an optimal CNN model can be challenging due to the vast space of possible combinations of layer numbers and associated hyperparameter values. Selecting the best CNN model for a specific task often requires extensive training of numerous models, resulting in time-consuming processes. To address this problem, we propose a novel automated approach for CNN architecture design. Our proposed framework consists of two alternative variants : the first one utilizes pre-trained models, while the second one utilizes a bloc system as a backbone for the two variants, we employ two different evolutionary algorithms, namely gray wolf optimization and genetic algorithm. The primary objective of our research is to automatically generate and evaluate candidate CNN architectures for plant seedling classification, specifically distinguishing between weed and crop seedlings. In addition, we introduce an encoding system for representing CNN architectures and their corresponding hyperparameters. By combining evolutionary algorithms with transfer learning and the bloc system, we aim to extract meaningful features from images. Through extensive experimentation, our proposed method achieves exceptional accuracy, surpassing state-of-the-art approaches, with a validation accuracy of up to 97.83%. This approach provides a revolutionary tool for enhancing the accuracy of CNN models, tailored specifically to the dataset under consideration.

Key words: Convolutional neural networks,automated approach, pre-trained model ,evolutionary algorithms...

Résumé

Les réseaux neuronaux convolutifs (CNN) sont devenus de plus en plus populaires pour les tâches de classification d'images, mais leur conception optimale peut être difficile en raison de l'immensité de l'espace des combinaisons possibles du nombre de couches et des valeurs hyperparamétriques associées à chaque couche. Sélectionner un modèle CNN optimal pour une tâche spécifique peut encore nécessiter beaucoup de temps (pour entraîner de nombreux modèles différents). Afin de résoudre ce problème, nous proposons une approche de conception automatisée de l'architecture CNN qui repose sur plusieurs techniques. La première repose sur l'utilisation de modèles pré-entraînés et la seconde sur l'utilisation d'un bloc en tant que structure de base en utilisant deux algorithmes évolutionnaires différents : l'optimisation du loup gris et l'algorithme génétique. Dans le cadre de ce travail, nous cherchons à générer le meilleur modèle possible capable de classer les plantules de mauvaises herbes et de cultures. Nous introduisons également un système d'encodage pour les architectures CNN et leurs hyperparamètres correspondants. Nos approches combinées exploitent la puissance des algorithmes évolutionnaires, de l'apprentissage par transfert et du système de bloc pour extraire des caractéristiques significatives à partir d'une image. En utilisant ces techniques, notre méthode atteint une précision exceptionnelle, surpassant les méthodes de pointe avec une précision de validation pouvant atteindre 97,74 %. Cette approche offre un outil révolutionnaire pour améliorer la précision des modèles CNN, spécifiquement adapté à l'ensemble de données utilisé.

Mots clés: Les réseaux neuronaux convolutifs,une approche de conception automatisée, modèles pré-entraînés,algorithmes évolutionnaires...

Contents

List of Figures

List of Tables

Abreviations list

CNN	Convolutional neural network
RNN	Recurrent neural network
GAN	Generative adversarial network
NAS	Neural architecture search
EA	evolutionary algorithm
GWO	Gray Wolf Optimization
GA	Genetic Algorithm
TL	Transfert Learning
BL	Bloc
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives
N	Total instances

General introduction

The agricultural industry is undergoing a profound transformation propelled by advancements in technology and the integration of artificial intelligence (AI) into farming practices. Within this context, the emergence of smart farming has become a focal point, ushering in a new era of intelligent and efficient agricultural methodologies. Precision agriculture, in particular, plays a crucial role in optimizing farming practices by utilizing tailored solutions based on the specific characteristics of agricultural fields. This approach aims to maximize crop productivity, minimize resource usage, and mitigate environmental impact, all of which are imperative in addressing the challenges posed by a growing global population.

One significant aspect of precision agriculture revolves around the accurate identification and differentiation of plants and weeds within agricultural fields.

Convolutional Neural Networks (CNNs), a powerful class of deep learning algorithms, have demonstrated exceptional capabilities in image recognition and classification tasks. Their ability to automatically learn and extract relevant features from images makes them well-suited for the intricate task of plant and weed classification. However, the design of an optimal CNN architecture specifically tailored to agricultural datasets remains a challenge.

Therefore, the primary objective of this dissertation is to propose and evaluate an automated approach for generating CNN architectures specifically optimized for plant and weed classification within the realm of precision agriculture.

The objective of the work

The objective of our work is to develop a framework for automatic CNN generation by leveraging the power of evolutionary algorithms, specifically genetic algorithms and gray wolf optimization. The focus will be on harnessing the capabilities of pre-trained models and the bloc system as fundamental components of the architecture generation process. By integrating these techniques, the aim is to:

- develop a framework that can automate the generation of high-performing CNN architectures tailored to specific datasets, eliminating the extensive need for human interventions, reducing dependence on expert knowledge, and minimizing the need for trial and error.
- overcome the limitations of manual design by leveraging the power of evolutionary algorithms in combination with pre-trained models and the bloc system to efficiently explore the design space and identify architectures that exhibit superior performance

Dissertation Outline

The remaining sections of the dissertation are organized as follows:

- In the first chapter, We will start by providing a brief overview of agriculture, its evolution, and its type. Then, we will present an overview of precision agriculture.
- In the second chapter, the background of our work is presented. We focus on convolution neural networks (CNN), neural architecture search (NAS). On the other hand, we will explain the evolutionary algorithm used in our framework. Ultimately, a stat of art about deep learning and weed classification is presented.
- The third chapter covers the used dataset and the design of our proposed framework.
- The last chapter summarizes our results, analyzes the different outcomes of our proposed framework, and includes a comparative section to highlight the differences between the 4 variants of our framework as well as this work and earlier ones.

Chapter 1

Precision agriculture

1.1 Introduction

The practice of agriculture has undergone tremendous change over the years, shaped by advances in technology and a growing global population. On the one hand, we have traditional agriculture, which relies on centuries-old techniques and knowledge passed down from generation to generation. On the other hand, precision agriculture leverages technology and data analysis to maximize efficiency and yield. In this chapter, we will delve into the key differences between traditional and precision agriculture, exploring their respective approaches, techniques, and benefits. By comparing and contrasting these two forms of agriculture, we will gain a deeper understanding of the impact they have on the environment, the economy, and our lives.

By 2050, the world's population is projected to reach 9 billion, requiring a significant increase in food production. In developing countries, food production must nearly double to meet the demands of a growing population. The global climate is expected to warm by an additional 1.5 degrees Celsius by 2050, as predicted by NASA, resulting in changes in temperature, rainfall patterns, and extreme weather events. These changes are anticipated to negatively impact food production.

To counteract these challenges, farmers must adapt their planting schedules, irrigation and fertilizer management, and crop selection to mitigate the effects of a changing climate.

In addition to the impacts of climate change, traditional agriculture faces other significant threats, such as pests, weeds, and plant diseases.

These stressors not only threaten food security but also have a devastating effect on small-scale farmers, where up to 50% of crops can be lost to these factors.

1.2 Agriculture

1.2.1 Definition of agriculture

Agriculture is the practice of cultivating land, raising animals, and producing food, fiber, and other products through systematic and sustainable methods. It has played a critical role in the development of human civilization, providing a reliable source of food that allowed populations to grow and settle in one place, leading to the formation of cities and the rise of civilization.

1.2.2 The evolution of agriculture

Agriculture is the practice of cultivating soil, raising livestock, and producing food, fiber, and other useful products. It is the foundation of human civilization and has undergone a tremendous evolution over the past 10,000 years.
In this article, we will explore the major milestones in the evolution of agriculture and its impact on human society.

1.The beginning of agriculture:

The transition from hunting and gathering to agriculture is considered one of the most important developments in human history. It was a gradual process that took place independently in several regions of the world, including the Fertile Crescent in the Middle East, the Yellow River Valley in China, and the Indus Valley in South Asia.
The first agricultural societies emerged around 10,000 years ago, at the end of the last Ice Age.
People began to settle in one place and cultivate crops, such as wheat, barley, and legumes. They also domesticated animals, such as sheep, goats, and cattle, for food and labor.

2.The agricultural revolution:

Around 5,000 years ago, a major agricultural revolution occurred in several parts of the world. New techniques, such as irrigation, plowing, and the use of draft animals, were developed, allowing farmers to increase their yields and produce surplus food.
This surplus allowed for the growth of cities and the development of specialized trades

and professions.

The use of iron tools and the development of new crop varieties further increased agricultural productivity.

The spread of agriculture also led to the diffusion of ideas and technologies, such as the use of the wheel and writing systems.

3.The green revolution:

In the mid-20th century, a new agricultural revolution, known as the Green Revolution, took place.

It was a period of rapid technological innovation and scientific advancement in agriculture. New crop varieties were developed through hybridization and genetic engineering, and new agricultural practices, such as the use of chemical fertilizers and pesticides, were introduced. The Green Revolution led to a significant increase in crop yields, particularly in developing countries, and helped to alleviate hunger and poverty.

However, it also had some negative consequences, such as the depletion of soil nutrients and the contamination of water resources.

4.Modern agriculture:

Today, agriculture has become increasingly industrialized and commercialized. Large-scale farms use advanced technologies, such as precision farming and satellite imagery, to increase efficiency and productivity.

The use of genetically modified crops and synthetic fertilizers and pesticides is also widespread.

However, modern agriculture is facing new challenges, such as climate change, soil degradation, and water scarcity. There is a growing interest in sustainable and organic agriculture, which emphasizes the use of natural resources and the protection of the environment.

The evolution of agriculture has been a long and complex process that has shaped human societies in profound ways.

From the first farmers to modern industrialized agriculture, agriculture has played a crucial role in providing food, livelihoods, and economic development. As we face new challenges in the 21st century, it is essential to continue to innovate and develop sustainable and equitable agricultural practices that benefit both people and the planet.

1.2.3 Different types of agriculture

There are different types of agriculture, each with its unique features and practices. In this article, we will explore the various types of agriculture, including the latest type, precision agriculture.

1.Subsistence agriculture:

Subsistence agriculture is the oldest and most traditional form of agriculture. It involves small-scale farmers who cultivate crops for their own consumption or to feed their families. Subsistence farmers use traditional farming techniques, such as hand tools, and rely on natural rainfall to grow their crops. This type of agriculture is common in developing countries and rural areas.

2.Commercial agriculture:

Commercial agriculture involves large-scale farmers who grow crops for sale to markets and processing companies.

Commercial farmers use modern farming techniques, such as irrigation, mechanization, and synthetic fertilizers and pesticides, to increase yields and productivity. This type of agriculture is common in developed countries and is a significant contributor to the global food supply chain.

3.Intensive agriculture:

Intensive agriculture is a type of agriculture that involves high-input farming techniques to maximize productivity.

Intensive agriculture includes practices such as crop rotation, intercropping, and the use of high-yield crop varieties. This type of agriculture is common in areas with high population densities and limited arable land.

4.Extensive agriculture :

Extensive agriculture involves farming on a large scale with low inputs and low productivity.

Extensive agriculture is common in areas with large tracts of land and low population densities.The farming techniques used in extensive agriculture are often traditional, such as grazing and nomadic herding.

5.Organic agriculture:

Organic agriculture is a type of agriculture that emphasizes the use of natural inputs and sustainable farming practices. Organic farmers use natural fertilizers, such as compost, and avoid synthetic pesticides and fertilizers. Organic agriculture is gaining popularity among consumers who prefer natural and healthy foods.

6.Precision agriculture:

Precision agriculture is the latest type of agriculture that uses modern technology, such as satellite imagery, drones, and sensors, to optimize farming practices. Precision agriculture involves the use of data and analytics to increase efficiency and productivity while minimizing waste and environmental impact.

Precision agriculture is based on the principle that every farm is unique and requires a personalized approach. Farmers can use precision agriculture to identify areas of their fields that need more or less water, fertilizer, or pesticides. This allows farmers to apply inputs only where they are needed, reducing waste and improving yields.

Precision agriculture also enables farmers to monitor crops remotely, detecting potential problems such as disease outbreaks or nutrient deficiencies before they become severe. This allows farmers to take timely corrective action and minimize losses.

In conclusion, agriculture has evolved over the years, and there are different types of agriculture, each with its unique features and practices. Subsistence agriculture is the oldest and most traditional form of agriculture, while precision agriculture is the latest type of agriculture that uses modern technology to optimize farming practices. Each type of agriculture has its advantages and disadvantages, and farmers must choose the one that best suits their needs and goals. As we face new challenges in the 21st century, such as climate change, soil degradation, and water scarcity, precision agriculture offers a promising solution to improve productivity while minimizing waste and environmental impact.[1]

1.2.4 The daunting obstacles conventional agriculture must overcome

1.Environmental problem

Climate change and water scarcity are two significant challenges that are having a profound impact on traditional agriculture. Rising temperatures and changing weather patterns are causing crop yields to decline and exacerbate the spread of pests and diseases. Additionally, water scarcity is becoming a growing concern for traditional farmers, as prolonged droughts, declining water tables, and changes in rainfall patterns are reducing

the availability of water for irrigation and crop growth.
This is particularly true in arid regions, where water resources are already limited. The combined effects of climate change and water scarcity are making it increasingly difficult for traditional farmers to grow crops and raise livestock, and threaten the viability of traditional agriculture as a source of food and livelihood for communities around the world.

2.Yield problem:

Pests, weeds, and plant diseases are significant challenges for traditional agriculture. These biotic stressors can cause significant damage to crops and reduce yields, negatively impacting the economic viability of traditional farming operations.
In traditional agriculture, managing these stressors is often done through manual labor and simple tools, making it difficult to effectively control their spread.
The impact of pests, weeds, and plant diseases highlight the need for more sustainable and effective methods of management in traditional agriculture.

A.The weed problem:

Weeds are considered a significant problem in traditional agriculture due to their ability to reduce crop yields significantly. They grow rapidly and compete with crops for space, sunlight, water, nutrients, and other essential resources, which results in decreased crop output.
Furthermore, weeds can harbor pathogens and insects that harm crops, leading to further yield loss. In traditional agriculture, ineffective methods like hand weeding result in large losses and rising labor costs.
In the US corn belt, it has been estimated that weed invasion can lead to a reduction in corn and soybean crop yields by nearly 50 % in fields without weed control compared to fields with weed control.[1] Similarly, a study in India estimated that crop yields were reduced by 31.5% due to weeds, with a higher reduction of 36.5% observed during the summer and rainy seasons.[2] Poor weed management has also resulted in significant yield loss in cotton production, with the potential for a loss of up to 90%. On a global scale, the potential crop yield loss due to weeds is estimated at 43%. A study of soybean production demonstrated a drop of 50-76% in yields, while groundnut production saw a yield loss of 45-71% due to weeds.[2]
In conclusion, weeds pose a significant threat to crop yield and will continue to do so under

[1] **https://www.sciencedirect.com/science/article/abs/pii/S0261219418300073**

the impact of climate change. Therefore, it is crucial to implement effective weed control measures to ensure sustainable crop production and food security. The potential loss due to weeds is estimated to be the highest (34%) compared to losses due to pathogens and animal pests, which were less significant (18% and 16% respectively) globally. Effective weed control will play an essential role in ensuring sustainable agriculture and food security in the future.

B.The pest problem :

The consequences of pests in agriculture are significant, as they compete with humans for resources and cause significant damage to crops. The Food and Agriculture Organization of the United Nations (FAO) estimates that up to 40% of global crop production is lost annually due to pests, costing the global economy around $70 billion. [2] Insects account for much of these losses, with plant-protection measures only able to prevent 42.1% of the potential production from being lost. The global estimate of crop yield losses due to various pests is $500 billion, indicating the significant impact pests have on crop yield and food security. [3]

Moreover, the problem is expected to be exacerbated by climate change, which will increase the risk of pests spreading in agricultural and forestry ecosystems, particularly in the Arctic, boreal, temperate, and subtropical regions. Pests occur in many groups of organisms, including insects, weeds, diseases caused by fungi, viruses, bacteria, other microorganisms, nematodes, rodents, and birds. Food plants are damaged by 10,000 species of insects, 30,000 species of weeds, 100,000 diseases, and 1000 species of nematodes.[3] Therefore, controlling and managing pests effectively is crucial to minimize the damage they cause to crops and ensure food security.

C.The plant disease problem:

Plant diseases are considered a problem because they can prevent or alter the normal functioning of the plant. All plant species, whether wild or domesticated, are susceptible to disease. Diseases can vary in occurrence and prevalence depending on the presence of pathogens, surrounding conditions, and the crops and varieties being grown. Some plant varieties are more vulnerable to disease outbreaks than others.

Disease symptoms in plants often appear on leaves, fruit, buds, and young branches, causing destruction or waste of the fruit. Additionally, these diseases can lead to the

[2] **https://www.fao.org/news/story/en/item/1402920/icode/**

[3] **https://www.fao.org/3/y5800e/Y5800E06.htm**

development of new infections and the spread of the illness due to factors such as seasonal weather, making it important to identify and prevent the disease from spreading.

The Food and Agriculture Organization of the United Nations (FAO) estimates that pests and diseases result in the loss of 20-40% of global food production, posing a threat to food security. Plant diseases cost the global economy over $220 billion each year and account for roughly 10-20% reductions in the global production of food[4]. In the case of cotton, plant pests, and diseases can potentially reduce the yield by 82%. Plant diseases are also thought to be responsible for 10-16% of the annual harvest loss worldwide, with disease losses costing $220 billion.

Additional postharvest losses range from 6-12%, particularly high in developing tropical nations without proper infrastructure.

1.3 Precision agriculture

1.3.1 Definition of precision agriculture

Precision agriculture, also known as precision farming or site-specific crop management, is a farming management concept based on observing, measuring, and responding to intra-field variations in crops. The goal of precision agriculture is to optimize the yield of a field by applying the right amount of inputs (such as fertilizers, pesticides, water, and fuel) at the right time and place in order to maximize the efficiency of agricultural production.

Precision agriculture began in the early 1980s with the development of yield monitors, which were installed on combine harvesters to measure the amount of grain being collected from a field. This data was then used to create yield maps, which showed the variation in crop yields from one location to another. With the advent of GPS technology in the 1990s, farmers were able to map their fields and measure field boundaries, slopes, and aspects. This data, combined with yield maps, helped farmers to better understand the factors that affected crop yields.

In the past decade, there has been a continuous development of new technologies for precision agriculture, such as sensors, robotics, and drone technology. These technologies are being used to collect data about soil moisture, nutrients, crop canopy, and more. This data is then analyzed to help farmers make informed decisions about how to optimize their production.

Precision agriculture has its roots in the early 20th century when farmers started using aerial photos and maps to make informed decisions about planting and land management. Over time, the development of new technologies such as GPS, remote sensing, and big data analytics has allowed farmers to collect and process vast amounts of data to make more precise decisions.

Precision agriculture is becoming increasingly important as farmers strive to produce more food with fewer inputs. By using precision agriculture techniques, farmers can reduce their input costs, conserve natural resources, and improve their yields. As technology continues to advance, precision agriculture will likely play an increasingly important role in ensuring food security for a growing global population.

The use of precision techniques is expected to become increasingly widespread as more farmers adopt new technologies and as data processing and analysis become more accessible and cost-effective.[5]

1.3.2 The remarkable advantages of precision agriculture:

Precision agriculture is a modern approach to farming that leverages technology to increase efficiency, reduce costs, and improve crop yields. Some of the key benefits of precision agriculture include increased crop yields, reduced input costs, and improved resource utilization. Precision agriculture allows farmers to make more informed decisions about how and when to plant, irrigate, and apply fertilizers and pesticides, leading to better utilization of resources and improved soil health.

Another benefit of precision agriculture is enhanced sustainability. By using data-driven insights, farmers can make more informed decisions about how to use natural resources in a way that is both profitable and environmentally responsible. This can result in reduced water usage, lower greenhouse gas emissions, and a smaller carbon footprint. Precision agriculture also allows farmers to track and monitor crop performance in real-time, leading to more accurate yield predictions and improved decision-making.

Overall, precision agriculture has the potential to revolutionize the way food is grown, reducing waste and increasing profitability for farmers while providing consumers with more sustainably grown food.

1.3.3 The innovative solutions to tackle the complex challenges of agriculture

Precision farming is revolutionizing agriculture by using advanced technologies and innovative solutions to address the complex challenges facing the industry. By leveraging real-time data, precise resource management, and tailored inputs, precision agriculture technologies are helping farmers increase efficiency, reduce waste, and improve yields. In this article, we will explore ten key precision farming technologies and innovations.

1.Yield monitoring systems: real-time datafor better crop management .

Yield monitoring systems track and record crop yields in real time, providing farmers with data-driven insights into their operations. By measuring yield variability, farmers can identify areas of their fields that require different inputs, such as fertilizer or pesticides, allowing for more precise and efficient resource management.

2.Variable rate application technology: Optimizing resource use and efficiency

Variable rate application technology enables farmers to vary the rate of inputs, such as fertilizers and pesticides, based on the specific needs of different areas within a field. This reduces waste and improves overall efficiency by ensuring that inputs are applied only where and when they are needed.

3.Precision planting equipment: accurate seed placement for improved yields

Precision planting equipment uses advanced technologies to plant crops with accuracy and precision, optimizing seed placement and leading to improved yields. By planting seeds at the optimal depth and spacing, farmers can maximize their yields and reduce seed waste.

4.Precision irrigation systems: Precise water management for reduced waste

Precision irrigation systems enable farmers to optimize water usage by delivering precise amounts of water to specific areas within a field. This reduces water waste and improves efficiency by ensuring that crops receive the right amount of water at the right time.

5.Remote sensing technologies: real-time data for informed decision-making

Remote sensing technologies, such as satellites and drones, provide farmers with real-time data about crop health, soil moisture, and weather conditions. This information enables farmers to make more informed decisions about when and where to apply inputs and to detect potential issues before they become problems.

6.Unmanned aerial vehicles (UAVs) or drones: versatile tools for precision agriculture.

UAVs or drones equipped with cameras and sensors can be used for a range of precision agriculture applications, such as crop mapping, monitoring crop health, and applying inputs. This technology allows farmers to quickly and easily survey their fields and identify potential issues, reducing the need for manual labor and improving efficiency.

7.Soil and nutrient mapping systems: tailored inputs for optimal resource use

Soil and nutrient mapping systems use sensors and other technologies to map soil characteristics, nutrient levels, and other important factors, enabling farmers to tailor inputs and practices to specific areas within a field. This ensures that crops receive the right nutrients and resources to maximize yields while minimizing waste.

8.Climate and weather tracking systems: adaptation to a changing climate

Climate and weather tracking systems provide farmers with real-time weather and climate data, enabling them to make better decisions about how to adapt to a changing climate. By understanding the potential impacts of weather and climate variability, farmers can adjust their practices and inputs to minimize risk and maximize yields.

9.Data management and analysis Platforms: turning data into insights

Data management and analysis platforms enable farmers to collect, store, and analyze vast amounts of data, providing them with data-driven insights into their operations. By analyzing this data, farmers can make more informed decisions about inputs, resource usage, and overall farm management.

10.Artificial intelligence (AI) and machine learning algorithms: Predictive analytics for improved operations

AI and machine learning algorithms can analyze vast amounts of data, providing farmers with predictive analytics and other valuable insights into their operations. This enables farmers to make more informed decisions about when and where to apply inputs, how to optimize resource usage, and how to improve overall farm management.

1.3.4 The Potential impact of Precision farming technologies

AI and machine learning are transforming farming practices by offering the following benefits:

A.Increased efficiency:

Precision agriculture technologies enable farmers to optimize resource usage and reduce inputs, leading to more efficient and cost-effective operations.

B.Better resource utilization:

Precision agriculture provides farmers with site-specific management solutions that allow them to tailor inputs and practices to specific areas within a field, leading to improved resource utilization and increased yields.

C.Enhanced sustainability:

Precision agriculture enables farmers to make more informed decisions about how to use natural resources in a way that is both profitable and environmentally responsible, reducing waste and increasing sustainability.

D.Improved decision-making:

Precision agriculture provides farmers with data-driven insights into their operations, enabling them to make more informed decisions about inputs, resource usage, and overall farm management.

E.Increased profitability:

By utilizing precision agriculture technologies, farmers can reduce their overall input costs and increase yields, leading to improved profitability.

F.Predictive maintenance:

AI and machine learning algorithms can monitor and predict the maintenance needs of farming equipment, reducing downtime and increasing productivity.

G.Improved yields:

AI and machine learning can help farmers optimize planting and harvesting schedules, leading to increased yields and improved profitability.

Overall, AI and machine learning are revolutionizing farming practices by providing farmers with the tools and insights they need to optimize their operations, improve efficiency, and enhance sustainability.

1.4 Conclusion

The evolution of agriculture has led to two distinct approaches: traditional agriculture based on time-honored practices and precision agriculture fueled by technological advancements. By exploring the disparities between these methods, we have gained insights into their benefits and implications for the environment, economy, and society. Understanding the nuances and potential synergies between these approaches is essential for sustainable and efficient food production.In the next chapter we will provide a background to our project.

Chapter 2

State of the art

2.1 Introduction

The development of deep learning in computer vision has drawn the attention of many researchers throughout the years. Precision agriculture currently makes use of artificial intelligence, machine learning, and computer vision technologies to identify crop diseases, forecast weather, and determine crop yield.
In this chapter we present first a “Background” on deep learning , CNN applications in agriculture, Neural architecture search and finally some Research works in weed managements.

2.2 Machine learning

Machine learning is a branch of artificial intelligence that focuses on creating statistical models and algorithms that let computers "learn" from data and predict or decide based on it without being explicitly programmed to do so.

2.3 Deep learning

A branch of machine learning called deep learning is concerned with creating artificial neural networks, which are algorithms inspired by the structure and operation of the brain. It models and addresses difficult issues including speech and picture recognition, decision-making, and natural language processing using many layers of artificial neural networks.

These deep neural networks can adapt and get better with practice, and they can deal with huge, complicated, and unstructured datasets. Finance, healthcare, autonomous systems, and agriculture are just a few of the sectors that have benefited greatly from deep learning's breakthrough in the field of artificial intelligence.

2.3.1 Types of deep neural networks

Convolutional neural networks

A convolutional neural network (CNN) or ConvNet is a type of deep neural network primarily used in computer vision applications. CNNs are designed to process and analyze visual data, such as images and videos. By learning to automatically extract features from real-world inputs, CNNs can perform tasks such as image classification, face recognition, and image segmentation. The network achieves this by using a series of convolutional layers that apply filters to the input data, allowing it to identify patterns and features within the images.[6]

Recurrent neural networks (RNN)

A recurrent neural network (RNN) is a type of artificial neural network that is designed to process sequential data. It has been developed specifically to address time-series problems involving sequential input data. The input to an RNN consists of the current input as well as the previous samples. As a result, the connections between nodes form a directed graph along a temporal sequence. Additionally, each neuron in an RNN has an internal memory that stores information from previous computations. RNNs are used in various applications, such as speech recognition, machine translation, predicting the next action in a sequence.[7]

Generative adversarial network

A generative adversarial network (GAN) is a machine learning model consisting of a generator and a discriminator. The generator generates synthetic data samples, while the discriminator uses real and generated samples to differentiate between real and fake data. The training process involves updating weights and biases, allowing the generator to produce realistic data that deceives the discriminator. GANs have shown remarkable capabilities in various domains, including image generation, text synthesis, and audio synthesis, contributing to advancements in computer vision, natural language processing, and creative applications.[8]

2.4 Comparison between machine learning and deep learning

Machine learning and deep learning are both subfields of artificial intelligence, but they have some differences.

A.Approach

Machine learning algorithms are based on mathematical models that enable the computer to learn from and make predictions on data and can be divided into two categories: supervised and unsupervised learning.Deep learning, on the other hand, focuses on developing algorithms inspired by the structure and function of the brain, called artificial neural networks.

B.Complexity

Machine learning algorithms are generally less complex than deep learning algorithms and they are designed to handle simple to moderately complex problems. Deep learning algorithms, with their many layers, can handle more complex data and find more intricate patterns in it.

C.Data requirement

Machine learning algorithms typically require smaller amounts of data, while deep learning algorithms require larger amounts of data to train the model effectively.

D.Problem-solving ability

Deep learning is particularly good at solving complex problems that involve image and speech recognition, natural language processing, and decision-making. Machine learning, on the other hand, can be used to solve a wider range of problems, but may not perform as well on complex problems as deep learning does.

E.Use cases

Machine learning is used for a wide range of applications, including regression, classification, and clustering, while deep learning is mainly used for computer vision and natural language processing applications.

F.Accuracy

Deep learning algorithms often achieve higher accuracy than traditional machine learning algorithms, especially for complex problems, but this comes at the cost of increased computational resources and longer training times.

In conclusion, deep learning is a more complicated and specialized approach that is best suited for handling particular kinds of issues. Machine learning is a broader field that encompasses deep learning.

2.5 Computer vision

2.5.1 Definition of computer vision

Computer vision is a branch of artificial intelligence and computer science that examines how to make computers perceive, comprehend, and derive meaning from visual data in a manner similar to that of humans.This involves the development of algorithms and systems that can process, analyze and interpret images, videos and other forms of visual data to make intelligent decisions, recognize objects, track motion, recognize patterns and much more. The goal of computer vision is to enable computers to understand and interpret visual data in the same way that human beings do, with the ultimate aim of creating autonomous systems that can see, reason, and act.

2.5.2 Impact of computer vision

There are many domains where computer vision has made a big difference, like

A.Healthcare

Computer vision is used to analyze medical images, assist in diagnoses and surgery, and monitor patients.

B.Retail

Computer vision is used for monitoring customer behavior, product recognition.

C.Security

Computer vision is used for surveillance, facial recognition, and tracking individuals.

D.Entertainment

Computer vision is used in the production of movies and video games to create special effects, improve character animation, and provide new forms of interaction.

E.Transportation

omputer vision technologies are used in autonomous vehicles, enhancing safety and reducing the number of accidents caused by human error.

F.Precision agriculture

Computer vision can support precision agriculture by analyzing images of fields and crops to identify areas with different needs for irrigation, fertilization, or other interventions. This can help farmers optimize their use of resources, reduce costs, and increase efficiency. In general, computer vision has had a profound impact on many fields, enhancing effectiveness, accuracy, and safety while also opening up new possibilities and applications.

2.6 convolutional neural network (CNN)

2.6.1 Definition of CNN

A Convolutional Neural Network (CNN) is a type of artificial neural network used for image and video recognition tasks. It is designed to process data with grid-like topology, such as an image, and is particularly effective at identifying local patterns and correlations in image data.
The main idea behind a CNN is to use multiple filters to extract features from the input data, followed by one or more fully connected layers to perform the final classification.

2.6.2 CNN'S layers

Convolutional Neural Networks typically consist of multiple cascading layers, such as convolutional layers, pooling layers, and fully connected layers.
The architecture of cnn model as shown in figure 2.1[9]

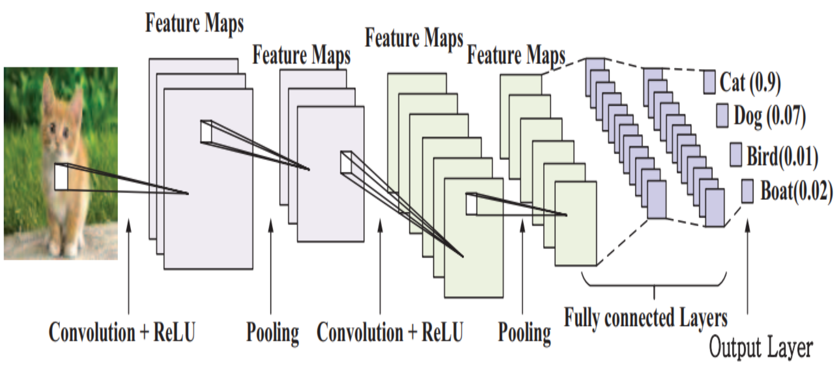

Figure 2.1: The deep convolutional neural network architecture.

A.convolutional layer

The Convolutional layer is the core component of Convolutional Neural Networks.
The main aim of the Convolutional layer is to learn feature representations of the inputs. The layer consists of several feature maps, each representing a specific type of feature. Each neuron in the same feature map extracts local characteristics of different positions in the previous layer. However, for a single neuron, it extracts local characteristics of the same position in different feature maps.
To obtain a new feature map, the input feature maps are first convolved with a learned kernel (also called a filter or a window). The kernel is a small matrix of weights that slides over the input feature map to compute the dot product between the kernel and the local region of the feature map. The result of the convolution operation is a new feature map that represents a specific type of feature. By applying different kernels, we can obtain different feature maps.
After the convolution operation, the results are passed into a nonlinear activation function, such as sigmoid, tanh, or ReLU (Rectified Linear Unit). The activation function introduces nonlinearity to the network, which enables it to learn complex representations of the input, the Convolutional layer is typically followed by a pooling layer.
The figure2.2 [10]

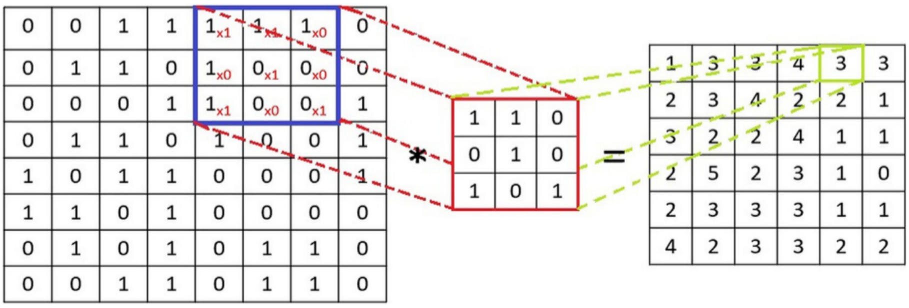

Figure 2.2: The convolution operation

B.pooling layer

The pooling layer plays an important role in convolutional neural networks.
It acts as a secondary feature extraction step and is typically placed between two convolutional layers. By dividing the input feature map into non-overlapping or overlapping regions and computing a summary statistic for each region, The feature maps' dimensions are reduced and feature extraction's robustness is increased by the pooling layer.
The size of the feature maps in the pooling layer is determined by the moving step of kernels. Additionally, the pooling layer helps to control over-fitting by progressively scaling

down the spatial size of the representation, which reduces the number of parameters and computations in the network.
The operation of pooling layer as shown in Figure 2.3 [9]

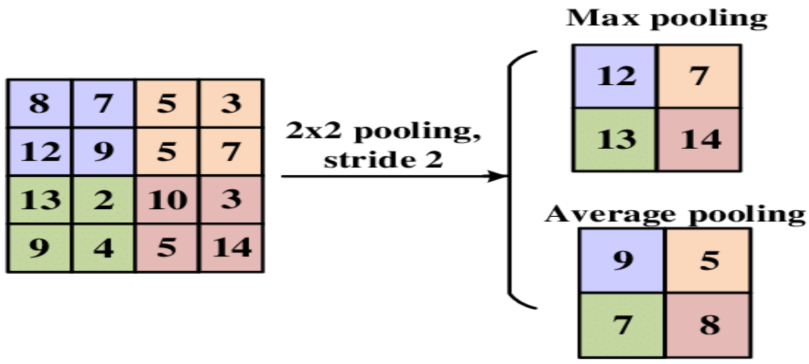

Figure 2.3: Pooling layer operation

C.Fully connected layers and softmax layers

Convolutional Neural Network (CNN) generally consists of one or more fully-connected layers. These layers take all the neurons from the previous layer and connect them to each and every neuron in the current layer. However, fully-connected layers do not preserve any spatial information. The last fully-connected layer is then followed by an output layer. In classification tasks. The Softmax function is often used in Convolutional Neural Networks (CNNs) to normalize the outputs of the last layer of a neural network. The Softmax function transforms the outputs into a probability distribution, which means that the outputs are between 0 and 1 and their sum is equal to 1. This allows to determine the most probable class for a given image.
Fully connected layer and softmax as shown in figure 2.4 [11]

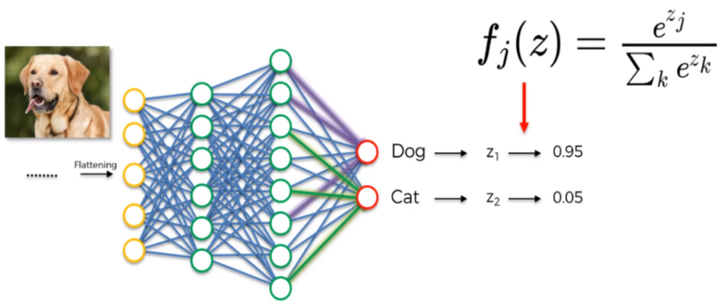

Figure 2.4: Fully connected layer and softmax layer

2.7 Pre-trained convolutional neural network

2.7.1 Transfert learning

transfer learning is a technique in deep learning where a pre-trained CNN model is used as a starting point for a new task. The pre-trained model is "transferred" to the new task by using its learned features as the initial weights for the new model, and then fine-tuning the model on the new task's dataset. This approach has several advantages, such as reducing the amount of data and computational resources required for training the new model, and often leading to improved performance compared to training a model from scratch. Transfer learning is commonly used in computer vision tasks where there is a shortage of labeled data.

Some popular pre-trained CNNs for transfer learning include

VGGNet

A very deep CNN architecture introduced in 2014, trained on the ImageNet dataset, which contains over a million images from 1000 different classes , VGG architecture serves as the foundation for innovative object recognition models, it is based on the most essential features of convolutional neural networks (CNN), The network is distinguished by its simplicity; the only other components are a fully connected layer and a pooling layer.[12]

ResNet-50

A deep residual network architecture introduced in 2015, is a 50-layer deep neural network that makes use of skip connections to solve the vanishing gradient issue. It performs well on image classification and object detection tasks, achieving state-of-the-art results and is widely used in research and industry.[13]

DenseNet-121

DenseNet121 is a deep convolutional neural network architecture that was introduced in 2016 for image classification tasks.It is part of the DenseNet family of architectures and is named after its number of layers, which is 121. It uses densely connected convolutional blocks, batch normalization, rectified linear units, and convolution in succession. It is efficient due to the use of bottleneck layers and has achieved state-of-the-art results on various image recognition tasks.[14]

Xception

Deep neural network design called Xception uses depth-wise separable convolutions instead of conventional layers of convolution, which requires less parameters and calculations. It uses skip and residual connections and performs well on image classification and object detection tasks, achieving state-of-the-art results.[15]

MobileNet

A light-weight CNN architecture designed for mobile and embedded devices that trade-off a little accuracy for a lot of speed.[16]

Inception v3 :

InceptionV3 is a deep neural network that uses a combination of convolutional layers and max pooling. It has an auxiliary classifier for training and regularization, and performs well on image classification tasks. InceptionV3 has achieved state-of-the-art results and is widely used in research and industry for computer vision applications.[17]

InceptionResNet V2

InceptionResNetV2 is a deep convolutional neural network architecture created by Google in 2016. It combines two popular architectures, Inception and ResNet, to create a highly accurate and efficient network for image classification tasks. It consists of inception modules followed by residual blocks with shortcut connections. It has achieved state-of-the-art results on image classification benchmarks and is commonly used as a starting point for transfer learning in computer vision tasks.[18]

2.8 Ovefitting

In the context of Convolutional Neural Networks (CNNs), overfitting can occur when the model becomes too complex and tries to fit the training data too closely, resulting in poor

performance on new data. This can happen when the model has too many parameters relative to the amount of available training data, or when the model is trained for too long.
One common symptom of overfitting in CNNs is a high accuracy on the training data but a lower accuracy on the validation or test data.
Another symptom is that the model may begin to memorize individual training examples, rather than learning general patterns that can be applied to new data.
To prevent overfitting in CNNs, various techniques can be used such as:

2.8.1 Transfer learning

Pre-trained models can be fine-tuned on a smaller dataset for a specific task, allowing the model to learn task-specific features without overfitting to the small dataset. By using the pre-trained model as a starting point, the model can learn from the pre-trained weights and avoid overfitting.

2.8.2 Dropout

This technique involves randomly dropping out some nodes during training to prevent the network from relying too heavily on any one particular node. This can help reduce overfitting by forcing the network to learn more robust and generalizable features.

2.8.3 Data augmentation

This technique involves applying random transformations to the training data, such as rotations or translations, to increase the size and diversity of the training dataset. This can help prevent the network from overfitting by providing more examples of the same class to the network.

2.8.4 Batch normalization

This technique involves normalizing the activations of each layer across the batch during training. This can help prevent the network from overfitting by reducing the internal covariate shift and making the network more robust to changes in the input data.

2.9 CNN applications in agriculture

Convolutional Neural Networks (CNNs) have various applications in agriculture, including:

A.Crop variety classification

CNNs can be used to classify different crop varieties based on visual characteristics, helping farmers select the best varieties for their conditions.

B.Plant disease detection

CNNs can be used to identify diseases in crops by analyzing images of infected plants. This can help farmers make early interventions to prevent crop losses.

C.Crop yield prediction

CNNs can be trained to predict crop yields based on factors such as weather patterns, soil moisture, and historical yield data.

D.Livestock Monitoring

CNNs can be used for monitoring livestock and detecting early signs of illness, injury, or stress.

E.Soil moisture estimation

CNNs can be used to estimate soil moisture levels based on factors such as climate data, topography, and satellite imagery.
These are just a few examples of how CNNs can be applied in agriculture. The technology is constantly evolving, and new applications are likely to emerge in the future.

2.10 Neural architecture search

Neural architecture search (NAS) has rapidly become a cutting-edge field in the realm of deep learning that aims to revolutionize the process of designing convolutional neural network (CNN) architectures. Traditionally, CNN structures have been created through manual trial-and-error procedures, which can be tedious and time-consuming and often require expert knowledge. However, with the advent of NAS, automatic CNN architecture search has become a feasible alternative, with optimization techniques being used to build and evaluate candidate structures automatically.

2.10.1 Traditional CNN design

The design of CNN architectures has been the focus of deep learning research for many years. A CNN consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers. These layers work together to extract features from input data and produce an output. The architecture of a CNN plays a crucial role in its performance, and researchers have spent significant effort trying to design effective architectures.

In traditional CNN architecture design, researchers manually design network architectures based on their intuition and prior knowledge. They start by selecting the number of layers and the type of layers to use, such as convolutional, pooling, or fully connected layers. Then, they decide on the size of the filters used in the convolutional layers and the number of filters in each layer. Finally, they set the hyperparameters, such as learning rate, batch size, and number of epochs, for training the network.

Once the architecture is designed, researchers train the network on a dataset and evaluate its performance. If the performance is not satisfactory, they repeat the process, making modifications to the architecture and hyperparameters until a satisfactory result is obtained. This trial-and-error approach can be time-consuming, as researchers may need to train hundreds or even thousands of models before finding an optimal solution.

Drawbacks of traditional CNN Architecture design:

A. Human Bias One of the most significant drawbacks of traditional CNN architecture design is human bias. Since humans design the architecture, their preconceived notions and biases can influence the design process. This can lead to suboptimal architectures that may not perform as well as they could.

B.Limited search space

Another drawback of traditional CNN architecture design is the limited search space of potential architectures that humans can explore. Since the design process is manual, it is not feasible for humans to explore a vast search space of potential architectures. This can lead to suboptimal architectures that may not perform as well as they could.

C.Time-Consuming

Traditional CNN architecture design can be time-consuming and require significant expertise in the field of deep learning. Depending on how complicated the process is, designing a new architecture may take many weeks or even months. This can limit the ability to develop new models quickly and can be a significant bottleneck in the research process.

2.10.2 Automatic CNN design

Automatic CNN architecture design is a relatively new and rapidly developing field that aims to automate the process of designing neural network architectures. This field is gaining more and more attention due to the significant improvements in neural network performance that can be achieved by automatic architecture design techniques. The primary goal of automatic architecture design is to discover high-performing CNN architectures that are optimized for specific tasks without human intervention.

There are several NAS methods available for the automatic creation of CNN architectures, including grid search[19], random search[20], reinforcement learning[21], Bayesian optimization[22], and evolutionary algorithms (EA)[23]. Evolutionary algorithms, in particular, have shown tremendous promise in automating the design process of CNN architectures. EA is an optimization method that employs the concept of natural selection to repeatedly enhance a population of candidate solutions. This process facilitates the discovery of high-quality CNN architectures that can outperform human-designed networks while satisfying specific constraints or objectives.

The versatility and adaptability of evolutionary algorithms make them an ideal choice for the automated design of CNN architectures.

By utilizing the power of evolutionary algorithms, NAS can effectively explore a vast number of alternative CNN designs and uncover viable options that may have been overlooked in the traditional manual design process. The potential benefits of this approach include increased speed, reduced costs, and improved performance in a range of applications, such as image recognition, natural language processing, and speech recognition.

Advantages of Automatic CNN Architecture Design

A.Efficiency

One of the most significant advantages of automatic CNN architecture design is efficiency. The process of designing a CNN architecture can be time-consuming and require significant expertise in the field of deep learning. Automatic architecture design techniques eliminate the need for manual design, reducing the time required to develop new models. Furthermore, automatic architecture design techniques can explore a vast search space of potential architectures, which is not feasible for humans to do manually.

B.Better Performance

Another advantage of automatic CNN architecture design is that it can often produce

better-performing models than human-designed architectures. Since automatic architecture design techniques can explore a vast search space of potential architectures, they can discover complex and non-intuitive architectures that humans may not consider.
This can lead to models with improved accuracy and generalization performance.

C. Domain-Specific Optimization

Automatic CNN architecture design techniques can also optimize architectures for specific tasks and domains. For example, NAS can optimize architectures for tasks such as image classification, object detection, and semantic segmentation. Similarly, automatic architecture design techniques can optimize architectures for specific domains, such as medical imaging, where the architecture must be optimized for the characteristics of medical images.

2.11 Evolutionary algorithms

Evolutionary algorithms, inspired by the elegant and complex process of natural selection and evolution in biology, are a remarkable class of optimization algorithms.The fundamental concept behind evolutionary algorithms lies in their ability to simulate the awe-inspiring process of natural selection. In nature, organisms with favorable traits survive and thrive, while weaker ones struggle to adapt and often fall by the wayside. Similarly, in optimization problems, the fitness of candidate solutions is evaluated by a fitness function based on some objective criteria, and the fittest individuals are more likely to survive and be selected for the next generation.
Evolutionary algorithms have been instrumental in solving a vast array of optimization problems, from engineering design and finance to data analysis and beyond. Two widely used examples of evolutionary algorithms are genetic algorithms (GA) and Gray Wolf Optimization (GWO), both of which have found diverse applications in numerous fields.

2.11.1 Genetic algorithm

The Genetic Algorithm (GA) is an evolutionary algorithm that mimics the natural selection process of genetics and evolution. The algorithm operates on the assumption that natural selection and survival of the fittest can be used to solve complex optimization problems. Just like in nature, the genetic algorithm applies principles of crossover, mu-

tation, and selection to search for the optimal solution to a given problem. The genetic algorithm operates on a population of potential solutions, each of which is represented by a set of parameters known as chromosomes. The algorithm begins by generating a random population of chromosomes, with the number of chromosomes determined by the size of the optimization problem. Each chromosome represents a potential solution to the problem, and its parameters are randomly assigned during initialization.
The GA's core operations are repeated iteratively until the algorithm converges to the optimal solution. The figure 2.5 describes the process in more detail:

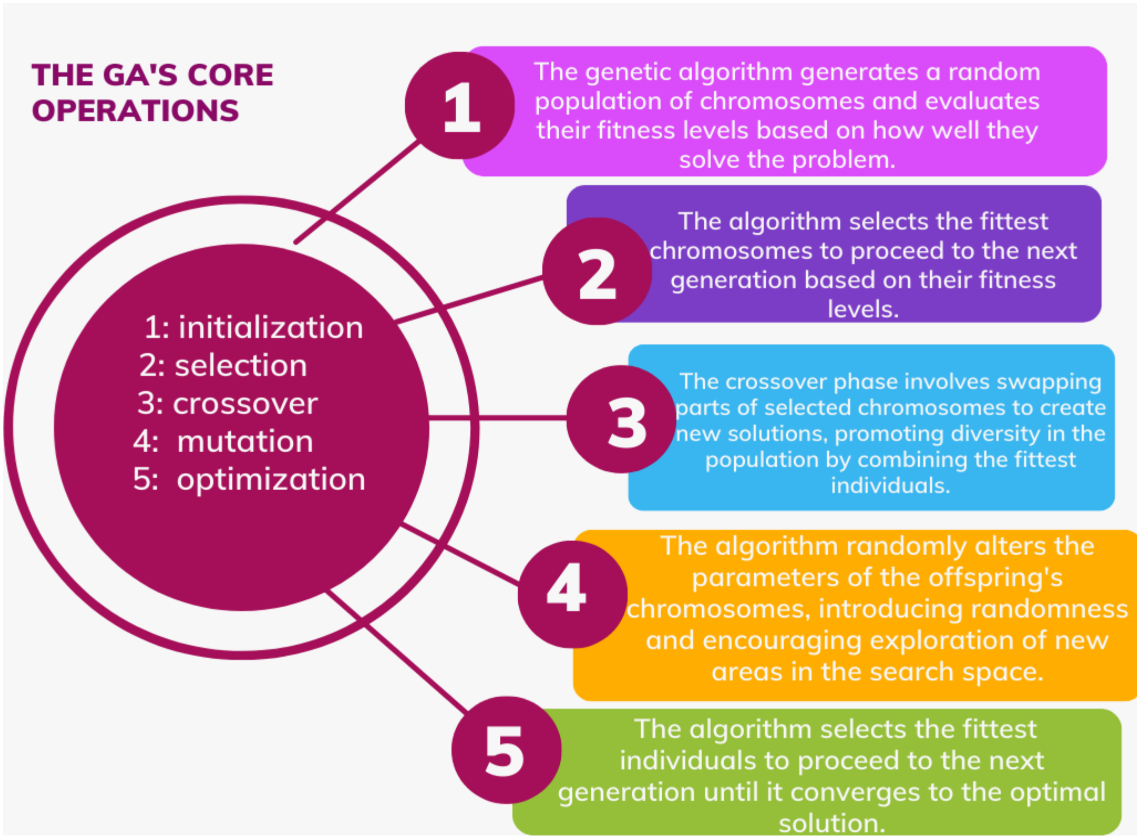

Figure 2.5: The GA's core operations

Genetic algorithms have also been extensively researched and developed, with several variants proposed in the literature to enhance their performance. These variants include the steady-state genetic algorithm [24], the adaptive genetic algorithm [25], and the genetic algorithm with elitist selection [26], among others. These variants aim to improve the efficiency and convergence speed of the genetic algorithm, which is a widely used optimization technique that mimics the process of natural selection and evolution. Genetic algorithms have been applied to various optimization problems in fields such as engineering, finance, and image processing. For example, genetic algorithms have been used for feature selection in gene expression data analysis [27], parameter optimization in machine learning models for image classification [28], optimal design of chemical processes [29], and solving complex scheduling problems [30]. The versatility and effectiveness of genetic algorithms make them a popular choice for solving optimization problems in a wide range of fields.

2.11.2 Gray wolf optimization

Gray Wolf Optimization (GWO) is a metaheuristic algorithm that simulates the social order and hunting behavior of gray wolves. The algorithm is based on the assumption that wolves coordinate their movements to hunt prey efficiently. The wolf pack has a hierarchical social structure, with each wolf assigned to a specific position and role. The wolf pack's social structure consists of an alpha wolf, beta wolf, delta wolf, and omega wolf, and each wolf has its own position in the pack as shown in figure 2.6 .

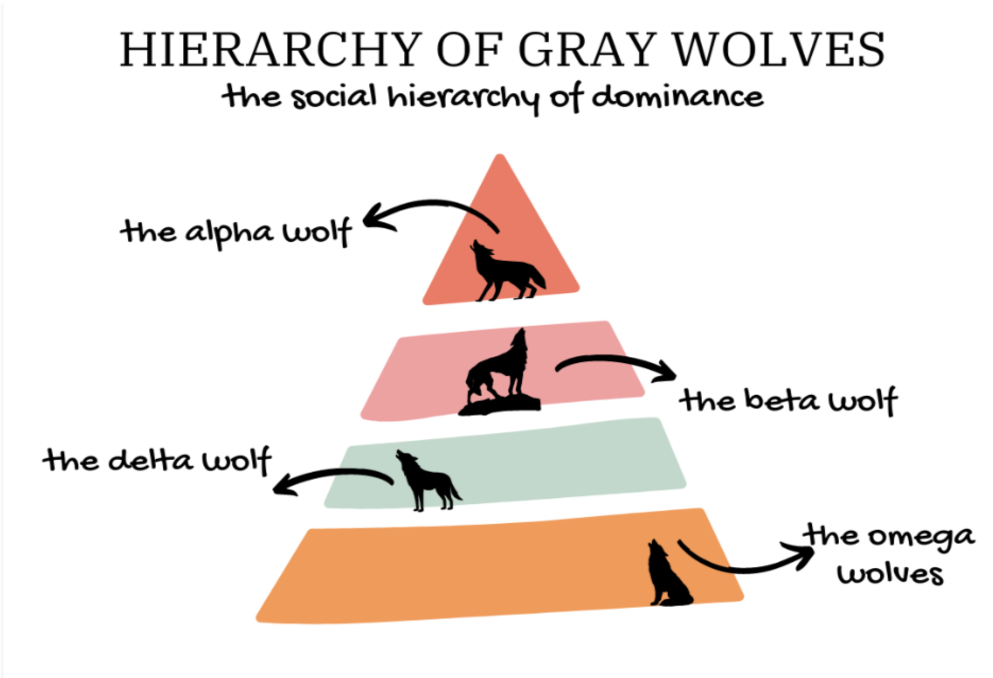

Figure 2.6: The GA's core operations

The alpha wolf is considered the pack's leader and commander at the top of the hierarchy. The alpha wolf is in charge of making decisions and guiding the pack's hunt expedition.

The beta wolf is the alpha's second in command, taking commands from the alpha and assisting in decision-making. The beta wolf also aids in hunting by tracking prey movements and detecting the weakest link in the prey chain. The delta wolf, ranked third in the social hierarchy, is in charge of organizing the omega wolves and keeping the pack in order.

The delta wolf also helps in hunting by striking the flank of the prey group, producing confusion and scattering the victims.

The omega wolves, who are regarded as a crucial component in the hunting process, are at the bottom of the hierarchy. The omega wolves provide backup and support to the alpha, beta, and delta wolves during the hunting phase. They also aid in the capture of the prey by attacking from behind and preventing the victim from escaping.

The GWO algorithm begins by initializing a group of potential solutions, which are represented by wolves in the search space. The alpha wolf is in charge of exploring the search space and keeping track of the best solution discovered thus far. The alpha wolf is followed by the beta and delta wolves, who aid in the exploratory phase. The omega wolves are a distinct group that assists in the hunting process by offering backup and support. GWO's hunting mechanism is based on how wolves coordinate their movements to hunt prey. The algorithm separates the search space into sub-spaces, each of which corresponds to a potential solution. The wolves work together to find the optimal solution in each sub-space. During the hunt, the wolves communicate with one another by howling, which represents the best answer found by each wolf.
figure 2.7 shows GWO's core operations .

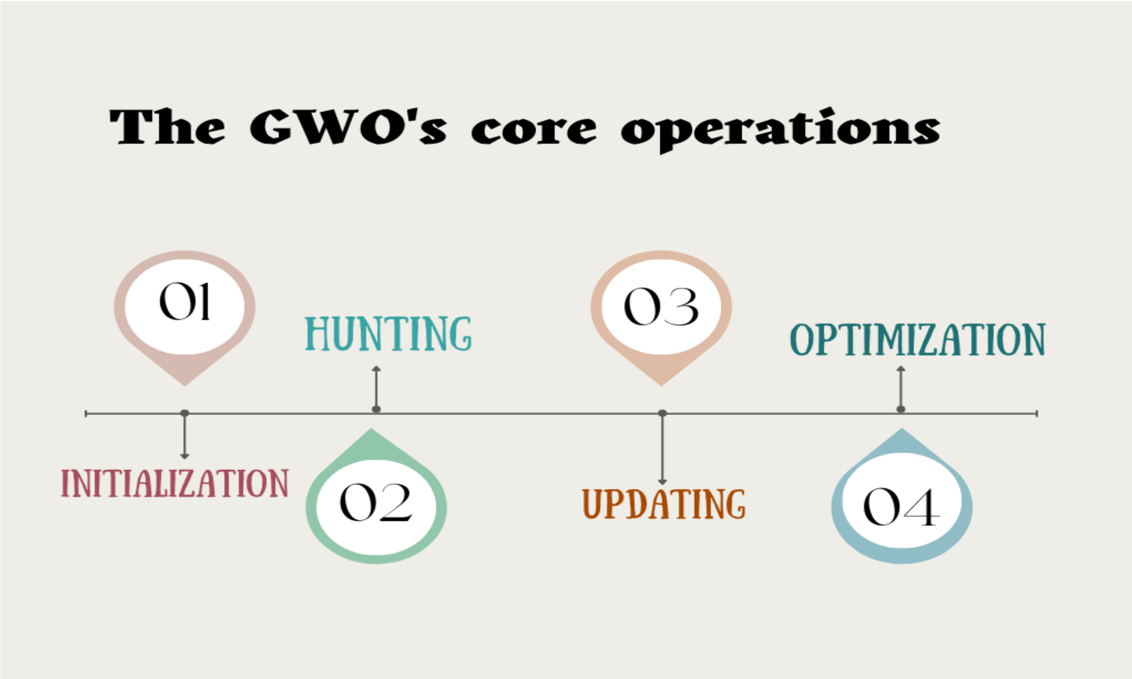

Figure 2.7: The GWO's core operations

Step 01 - Initialization

The Gray Wolf Optimization algorithm's initialization phase involves generating a random population of wolves, with the number of wolves determined by the size of the optimization issue. Each wolf symbolizes a potential solution, and its location in the search space is denoted by a vector. The wolves' initial placements are produced at random, and their roles in the algorithm are assigned based on their positions. Based on their positions in the search space, the alpha, beta, delta, and omega wolves are assigned their proper roles in the hierarchy.

Step 02 - Hunting

During the hunting phase, wolves search for the optimal option by subdividing the search space and coordinating their movements utilizing the surrounding effect method. They communicate with one another by howling in order to share the best answer discovered by each wolf. The alpha wolf guides the pack to the global optimum, while the beta and delta wolves aid in exploration and the omega wolves offer assistance. The best answer is passed on to the omega wolves for further development by the delta wolf.

Step 03 - Updating

During the updating phase, the algorithm changes each wolf's position in the search space and the best solution found thus far. The alpha wolf adjusts its position in relation to the best solution discovered thus far, while the beta and delta wolves adjust their positions in relation to the alpha wolf's position and the best solution discovered by the delta wolf. The positions of the omega wolves are updated based on the delta wolf's position and the best solution determined thus far.

Step 04 - Optimization

During the optimization phase, the algorithm refines the best solution obtained by GWO using an optimization approach such as gradient descent. The approach starts with the best answer found by the alpha wolf and then utilizes gradient descent to improve it further. The gradient descent technique iteratively refines the solution until it reaches the ideal solution.

These steps are shown in Figure 2.8 .

Figure 2.8: Hunting behavior of gray wolves: (A) chasing, approaching, and tracking prey (B-D) pursuing, harassing, and encircling (E) stationary situation and attack

Several variants of the GWO algorithm have been proposed in the literature, including the enhanced GWO algorithm [31] , the chaotic GWO algorithm[32] , and the GWO algorithm with opposition-based learning [33]. These variants aim to improve the search efficiency and convergence speed of the original GWO algorithm. In recent years, the GWO algorithm has been applied to various optimization problems in different fields, such as engineering, finance, and image processing. For example, the GWO algorithm has been used for feature selection in the classification of High-dimensional Biological Data [34], parameter tuning in a convolutional neural network classifier for skin cancer detection [35], and optimal sizing and placement of distributed generation systems in power systems [36], finding the shortest possible route in Traveling Salesman Problem [37].

In summary, GWO is a powerful optimization algorithm that mimics the hierarchical social structure and hunting behavior of gray wolves. The algorithm's hierarchical structure, hunting mechanism, and communication strategy enable it to efficiently explore complex search spaces and find high-quality solutions. The algorithm employs four main operations, including initialization, hunting, updating, and optimization, to find the optimal solution to the optimization problem.

2.12 Related work

In recent years, Convolutional Neural Networks (CNNs) have emerged as a popular and powerful approach for image classification tasks in various domains, including agriculture. CNNs have demonstrated outstanding performance in different areas, such as object detection, medical image analysis, digit recognition, and hyperspectral image classification, among others [38] [39] [40] [41].

Particularly, in the agricultural domain, CNNs have shown their effectiveness in tasks such as categorizing leaves, detecting unwanted plants, identifying plant diseases, and classifying fruits, among others [42]. Weeds are one of the main factors that affect crop production, and early identification and classification of weeds are crucial steps in successful weed management and control procedures in agriculture, especially in precision agriculture [43]. To distinguish weeds from the grass, authors in [44]carried out a classification task. Both ResNet and MobileNet were used to complete the classification task, which involved using both three and two image classes. ResNet architecture improved accuracy in both instances. In terms of accuracy, they achieved 95.6% accuracy for two image classes and 94.9% accuracy for three image classes using ResNet. Authors of [45] used a large, 17 509 image weeds dataset. Eight classes of images were formed after being gathered from various northern Australian regions. Higher accuracy for ResNet-50 was achieved after implementing CNN architectures. To take this further, [46] combined various datasets of weed species in the early stages of growth for classification. Using a CNN created from scratch, they achieved 86.2% CA for 22 species of plant seedlings. Similar to this, [47] used CNN to achieve up to 97.3% accuracy on two test sets. These examples show how CNN is effective at identifying weeds and how it can help advance the development of more effective and precise weed control techniques for agricultural use. several datasets on weeds have been proposed in the literature to help reduce the impact of weeds on crops while improving the weed management process [45] [48] [49]. In this context, the dataset proposed by [50] has been widely used in various studies. This dataset contains 5539 images of 12 different types of plants and weeds in their early stages, and it has been utilized in studies that have achieved high accuracy levels using different CNN-based approaches. For instance, in [51], transfer learning, model compression, and ensemble learning were used to obtain a 91.2% accuracy on this dataset. Similarly, [52] proposed a transfer learning strategy that utilizes a small network for detecting weeds among plant seedlings, which reached an accuracy of above 95%. In [53], four CNN architectures were used to discover the best-performing model for plant seedling classification, with ResNet-50 achieving the maximum accuracy of 96.21% on a test set. The ResNet architecture was also used in [54] to classify plant seedlings, with an accuracy of 85% with batch normalization and 83% without.

In [55], a system for categorizing weeds and crops employing five pre-trained CNNs achieved a ResNet50 testing accuracy of 95.23%. According to a comparison study in [56], Deep CNNs can successfully categorize crops and weeds, with ResNet152V2 achieving the best accuracy of 92.93% when fine-tuned. In [57], a classification framework based on three distinct deep CNN architectures was proposed, with EfficientNetB0 having an accuracy of 96.52%. In [58], the authors explore the use of IoT and digital image processing for weed detection and classification in the agriculture industry, achieving a test accuracy of 94.20% using the Weed-ConvNet model. Moreover, in [59], a deep-learning architecture fine-tuned for greater processing time and low memory consumption achieved an overall accuracy of 90.15% for classifying plant seedling images. In [60], a new CNN architecture using deep learning techniques for early weed detection in crops achieved an average classification accuracy of 94.38%. Finally, in[61] , a model for classifying crops and weeds images using transfer learning and data augmentation techniques achieved a validation accuracy of 96.04% on the test set. In addition to the use of pre-defined CNN architectures, the automatic CNN architecture search approach has shown promising results in achieving state-of-the-art performance on image classification tasks in agriculture. In [62], a target-dependent Bayesian optimization algorithm was used to automatically design CNN architectures for plant identification and achieved an accuracy of 92.31%. The authors in [63] proposed an approach that utilizes the Fuzzy C-Means-based Chameleon Swarm Algorithm for segmentation, a fast GLCM model for feature extraction, and Progressive Neural Architecture Search (PNAS) for Plant Disease Classification. This approach achieved an accuracy of 97.43%. In conclusion, CNNs have shown remarkable performance in various agricultural image analysis tasks, including weed detection, crop classification, and yield estimation. With the increasing availability of large-scale datasets and computing power, CNNs are expected to play an increasingly important role in improving the efficiency and accuracy of agricultural practices. Furthermore, the recent advances in automatic architecture search and transfer learning offer promising avenues for improving the performance of CNNs in agriculture-related tasks, enabling precision agriculture and sustainable food production.

2.13 Conclusion

This chapter provides a comprehensive overview of the background of deep learning, its applications in agriculture with a focus on convolutional neural networks (CNNs), the concept of neural architecture search, and highlights notable research efforts in the domain of weed management. By exploring these topics, we aim to enhance our understanding of the potential and advancements in utilizing deep learning techniques for optimizing agricultural practices.In the next chapter we will present our approch .

Chapter 3

Our proposed framework

3.1 Introduction

Convolutional neural networks (CNNs) have shown significant potential in solving computer vision problems, but developing an optimal architecture for a specific problem can be challenging and requires expertise and manual trial and error. In this chapter, we will present a solution to help overcome this problem. Our solution consists of two approaches: the first approach involves creating a CNN from scratch using blocks, and the second approach involves using a pre-trained model.

Given that evolutionary algorithms have shown exceptional results in different fields, we utilized two specific algorithms, namely Genetic Algorithm and Gray Wolf Optimization, in our research. We used these algorithms to optimize the architecture of the CNNs, both when building the CNN from scratch and when fine-tuning a pre-trained model. By using these optimization techniques, we aim to develop more efficient and accurate models for the specific problem at hand.

3.2 Dataset

The proposed work utilized the Plant Seedling dataset [50], which was made accessible by Aarhus University in partnership with the University of Southern Denmark. The second version of the dataset, which comprises 5539 images of 12 common plant species found in Danish agriculture, was utilized, while the initial version only had 4275 images. The images were taken several times over the course of 20 days, at 2 to 3-day intervals,

beginning immediately after emergence. The information is intended to aid farmers in weed management by recognizing plant species at an early development stage.
The dataset is divided into an 80:20 ratio for training and testing, with 80% of the data utilized for training and the remaining 20% for testing. Image augmentation and segmentation were used prior to training the network. The distribution of images in each of the 12 classifications is shown in Table 3.1, with Loose Silky-bent having the most and Common Wheat having the fewest. Each class contains colored pictures of plants at various phases of development in various sizes and in png format. Figure 3.1 depicts a sample picture from each class.

class	**Species**	**Training image**	**Test image**
1	Sugar Beet	370	93
2	Black grass	247	62
3	Charlock	361	91
4	Cleavers	268	67
5	Common Chickweed	572	143
6	Common Wheat	203	50
7	Fat Hen	430	108
8	Loosy Silky-bent	609	153
9	Maize	206	51
10	Scentless Mayweed	485	122
11	Shepherd's purse	219	55
12	Small-flowered cranesbill	461	115

Table 3.1: Distribution of images in the V2 Plant Seedling Dataset

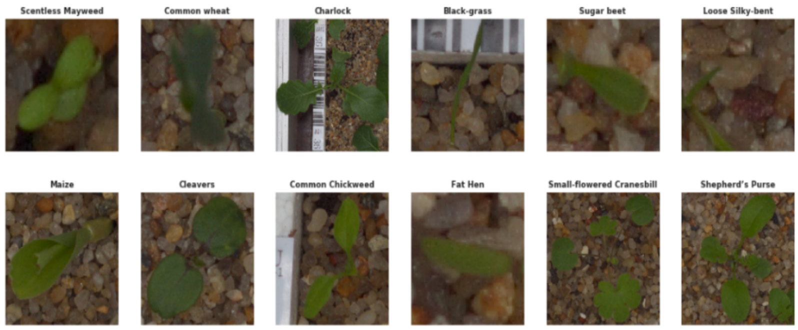

Figure 3.1: Random samples of each image class in dataset

3.3 Our proposed framework

The overall framework established to implement the proposed solution for solving the CNN design problem is divided into two phases: the preprocessing phase and the optimization phase. In the preprocessing phase, the dataset undergoes a pipeline to ensure its readiness for the optimization phase. This phase involves transforming the dataset into a format that can be effectively analyzed in the subsequent phase. Additionally, new images are generated to enhance the model's performance and mitigate over fitting.

The second phase utilizes evolutionary algorithms, specifically Gray Wolf Optimization and Genetic Algorithm, to optimize the CNN architecture design and provide a customized solution for the selected dataset. Figure 3.2 illustrates the overall research procedure diagram.

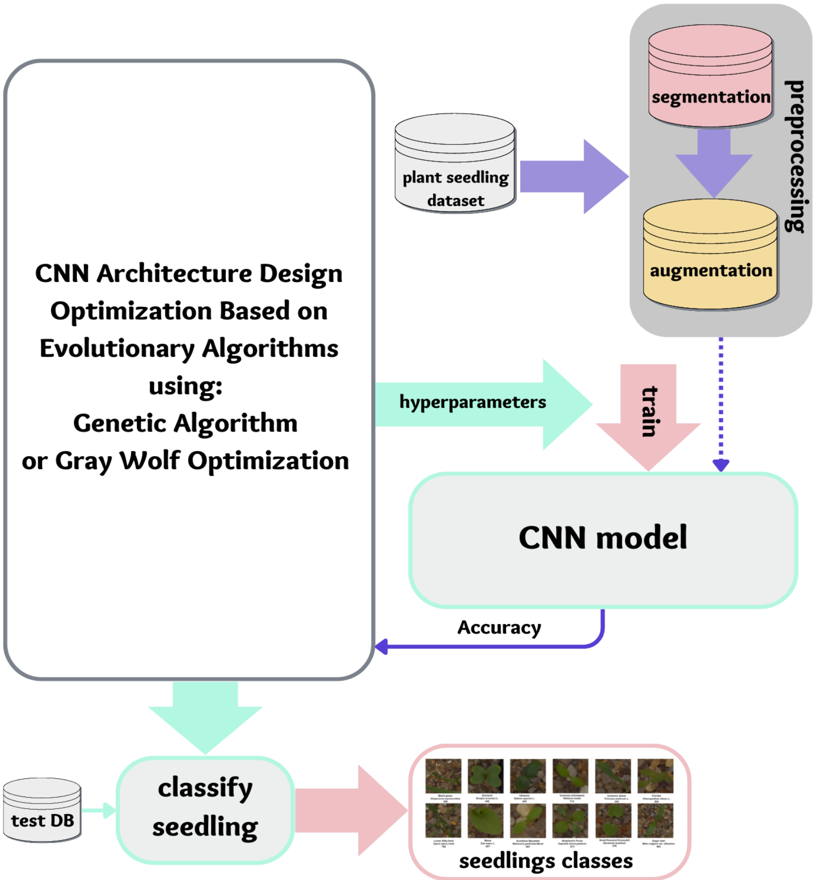

Figure 3.2: The overall framework of our proposed approach

3.3.1 The preprocessing phase

To develop useful and powerful models using our proposed methods, it is critical that our models can learn and extract features from images without external distractions and noise that can degrade their performance. Furthermore, our models must be able to perform well in real-world situations, regardless of factors such as picture angle, lighting conditions, or other external factors that may make seedling classification difficult. To solve this issue, we applied a pipeline to preprocess images, which is divided into two phases: image segmentation and data augmentation. By applying these preprocessing techniques, we aim to improve the performance and generalization ability of our models, allowing them to effectively classify seedlings in various real-world conditions.

A.Data segmentation:

The first phase of our pipeline involves image segmentation, which helps to isolate the regions of interest in the image and remove any unnecessary background noise. This is achieved through techniques such as thresholding, edge detection, and contouring. In this work initially, we utilized Gaussian Blur to reduce high-frequency content and achieve a smoother image. Next, we converted the blurred image to the HSV color space. To capture the seedlings, we created a mask by defining a range of potential color values. We employed morphological erosion with an 11x11 structuring kernel to produce foreground seedling images with subtracted backgrounds. Subsequently, we obtained a subset of seedling images before and after background subtraction, as depicted below.

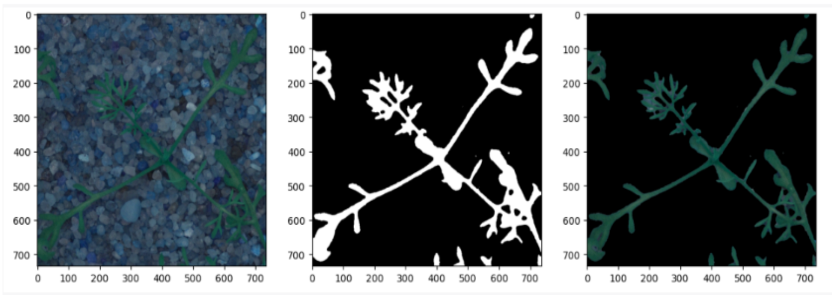

Figure 3.3: Background segmentation result

B.Data augmentation:

The second phase of our pipeline is data augmentation, which involves generating new training data by applying transformations to the original images, such as rotation, translation, scaling, and flipping. This helps to increase the variety of images available for training, thereby improving the model's ability to generalize to new data. In this phase, we applied several techniques to generate new training data and increase the variety of images available for training. These techniques include:

- **Rescale:** rescales the pixel values of the image to a range between 0 and 1.
- **Rotation range:** randomly rotates the image within a specified range of angles.
- **Width and height shift range:** randomly shifts the image horizontally and vertically within a specified range.
- **Zoom range:**randomly zooms in or out of the image.
- **Vertical and horizontal flip:** flips the image vertically and horizontally.

Additionally, we applied featurewise centering and featurewise standard normalization to ensure that the training data has zero mean and unit variance, which helps improve model performance.

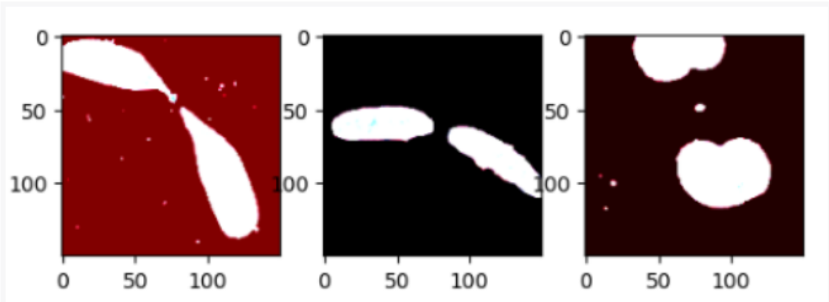

Figure 3.4: Image augmentation results.

3.3.2 The optimization phase:

In this work, we propose an automatic CNN architecture design approach that leverages the power of Evolutionary Algorithms, namely Gray Wolf Optimization (GWO) and Genetic Algorithm (GA). However, both GWO and GA share a common preliminary step that needs to be undertaken before initiating any optimization process. This step involves encoding the solutions (CNN architectures), which is an integral part of both optimization methods. To avoid redundancy within the algorithm procedures, we will implement this encoding step separately, as a pre-step to GWO and GA. Subsequently, we will explain the specific procedures of each optimization algorithm.

Prestep: Encoding system:

Since we are utilizing two distinct evolutionary algorithms, the encoding step holds significant importance as it greatly influences the effectiveness of both Genetic Algorithm (GA) and Gray Wolf Optimization (GWO). In light of this, we have opted to employ two different encoding systems, aiming to explore alternative approaches in this phase.
Our intention was to extend beyond the conventional methods and harness the capabilities of established techniques that have proven their efficacy. As a result, we selected transfer learning as the foundational element for the first type of encoding system, leveraging its strengths and incorporating it into our approach. For the second encoding system, we

adopted the concept of blocks as the primary building blocks, offering a different perspective on the encoding process. Our purpose behind this decision was to leverage the strengths and advantages offered by transfer learning and the concept of blocks.

i. **Bloc encoding system:** Before proceeding with the encoding process, it is crucial to identify the CNN hyperparameters that we intend to optimize in order to create an improved representation of the CNN architecture. We acknowledge the significance of the hyperparameters and their values in determining the performance of the CNN architecture. These chosen hyperparameters are outlined in Table 3.2.

Hyperparameter	**Range**
# Number of bloc	[3 - 8]
# filter number	[64–512]
Learning rate	[0.00001, 0.0001, 0.001, 0.01]

Table 3.2: CNN architecture hyperparameters and values.

for the bloc encoding system, we employ blocks similar to residual blocks to construct the feature extractor. In this system, the fully connected layer remains fixed. Each block consists of two convolutional layers: the first with a kernel size of 3 and the second with a kernel size of 1. Additionally, it includes a pooling layer with a pool size of 3 and a stride of 2, batch normalization, and the ReLU activation function. As shown in Table 3.2, the hyperparameters we aim to optimize are the number of blocks per CNN, the number of filters in each block, and the learning rate for each CNN.

To encode the CNN architecture, we employ a set of hyperparameters outlined in Table 3.2. This set represents a potential solution for the CNN architecture design. The architecture itself is represented by a vector composed of three hyperparameters. An example of a randomly generated CNN architecture is illustrated in Figure 3.5 . It is crucial to note that the hyperparameter values within the vector are restricted to the ranges defined in Table 3.2. This constraint ensures that the resulting CNN architecture is both effective and feasible for our research objectives.

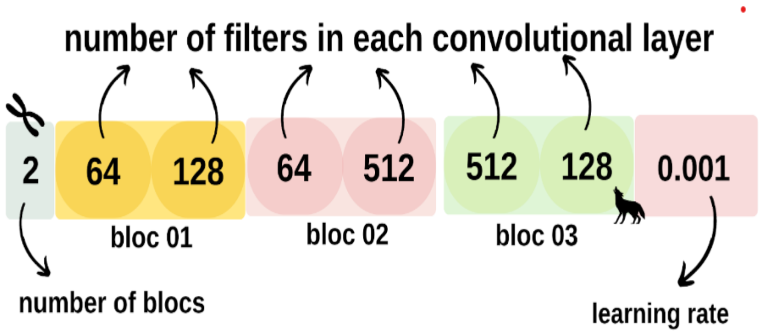

Figure 3.5: An example of the proposed encoding strategy representing a CNN.

ii. **Transfer learning encoding system** For the chosen encoding system, we have identified the relevant hyperparameters along with their respective values, which are presented in Table 3.3.

Hyperparameter	**Range**
pre-trained models	[Exception, InceptionV3, MobileNet, ResNet50, DenseNet121, InceptionResNetV2]
# Dense bloc	[1–3]
# Neurons for each dense layer	[64–1024]
# dropout rate for each dropout layer	[0 - 0.5]
Learning rate	[0.00001, 0.0001, 0.001, 0.01]

Table 3.3: CNN architecture hyperparameters and values.

In the transfer learning (TL) encoding system, as described in Table 3.3, we utilize a pre-trained model and a specific number of dense blocks. Each block consists of a dense layer and a dropout layer. The hyperparameters to be optimized by evolutionary algorithms (EAs) include the number of neurons and the dropout rate for each block. Our objective in this system is to identify the optimal pre-trained model and fine-tune its hyperparameters to suit our dataset.

To encode the CNN architecture, we employ a set of hyperparameters outlined in Table 3.3. This set represents a potential solution for the CNN architecture design. The architecture itself is represented by a vector composed of five hyperparameters. An example of a randomly generated CNN architecture is illustrated in Figure 3.8. It is crucial to note that the hyperparameter values within the vector are restricted to the ranges defined in Table 3.3. This constraint ensures that the resulting CNN architecture is both effective and feasible for our research objectives.

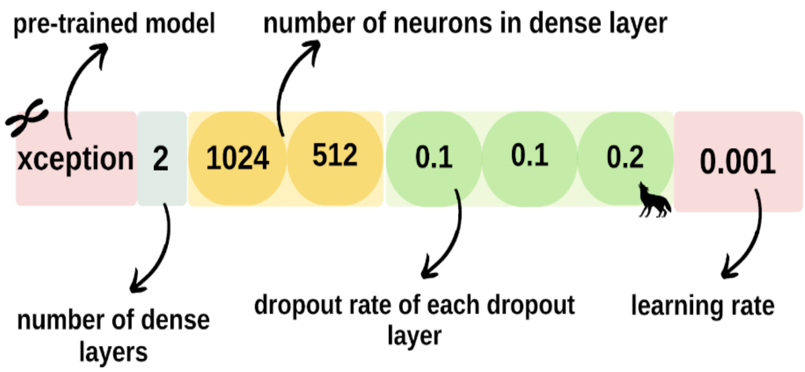

Figure 3.6: An example of the proposed encoding strategy representing a CNN.

1. **Gray wolf Optimization**

   Gray Wolf Optimization, a newly developed evolutionary algorithm, is regarded as a potential optimization technique. It is one of the few optimization approaches that is aimed at calculation optimization, making it better suited to the problems at hand. Furthermore, as a relatively new method in the field of optimization, this approach has a lot of untapped potential.

   The algorithm simulates the hunting mechanism of gray wolves, which is one of the most fascinating phenomena in the real world. Gray Wolf Optimization has already demonstrated incredible results, surpassing most other algorithms and providing a better answer to numerous problems. As a result, it is considered to be a better alternative than some of the more well-known optimization techniques.

   The overall procedure and how the GWO algorithm was used to solve the problem are shown in our proposed diagram (Figure 3.7). Before starting the optimization process, a few parameters must be set, including the wolf pack size, which is set to 10, and the number of iterations, which is set to 15.

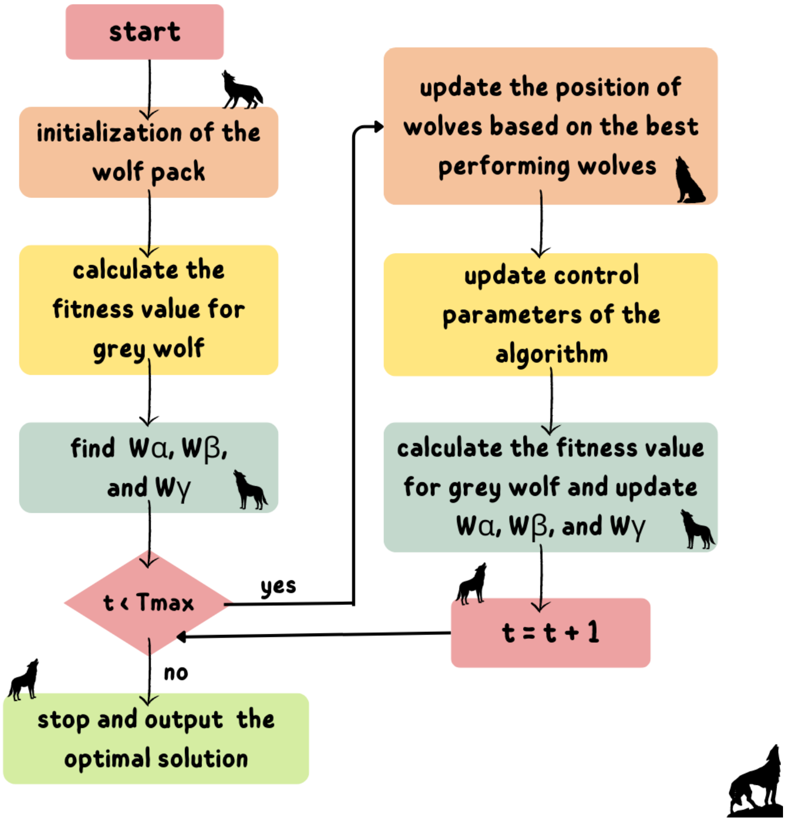

Figure 3.7: Proposed GWO–CNN architecture optimization procedure.

A) **Wolf pack initialization:**

The gray wolf is one of the communal species out there in the wilderness as they live and hunt as a pack. So, the first step in our method is to try to mimic this behavior and create our initial wolf pack (initial solution). In this matter, we need two main things. First, we need to identify the encoding system followed by its corresponding hyperparameters that will give us a clear structure to use to represent each wolf in the pack and how they will be handled in future steps. Second, we need to decide on the number of wolves representing the pack. The larger the number, the more diverse the solution, and the better our chance of getting to the optimal solution. As mentioned before, we have chosen a size of 10 for the wolf pack as it balances diversity and resource consumption. These wolves will be our starting point in the process.

Figure 3.8: An example of the initial wolf pack using the TL encoding system.

B) **Evaluate the wolf pack and identify the Alpha, Beta, and Gama wolves:**

As mentioned before, gray wolves have a hierarchical system within the pack, where the wolves are ordered as alpha, beta, gamma, or omegas. To follow and use this hierarchy in our work, we first need to choose a metric based on which we will organize our pack. Additionally, we need to track the progress and evolution of our algorithm over time. In our case, since our wolves are CNN architectures, we have determined that the best evaluation metric to use to evaluate our wolves is accuracy, as it measures the effectiveness of our solution. For accuracy, we will use the following equation:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

After defining the evaluation metric, and as a pre-step to organizing our wolf pack, we need to evaluate each member. This will be done by training them using the preprocessed training dataset (80% of the main dataset) and then evaluating them using the evaluation data (the remaining 20%) in each epoch. We will choose the best accuracy from the evaluation process as the final value of our wolves' accuracy.

This brings us to our goal of organizing the wolf pack to follow the hierarchy system. We will order the wolves in the pack and identify the main three wolves in our algorithm: alpha as the CNN architecture with the best accuracy, beta as the CNN architecture with the second-best accuracy, and gamma wolves as the CNN architecture with the third-best accuracy. These three main wolves will be used as a guide for the other wolves in the search space during the hunting process. Keep in mind that these wolves will be inspected and updated if necessary after each minor change within the wolf pack.

C) **Updating wolves:**

In the hunting mechanism, the wolves update their positions in each step according to the main solutions alpha, beta, and gamma, ensuring that they are always closer to the prey than the other wolves. To ensure the effectiveness of the process, we need to add a phase to our scheme where we update the position of all wolves, one member at a time. For each pack member, we will use alpha, beta, and gamma to coordinate their next position, using a mathematical technique similar to crossover but with some differences related to our encoding system. This will improve the quality of each wolf and the entire wolf pack, by exchanging information between the leading three wolves and the rest of the pack. We will use the following equation for the crossover method, with minor changes related to our encoding system.

$$new\_wolf = |\frac{W_\alpha - A^*|C^*W_\alpha - W|W_\beta - A^*|C^*W_\beta - W| + W_\gamma - A^*C^*|W_{\alpha.learn\,rate} - W_{.learn\,rate}}{3}| \quad (3.2)$$

This equation requires a set of parameters that can be calculated using the following formulas:

$$a = 2 - 2^{*}\frac{iteration\ number}{max\ number\ of\ iteration}$$

$$r1, r2 = random\ number$$

$$A = 2^{*}a^{*}r1 - a$$

$$C = 2^{*}r1$$

- **Bloc system**

  In this method of encoding, we limit the number of blocks to ensure variation and avoid getting stuck in a local optimum. We utilize the equation *new_wolf* and round the result to suit the hyperparameter for other hyperparameters, such as the number of filters. We use the equation as is for the learning rate, with no adjustments.

- **TL system**

  We have a combination of hyperparameters in this sort of encoding. The fully connected layers and learning rate are computed using the *new_wolf* equation, with a little adjustment for the number of neurons. The equation's result is then rounded to fit the nature of the parameter. For the pre-trained model, the new wolf will have the same pre-trained model as the current wolf. This helps preserve the diversity of our solution and ensures that the GWO won't get stuck on a local optimum. After performing a crossover for each member, a new wolf is created. This new wolf is then evaluated using the evaluation method described in the step above. If the accuracy of the new wolf is better than that of the old one, the old wolf is replaced by the new one in the pack. The presence of the new wolf may require changes in the hierarchy of the pack, including the positions of the alpha, beta, and gamma wolves. Therefore, the entire wolf pack must be re-evaluated to determine if the new wolf necessitates any changes in the hierarchy. This process is repeated every time a new wolf is created. an example of this process using the TL encoding system is shown in figure 3.9.

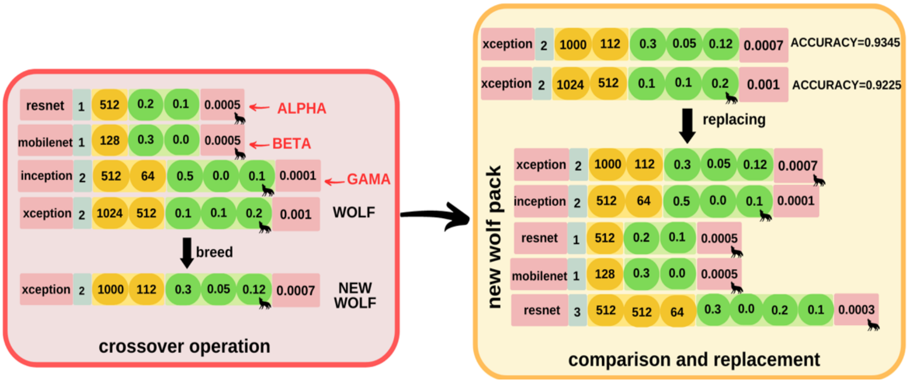

Figure 3.9: An example of the wolves updating process using the TL encoding system.

D) **Stop criteria** After reviewing the entire pack and guiding them through the crossover as an initiation to improve their position, this entire process will be considered as one iteration. The process will continue by updating the wolf pack again and again until the termination condition is met. In the gray wolf algorithm, this condition is typically achieving a certain prey or goal, but in our case, we have chosen a specific number of iterations for the algorithm to go through before it stops. As mentioned above, our termination condition is set at 15 iterations.

2. **Genetic algorithm**

The genetic algorithm is a promising and popular evolutionary algorithm that has been widely employed for solving various optimization problems. It consistently provides solutions that are superior to traditional approaches, making it a preferred choice even today. The genetic algorithm is inspired by Darwin's theory of evolution, which aims to create better species while leading to the extinction of less fit ones based on the environment.

Moreover, the genetic algorithm is considered a classic optimization algorithm due to its effectiveness and wide applicability. In our work, we utilized the genetic algorithm, and the complete procedure we developed using this algorithm is illustrated in Figure 3.10. Before initiating the process, it is crucial to initialize the necessary parameters for the genetic algorithm, which will be utilized throughout our procedure. These parameters consist of:

- Population size: 10 chromosomes
- Probability of mutation: 0.1 (10%)

- Probability of crossover: 0.8 (80%)
- Number of crossover pairs: 3 pairs (6 chromosomes)
- Number of offspring: 6 chromosomes
- Number of generations: 15

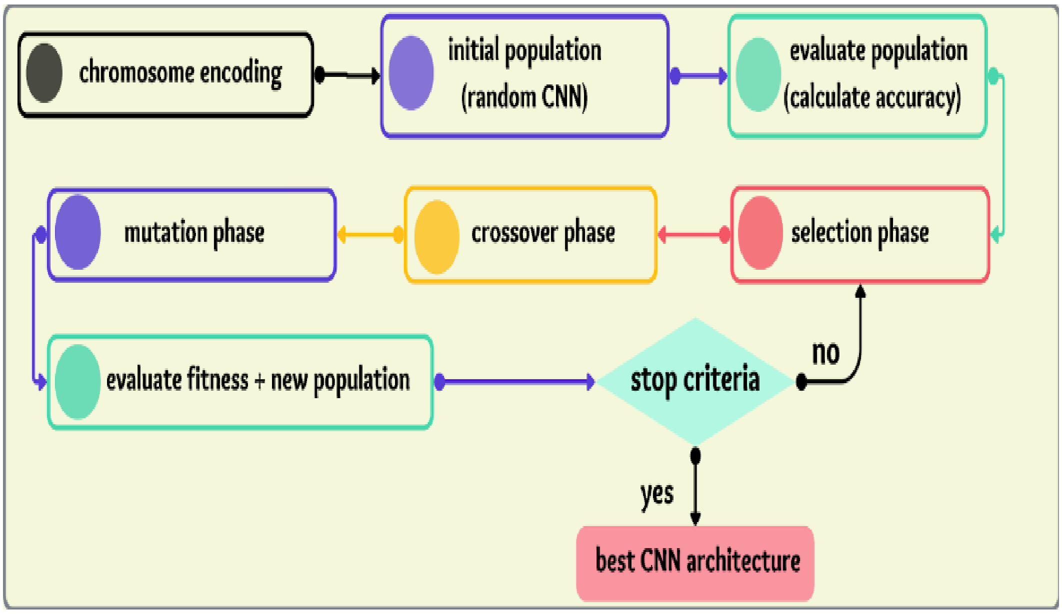

Figure 3.10: Proposed GA–CNN architecture optimization procedure

A) **Initialization**

One crucial aspect of the evolution theory is the presence of a population of species that undergoes the algorithm's steps. Therefore, it is essential to establish a population of solutions and ensure its evolution towards better and more fit ones.

Prior to any initialization, it is necessary to determine the encoding system and its associated hyperparameters. Additionally, the initial population size significantly influences the functioning of the genetic algorithm. A larger population size promotes a greater diversity of solutions, thereby increasing the chances of reaching an optimal solution. To strike a balance between diversity and resource utilization, we chose a population size of ten. Subsequently, we generated ten chromosomes (representing CNN architectures) using the selected encoding scheme as our initial starting point. Figure 3.11 provides an illustration of a sample random chromosome population.

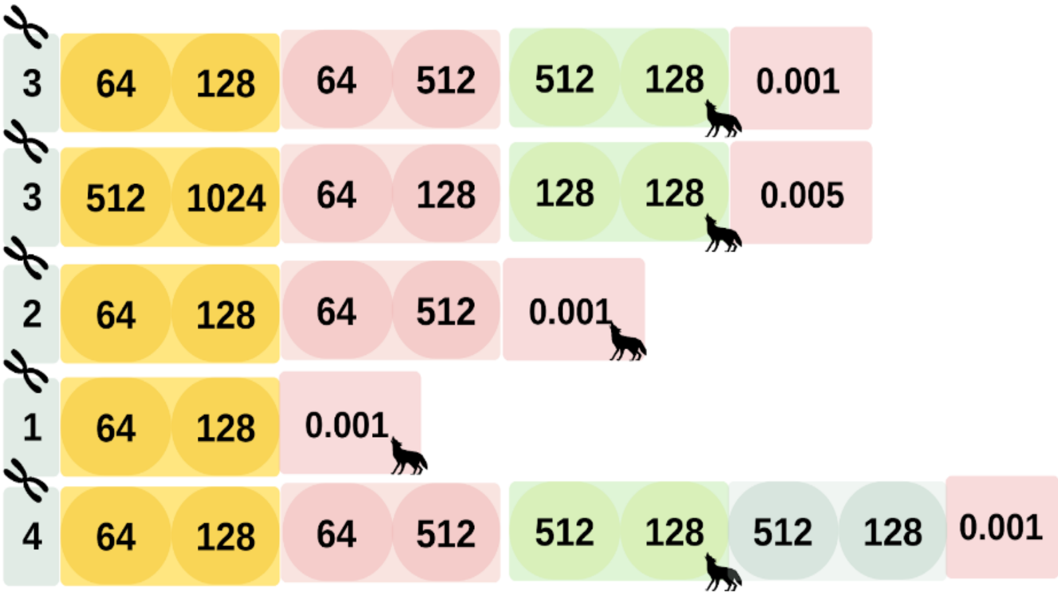

Figure 3.11: An example of the initial wolf pack using the bloc encoding system.

B) **Fitness**

In order for our genetic algorithm to be effective, we need quantitative measures to assess its progress. In our case, we have chosen the accuracy of the architecture as an indicator of the solution's effectiveness. We will calculate the solution fitness using the following equation:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.3}$$

After defining the evaluation metrics, the second step in the genetic algorithm is to evaluate the chromosomes. We achieve this by training them on the preprocessed training dataset, which constitutes 80% of the main dataset, and then assessing them using the evaluation data, which is the remaining 20%. We consider the highest accuracy obtained during the evaluation process as the ultimate value of the architecture's overall accuracy.

C) **Selection**

The creation of new and more fit individuals is a crucial step in the evolutionary algorithm. Therefore, it is essential to carefully select individuals that exhibit higher fitness. Consequently, after assessing the fitness of each chromosome in the population, we ordered them in descending order based on their fitness scores. During this phase, the selection of individuals for mutation and crossover is necessary. There are various selection techniques available, including lucky fortune selection, random selection, and rank selection, among others. In our study, we opted for the rank selection method, which involves selecting the best chromosomes based on their fitness scores. Using this approach, we chose six individuals to undergo crossover and mutation operations, resulting in the creation of six offspring.

D) **Crossover**

This phase stands out as the most crucial stage within the entire genetic algorithm process, as it drives the real change and evolution within each population. It plays a pivotal role in propelling the algorithm towards improved search territories, making it the core component responsible for the effectiveness of genetic algorithms.

In the preceding phase, we have already selected six individuals to undergo the crossover process. In this subsequent phase, we evaluate each chromosome to determine its inclusion in the crossover process. To accomplish this, we randomly generate a number between 0 and 1 and compare it to the crossover probability, which, in this case, is 0.8. If the generated number is less than 0.8, the corresponding chromosome is selected for crossover. We repeat this process until we obtain three pairs of chromosomes. Finally, the genetic information of each pair is exchanged, facilitating the crossover operation.

- **Bloc system** In this type of encoding, we fix the number of blocks to ensure solution diversity and avoid getting stuck in local optima. We randomly select a number less than the number of blocks, which determines the number of swaps between individuals. In each swap, we randomly choose a position in the individuals and exchange their values.

Figure 3.12: An example of the crossover phase using the Bloc encoding system.

- **TL system** In this type of encoding, we use the same method as the block system, except that we fix the number of fully connected layers instead of blocks. We randomly select a number less than the chromosome length, which determines the number of swaps between individuals. In each swap, we randomly choose a position in the individuals and exchange their values.

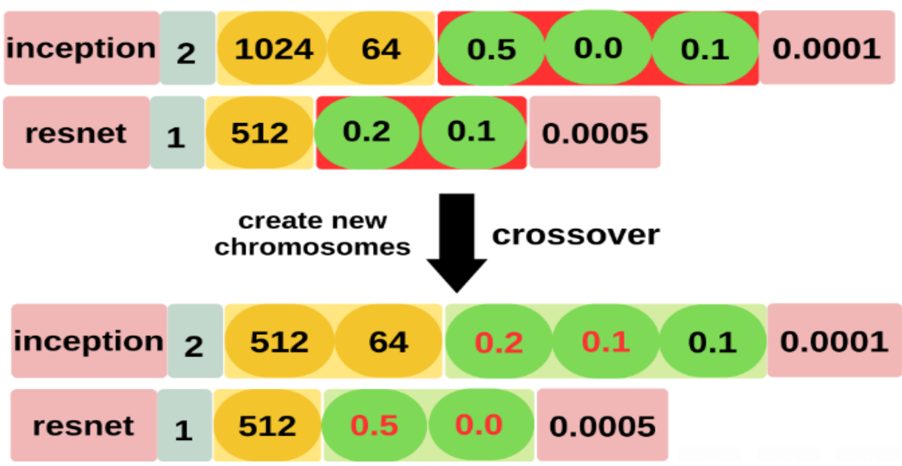

Figure 3.13: An example of the crossover phase using the TL encoding system.

E) **Mutation** In this phase, we aim to introduce random changes to the chromosomes in order to explore and expand the search space, with the goal of discovering new territories that may potentially contain the optimal solution. The mutation phase in our process encompasses the following steps: Firstly, we generate six offspring from the previous step, and these offspring will be subjected to the mutation phase. The decision to mutate a chromosome is based on the probability of mutation, which we have set to 0.1 (10%) in our case. For each chromosome in the offspring list, we randomly generate a number between 0 and 1. If the generated number is less than 0.1, the chromosome will undergo mutation. When a chromosome is selected for mutation, we randomly choose a gene (hyperparameter) from the chromosome and replace it with a new one obtained randomly from the search space corresponding to that gene. Subsequently, the mutated chromosome replaces the original one in the offspring list.

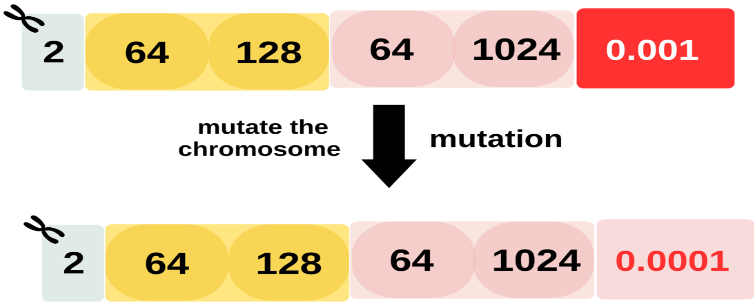

Figure 3.14: An example of the mutation phase using the Bloc encoding system.

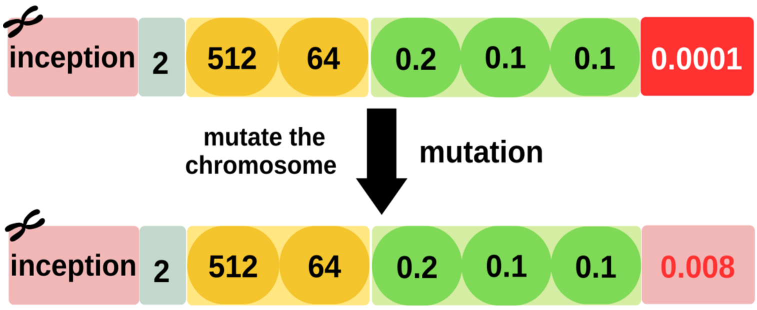

Figure 3.15: An example of the mutation using the TL encoding system.

F) **Evaluation and selection of the new population** If we have reached this phase, we already have the main population that we started the cycle with, in addition to the newly created offspring list. In this phase, there are two main steps to follow: evaluation and selection.
In the evaluation step, we define our evaluation metrics, which in this case is the accuracy of the CNN architectures (chromosomes) by training and evaluating them on our chosen dataset. We follow the same steps as in the evaluation phase to evaluate the main population and the offspring list. After the evaluation step, we move to the selection step, which is one of the most important steps in the genetic algorithm cycle. It ensures the survival of the fittest solutions and the extinction of the least fit and bad ones, allowing the algorithm to progress toward a better and more efficient solution. In this step, we rank the population and offspring in descending order based on their fitness. Then, the best chromosomes are selected as our new population. In our case, we select 10 individuals to represent our new generation of the population that will continue forward in the cycle.

G) **Stop criteria** This step will be the last step in our process of deciding if the cycle will continue forward or if it will stop here. As the process repeats itself starting from step 3 where the selection phase begins, preparing for a new generation, until the termination condition is met where it can be a certain number of iterations or to get to a certain accuracy . and for this matter we chose the first option which is 15 generation as it shows a good balance between result and resource consumption.

3.4 Conclusion

This chapter presents a solution to developing optimal architectures for convolutional neural networks (CNNs) in computer vision tasks. Two approaches are proposed: building a CNN from scratch using blocks and utilizing a pre-trained model. The first approach allows flexibility and customization in designing the network architecture based on specific problem requirements. Using evolutionary algorithms like Genetic Algorithm and Gray Wolf Optimization, the architecture is optimized, reducing reliance on manual trial and

error. The second approach leverages pre-trained models, trained on large-scale datasets for general visual recognition tasks, to fine-tune the network architecture.
This process improves the accuracy and efficiency of the pre-trained model, allowing it to adapt to the specific problem domain. By utilizing these algorithms, the research contributes to the advancement of CNN-based models, enabling more efficient and accurate solutions to various computer vision problems.

Chapter 4

Experiments, reults, and discussion

4.1 Introduction

In this chapter, we delve into the experimental results and analysis of our research on automatic convolutional neural network (CNN) design using genetic algorithm and gray wolf optimization. Building upon the methodology outlined in Chapter 3, we present the outcomes of our proposed approaches and provide an in-depth discussion of the obtained results. This chapter serves as a comprehensive evaluation of the effectiveness and efficiency of our novel method, shedding light on its performance in comparison to the existing state-of-the-art approaches.

4.2 Experiments configurations

The proposed approach was designed, trained, and validated using Kaggle platform utilizing Keras running on top of TensorFlow, we used GPU p100 for our experimentations. Each algorithm of the already 4 proposed frameworks in chapter 03 was used to generate a suboptimal CNN architecture capable of correctly classifying our dataset. Each approach was run for 15 iterations. Each CNN architecture generated by one of the 4 approaches was trained for a fixed number of epochs that equals 50 epochs using Ada grad optimizer and a batch size of 20.

4.3 Experimental material and platforms

4.3.1 Kaggle

Kaggle is a platform for data science and machine learning that offers access to powerful computing resources like GPUs and TPUs. It provides a diverse collection of datasets for practice and competition, attracting a community of data scientists and researchers. Kaggle enables users to collaborate, learn, and compete by exploring datasets, building models, and sharing insights. Its GPU and TPU support allows for accelerated computation, empowering users to tackle large-scale machine learning tasks efficiently.

4.3.2 Keras

Keras is a widely-used high-level neural network library in Python that provides a user-friendly and intuitive interface for designing, training, and evaluating deep learning models. It offers a comprehensive set of pre-defined layers, activation functions, and optimization algorithms, allowing researchers and developers to easily construct complex neural architectures. Keras seamlessly integrates with TensorFlow, leveraging its powerful computational capabilities for efficient execution on CPUs and GPUs. With its simplicity and flexibility, Keras has become a popular choice for both beginners and experienced practitioners in the field of deep learning. Its extensive functionality, combined with its compatibility and ease of use, make it a valuable tool for a wide range of machine-learning tasks.

4.3.3 TensorFlow

TensorFlow is an open-source machine learning framework developed by Google. It provides a flexible platform for building and deploying deep learning models. TensorFlow uses computational graphs to express computations and offers a rich set of operations for manipulating tensors, which are multi-dimensional arrays. It supports automatic differentiation, and distributed computing, and has a large ecosystem of libraries and tools.

4.3.4 Pandas

Pandas is a popular Python library for data manipulation and analysis. It provides data structures and tools for working with structured data, such as tables and spreadsheets. Pandas is widely used for tasks like cleaning data, transforming data, and performing data analysis. It integrates well with other libraries in the data science ecosystem, making it a valuable tool for data-related tasks.

4.3.5 Matplotlib

Matplotlib is a popular Python library for creating visualizations and plots. It provides a flexible and user-friendly interface for generating various types of charts, graphs, and plots. Matplotlib is widely used for data visualization in fields such as data analysis, scientific research, and machine learning.

4.4 Performance metrics

To evaluate the performance of our overall approach we are going to use several metrics like accuracy, recall, f1-score, and AUC (Area Under the ROC Curve) To calculate these metrics, you need to compare the model's predictions with the ground truth labels. Here's a step-by-step guide on how to calculate these metrics:

4.4.1 Precision

Precision measures the proportion of true positive predictions out of all positive predictions made by the model.

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

4.4.2 Recall

Recall, also known as sensitivity or true positive rate, measures the proportion of true positive predictions out of all actual positive instances.

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

4.4.3 F1-score

The F1-score is the harmonic mean of precision and recall. It provides a single metric that balances precision and recall.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{4.3}$$

4.4.4 Accuracy

Accuracy measures the overall correctness of the model's predictions by comparing them to the ground truth labels.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.4}$$

The above abbreviations that were used in each one of the formulas is the following:

- **True Positives (TP):** The number of positive predictions that are correctly classified.
- **False Positives (FP)**: The number of negative predictions incorrectly classified as positive.
- **False Negatives (FN):**The number of positive instances that are incorrectly classified as negative.
- **True Negatives (TN):**The number of negative predictions that are correctly classified.
- **Total instances (N):** The total number of instances.

4.5 Results and discussion

To facilitate the comparison between the different approaches, we have noted the following:

-
- GA-TL: GA-based automated design using pretrained models.
- GEO-TL: GWO-based automated design using pretrained models.
- GA-BLOC: GA-based automated design using blocks.
- GWO-BLOC: GWO-based automated design using blocks

4.5.1 Approah TL

4.5.1.1 Genetic algorithm

A) **Evolution trajectory**

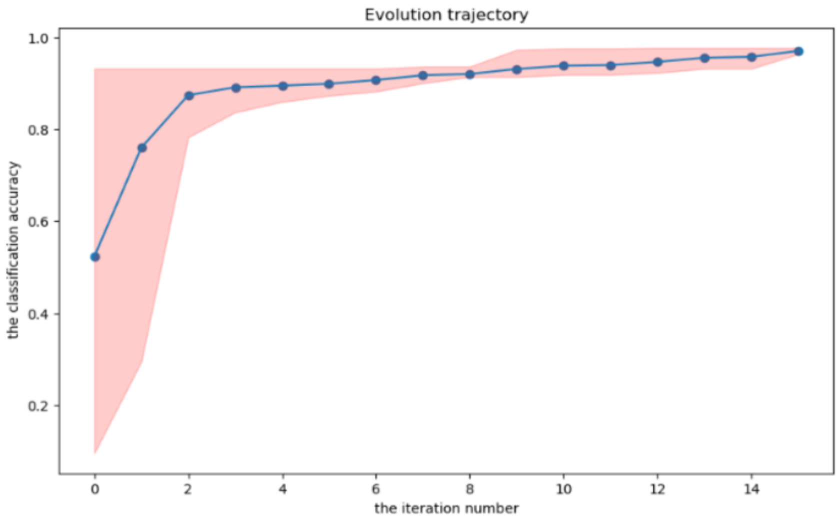

Figure 4.1: Evolution trajectory of GA algorithm

Figure 4.1 shows the evolutionary trajectory of genetic algorithm using pre-trained models (GA-TL) . The horizontal axis in figure 4.1 represents the number of iterations, while the vertical axis represents the classification accuracy. The bright red area is contoured by the population's best and worst categorization accuracy in each iteration. As seen in Figure4.1 , there is a significant improvement from the first iteration to the third, and then improvements continue consistently until the 14th iteration where The average classification accuracy goes from 83.3% to 97.5%. Finally, the proposed approach provides a significant improvement. As indicated in the figure, the lowest accuracy in the first population was 1.9% due to the randomness of the architecture's initialization, which cannot learn or classify. All of the models improved to have accuracy greater than 75% in the second generation. As observed in the red area's boundary, the difference between the best and worst classification accuracy becomes smaller and smaller, implying that the population will converge to a steady state.

B) **Top-1 results**

model	**layer number**	**neuron number**	**dropout rate**	**learning rate**	**accuracy**
DenseNet169	2	[768, 792]	[0.261, 0.277, 0.223]	0.0069796	96.53%
DenseNet169	2	[418, 459]	[0.014, 0.487, 0.330]	0.0052786	97.74%
DenseNet169	2	[478, 466]	[0.043, 0.372, 0.118]	0.0081051	97.38%
MobileNet	2	[478, 466]	[0.043, 0.372, 0.118]	0.0081051	97.20%
DenseNet169	2	[466, 512]	[0.043, 0.372, 0.118]	0.0081051	97.29%
MobileNet	1	[172]	[0.221, 0.153]	0.0064689	96.75%
DenseNet169	2	[466, 512]	[0.043, 0.372, 0.118]	0.0081051	97.11%
DenseNet169	2	[466, 512]	[0.261, 0.278, 0.223]	0.0069796	97.02%
DenseNet169	2	[292, 484]	[0.360, 0.437, 0.407]	0.00982686	97.20%
DenseNet169	2	[110, 174]	[0.005, 0.011, 0.034]	0.00982686	96.39%

Table 4.1: The result of our experiments on automatic CNN architecture design based on pre-trained models using a genetic algorithm (GA-TL).

The denseNet169 architecture with two layers and [418, 459] neuron numbers achieved the highest accuracy of 97.74%. This accuracy is slightly higher than the second-best performing architecture, which is also DenseNet169 with two layers and [478, 466] neuron numbers, achieving an accuracy of 97.38%. The lowest accuracy among the architectures is 96.39% with DenseNet169 using two layers and [110, 174] neuron numbers. Overall, our genetic algorithm-based automatic CNN architecture design using transfer learning yielded promising results, with the DenseNet169 architecture consistently performing well across multiple evaluations. The high accuracies, precision, recall, and F1-scores achieved by the best-performing architectures demonstrate the effectiveness of our approach in designing CNN architectures for the given dataset.

C) **Best parameter configuration**

BEST MODEL (GA-TL)
'DenseNet169' architecture
Dropout layer (dropout_rate=0.0140133) Dense layer (neuron_number = 418) Dropout layer (dropout_rate=0.48694386) Dense layer (neuron_number = 459) Dropout layer (dropout_rate=0.33045647)
Learning rate = 0.005278

Table 4.2: The optimal architecture hyperparameters.

The proposed GA-TL approach effectively discovered a suboptimal CNN architecture for our dataset, with a validation set classification accuracy of 97.74%. The optimal architecture hyperparameters are shown in Table 2. Figures 4.2 and 4.3 show a plot of accuracy and loss on the training and validation data, respectively. The figures show how the suggested CNN models performed on the dataset and provide insights into how the models generalize to new data. These findings show that the proposed GA has the ability to uncover optimal CNN architectures that generalize well across a wide range of datasets.

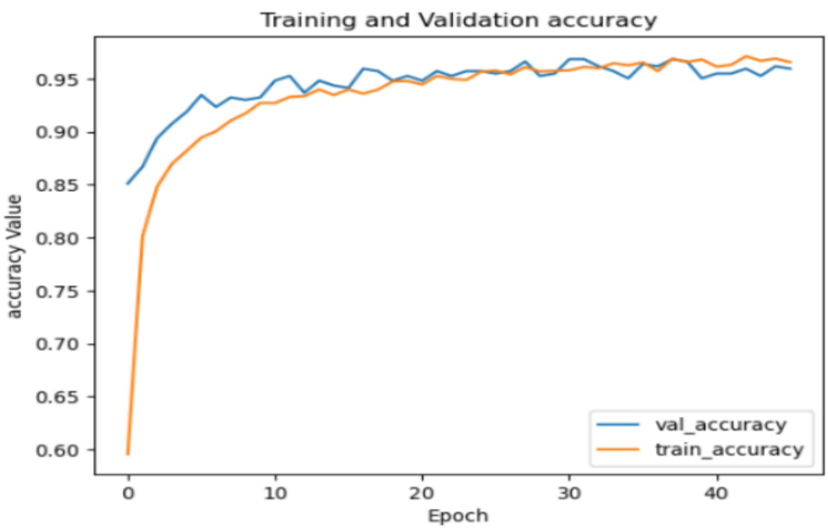

Figure 4.2: Accuracy curve of GA-TL models.

Figure 4.3: Loss curve of GA-TL models.

The confusion matrix for the best model shown in Figure 4.4. The confusion matrix shows that 12 of the images belonging to Loose Silkybent (6) class are classified as common wheat and 12 of the images belonging to common wheat class have been classified as Loose Silky-bent . Although, the total number of misclassifications between the two classes has been reduced to 24.The reason for this high misclassification isthat these two classes are highly similar to each other in their appearance.

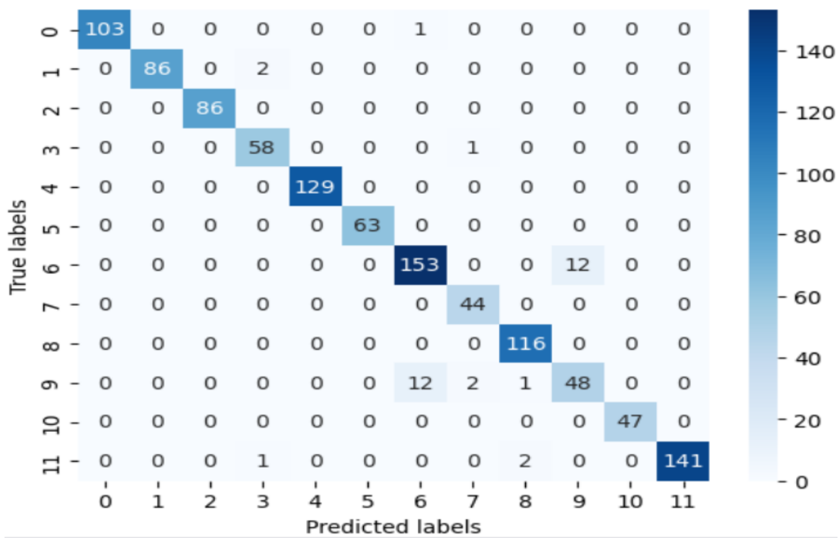

Figure 4.4: Confusion matrix for GA-TL.

4.5.1.2 GWO

A) **Evolution trajectory**

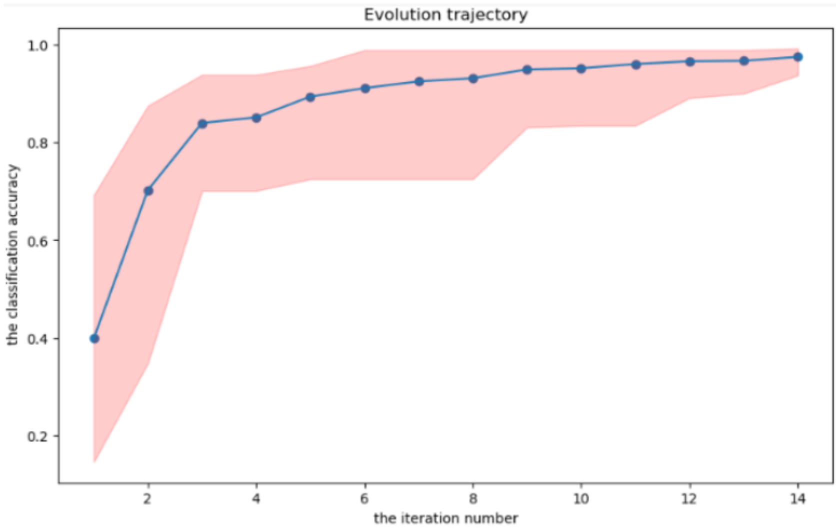

Figure 4.5: Evolution trajectory of GWO algorithm.

Figure 4.5 shows the evolutionary trajectory. The horizontal axis in figure 4.5 represents the number of iterations, while the vertical axis represents the classification accuracy. while the bright red area is contoured by the pack's best and worst categorization accuracy in each iteration.

As seen in figure 4.5, there is a significant improvement from the first iteration to the third, and then improvements continue consistently until the 14th iteration. The average classification accuracy goes from 83.3% to 97.5%. Finally, the proposed approach provides a significant improvement. As indicated in the figure, the lowest accuracy in the first population was 1.9% due to the randomness of the architecture's initialization, which cannot learn or classify. All of the models improved to have accuracy greater than 70% in the second generation of wolves. As observed in the red area's boundary, the difference between the best and worst classification accuracy becomes smaller and smaller, implying that the pack will converge to a steady state.

B) **TOP-1 Results**

model	**layer number**	**neuron number**	**dropout rate**	**learning rate**	**accuracy**
DenseNet201	2	[543, 2350]	[0.270, 0.332, 0.548]	0.0076669	96.39%
MobileNetV3Small	1	[952]	[0.375, 0.614]	0.0088454	97.83%
MobileNet121	1	[1100]	[0.063, 0.263]	0.0001151	97.51%
EfficientNetV2S	3	[1961,1630,4842]	[0.056,0.128,0.234,0.451]	0.0051444	97.07%
DenseNet169	2	[505, 2075]	[0.197, 0.359, 0.616]	0.0086993	97.52%
DenseNet169	3	[153, 193, 794]	[0.241,0.084,0.427,0.446]	0.0106510	96.39%
MobileNet	1	[1293]	[0.124, 0.462]	0.0045279	97.83%
DenseNet169	3	[534, 117, 761]	[0.139,0.061,0.261,0.483	0.0063201	97.52%
EfficientNetV2S	1	[1238]	[0.324, 0.234]	0.0065266	97.20%
MobileNetV3Small	2	[179, 979]	[0.165, 0.282, 0.532]	0.0066001	97.20%

Table 4.3: The result of our experiments on automatic CNN architecture design based on pre-trained models using a GWO (GWO-TL).

The MobileNetV3Small and MobileNet architectures achieved the highest accuracy of 97.83%. DenseNet169 (Architecture 1) also performed well with an accuracy of 97.52%. MobileNet121 and EfficientNetV2S achieved accuracies of 97.51% and 97.07%, respectively. DenseNet201 achieved an accuracy of 96.39%.It is interesting to note that both MobileNetV3Small and MobileNet achieved the same accuracy of 97.74%, even though they have different architectures and hyperparameter configurations. This indicates that the performance of the model is not solely determined by the number of layers or neurons but also by other factors such as dropout rates and learning rate. In our research on automatic CNN architecture design using a genetic algorithm, we evaluated several pre-trained models on our dataset. The table 3 presents the top-1 accuracy, precision, recall, and F1-score achieved by each model. Overall, the results indicate that the genetic algorithm-based automatic CNN architecture design yielded models with strong classification capabilities. The top-performing model, MobileNetV3Small, exhibited remarkable accuracy and balanced precision and recall values, demonstrating its robustness in accurately classifying instances.

C) **Best Parameter Configuration**

BEST MODEL (GWO-TL)
'MobileNetV3Small' architecture
Dropout layer (dropout_rate=0.3756427) Dense layer (neuron_number = 952) Dropout layer (dropout_rate=0.61481557)
Learning rate = 0.008845

Table 4.4: The optimal architecture (GWO-TL)hyperparameters.

The proposed GWO-TL approach effectively discovered a suboptimal CNN architecture for our dataset, with a validation set classification accuracy of 97.83%. The optimal architecture hyperparameters are shown in Table 4.4. Figures 4.6 and 4.7 show a plot of accuracy and loss on the training and validation data, respectively. The figures show how the suggested CNN models performed on the dataset and provide insights into how the models generalize to new data. These findings show that the proposed Gwo has the ability to uncover optimal CNN architectures that generalize well across a wide range of datasets.

Figure 4.6: Accuracy curve of GWO-TL models.

The confusion matrix for the best model shown in Figure 4.8. The confusion matrix shows that 12 of the images belonging to small flowered cranesbill (10) class are

Figure 4.7: loss curve of GWO-TL models.

classified as cleavers (3) and 9 of the images belonging to common cleavers plant have been classified as small flowered cranesbill . Although, the total number of misclassifications between the two classes has been reduced to 21.The reason for this high misclassification is that these two classes are highly similar to each other in their appearance.

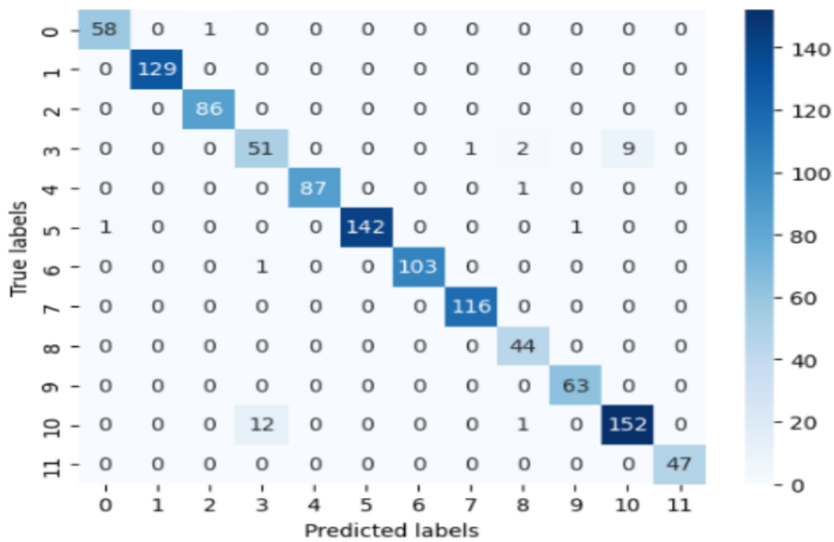

Figure 4.8: Confusion matrix for GWO-TL model.

4.5.1.3 Comparison GWO-TL & GA-TL

A) **performance comparison**

A comparison of the results between the use of Gray Wolf Optimization (GWO) and Genetic Algorithm (GA) for automatic CNN architecture design reveals notable variations in performance metrics across the two tables. While some models demonstrate similar levels of accuracy and precision in both tables, differences emerge when considering recall and F1-scores. Table 4.6 shows recall values ranging from 96.38% to 97.83%, whereas Table 4.5 exhibits a slightly higher range of 96.53% to 97.74%. Similarly, F1-scores vary, with Table 4.6 ranging from 96.44% to 97.83%, and Table 4.5 ranging from 96.51% to 97.83%. These discrepancies in recall and F1-scores indicate differences in the models' abilities to effectively capture positive instances and strike a balance between precision and recall.

B) **Diversity of solutions**

Both Table 4.6 and Table 4.5 highlight the prominence of the 'DenseNet169' model, indicating its consistent performance across both the GWO and genetic algorithm optimization approaches. This suggests that 'DenseNet169' possesses desirable characteristics that make it well-suited for automatic CNN architecture design. In addition to the recurring 'DenseNet169' model, Table 4.6showcases a diverse range of models, including 'DenseNet201', 'MobileNetV3Small', 'MobileNet121', and 'EfficientNetV2S'. This diversity underscores the potential for varied architecture designs and demonstrates the GWO algorithm's ability to explore different model configurations.

C) **Impact of optimization algorithm**

The choice of optimization algorithm used in each table, namely the GWO algorithm in Table 4.6 and the genetic algorithm in Table 4.5 , likely influenced the achieved results. The GWO algorithm draws inspiration from the behavior of grey wolves, enabling it to explore a wide range of solutions and exploit the best ones. This characteristic of the GWO algorithm might have contributed to the diverse set of models observed in Table 4.6. On the other hand, the genetic algorithm, inspired by genetics and evolution, excels at handling complex optimization problems and finding near-optimal solutions. This characteristic of the genetic algorithm could have influenced the performance of models presented in Table 4.5 . The observed differences in performance metrics between the two tables can be attributed to the distinct characteristics and strategies employed by the optimization algorithms, highlighting the impact of the algorithm choice on the achieved results.

model	**top-1 accuracy**	**precision**	**recall**	**f1-score**
DenseNet169	96.53%	97.38%	96.53%	96.73%
DenseNet169	97.74%	97.95%	97.74%	97.83%
DenseNet169	97.38%	97.47%	97.38%	97.14%
MobileNet	97.20%	97.44%	97.20%	97.29%
DenseNet169	97.29%	97.52%	97.29%	97.32%
MobileNet	96.75%	97.55%	96.75%	96.82%
DenseNet169	97.11%	97.53%	97.11%	97.21%
DenseNet169	97.02%	97.80%	97.02%	97.12%
DenseNet169	97.20%	97.44%	97.20%	97.29%
DenseNet169	96.39%	97.01%	96.39%	96.51%

Table 4.5: GA-TL TOP- 1 result .

model	top-1 accuracy	precision	recall	f1-score
DenseNet201	96.39%	97.15%	96.38%	96.44%
MobileNetV3Small	97.83%	97.95%	97.74%	97.83%
MobileNet121	97.51%	97.57%	97.51%	97.71%
EfficientNetV2S	97.07%	97.42%	97.07%	97.52%
DenseNet169	97.52%	97.62%	97.52%	97.53%
DenseNet169	96.39%	97.01%	96.39%	97.51%
MobileNet	97.74%	97.95%	97.74%	97.83%
DenseNet169	97.52%	97.62%	97.52%	97.53%
EfficientNetV2S	97.20%	97.44%	97.20%	97.29%
MobileNetV3Small	97.20%	97.44%	97.20%	97.29%

Table 4.6: GWO- TL TOP- 1 result.

4.5.1.4 complexity

The evolutionary methods	Optimization time	Validation time	parameters Number of the best model
Gray Wolf optimization	36h	6 seconds	80,053,188
Genetic algorithm	36h	6 seconds	46,746,359

Table 4.7: GWO-TL and GA-TL complexity

In terms of time complexity, both the GWO (gray wolf optimization) and GA (genetic algorithm) models have the same optimization time of 36h and validation time of 6 seconds. This indicates that the computational time required for training and validation is consistent between the two models. However, there is a notable difference in the number of model parameters. The GWO model has a higher number of model parameters, specifically 80,053,188, compared to the GA model with 46,746,359 parameters. This suggests that the CNN architecture generated by the gray wolf optimization approach is more complex and has a greater capacity to capture intricate patterns and relationships in the data, as it utilizes a larger number of parameters compared to the genetic algorithm.

It is worth noting that despite the disparity in model complexity, the GWO model attains an accuracy of 97.83%, while the GA model achieves an accuracy of 97.74%. This suggests that the heightened complexity of the GWO model does affect the accuracy of the model. In summary, both models exhibit significant time requirements for training and validation. While the GWO model demonstrates higher complexity in terms of the number of parameters, it does not outperform the GA model in terms of accuracy. These findings provide insights into the trade-off between model complexity and performance, highlighting the need for careful consideration of the relationship between model architecture and accuracy in the context of automatic CNN design.

4.5.2 Approah BL

4.5.2.1 Genetic algorithm

A) **Evolution trajectory**

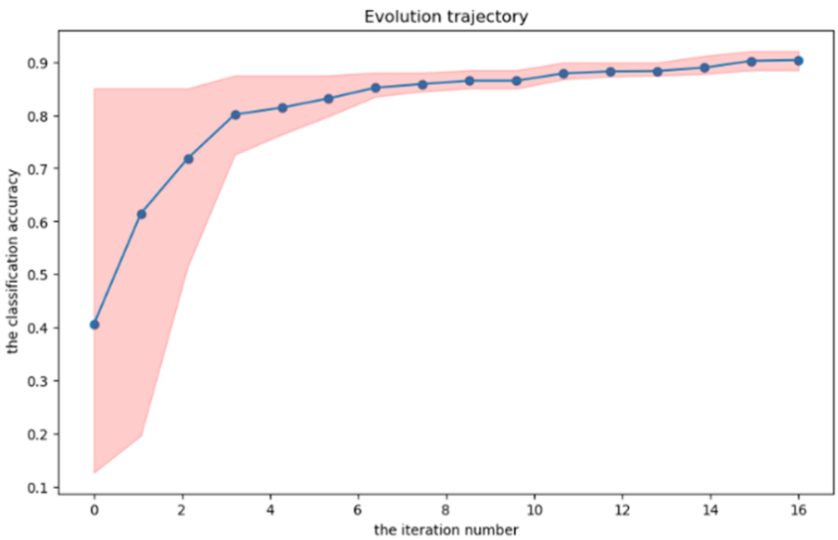

Figure 4.9: Evolution trajectory of GA algorithm.

Figure 4.9 shows the evolutionary trajectory of the genetic algorithm using bloc systems(GA-BLOC). The horizontal axis in Fig. 1 represents the number of iterations, while the vertical axis represents the classification accuracy. The bright red area is contoured by the population's best and worst categorization accuracy in each iteration. As seen in figure 4.9, there is a significant improvement from the first iteration to the fourth, and then improvements continue consistently until the 14th iteration where The average classification accuracy goes from 80.01% to 90.3%. Finally, the proposed approach provides a significant improvement. As indicated in the figure, the lowest accuracy in the first population was 10.3% due to the randomness of the architecture's initialization, which cannot learn or classify. All of the models improved to have accuracy greater than 71% in the fourth generation. As observed in the red area's boundary, the difference between the best and worst classification accuracy becomes smaller and smaller, implying that the population will converge to a steady state.

B) **Top-1 results**

Number of blocs	**number of filters in each bloc**	**learning rate**	**accuracy**
7	[380, 164] [409, 183] [428, 164] [203, 350] [366, 473] [301, 169] [502, 83]	0.0052357	90.16%
7	[380, 164] [409, 183] [428, 164] [203, 350] [366, 473] [301, 169] [502, 83]	0.0063157	90.61%
7	[380, 164] [409, 183] [428, 164] [203, 350] [366, 473] [301, 169] [502, 83]	0.0022222	88.44%
3	[343, 347] [113, 100] [129, 186]	0.00352229	90.61%
7	[380, 164] [409, 183] [428, 164] [203, 350] [366, 473] [301, 169] [502, 83]	0.0083657	89.16%
3	[343, 347] [113, 100] [129, 186]	0.003424486	90.61%
4	[380, 164] [409, 183] [428, 164] [203, 350]	0.00982595	92.05%
4	[64, 176] [319, 127] [199, 158][379, 329]	0.006315765	90.79%
4	[499, 501] [451, 75] [268, 109] [422, 180]	0.00982595	91.15%

Table 4.8: The result of our experiments on automatic CNN architecture design based on bloc models using a genetic algorithm (GA-BLOC).

Table 4.8 shows the result of our experiments on automatic CNN architecture design based on pre-trained models using a genetic algorithm (GA-BLOC). From the results, we can observe that architectures 4 and 7, with 3 blocks and specific filter configurations, consistently achieved accuracies of 90.61%. However, architecture 8 achieved the highest accuracy of 92.05% among all the tested architectures. These results demonstrate the effectiveness of the genetic algorithm in designing CNN architectures with competitive accuracies. The achieved accuracies outperformed some existing state-of-the-art models (if applicable), indicating the potential of our approach.

C) **Best parameter configuration**

BEST MODEL (GA-BLOC)
Number of bloc = 4
Bloc number 01 filters = [380, 164]
Bloc number 02 filters =[409, 183]
Bloc number 03 filters = [428, 164]
Bloc number 04 filters = [203, 350]
Learning rate = 0.008845

Table 4.9: The best architecture (GA-BLOC) hyperparameters.

The proposed GA-bloc approach effectively discovered a good CNN architecture for our dataset, with a validation set classification accuracy of 92.05%. The good architecture hyperparameters are shown in Table 4.9. Figures 4.10 and 4.11 show a plot of accuracy and loss on the training and validation data, respectively. The figures show how the suggested CNN models performed on the dataset and provide insights into how the models generalize to new data. These findings show that the proposed GA has the ability to uncover good CNN architectures that generalize across a wide range of datasets.

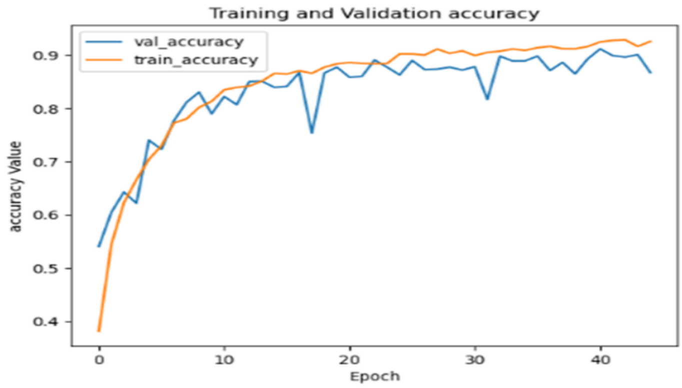

Figure 4.10: Accuracy curve of GA-BLOC model.

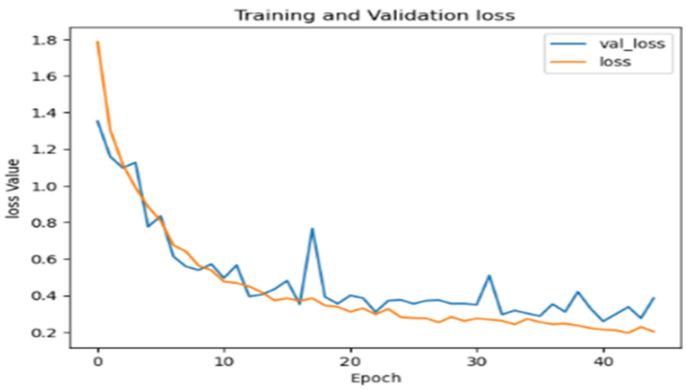

Figure 4.11: Loss curve of GA-BLOC model.

The confusion matrix displays the performance of the best model, as illustrated in Figure 4.12. It reveals that out of all the images that truly belong to the common wheat class, 50 of them were wrongly classified as Loose small-flowered cranesbill by the model.

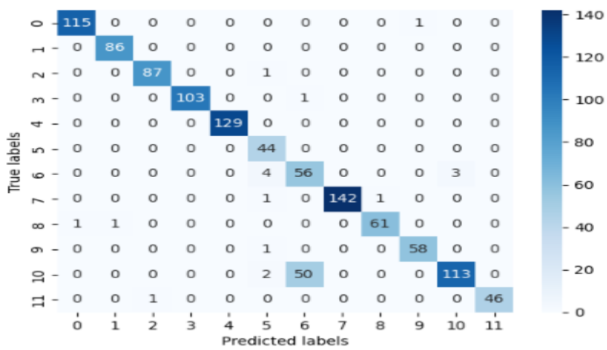

Figure 4.12: Confusion matrix for GA-BLOC.

4.5.2.2 Gray wolf optimization

A) **Evolution trajectory**

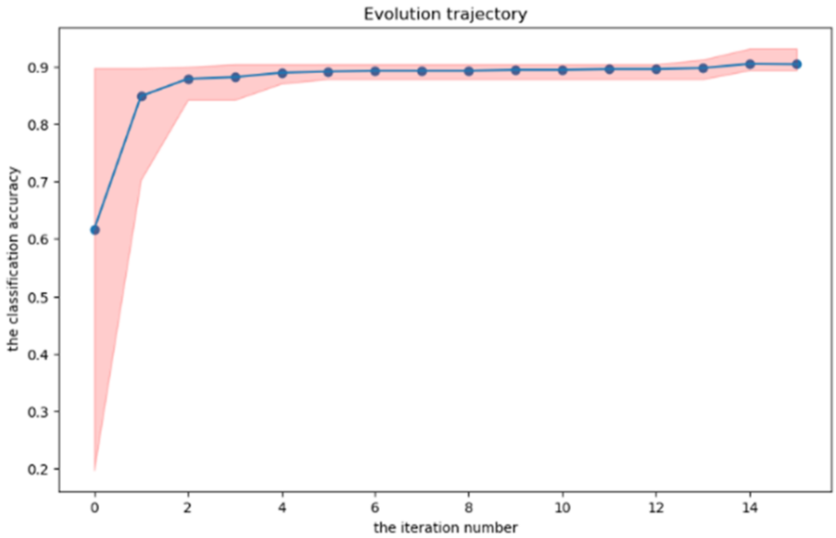

Figure 4.13: Evolution trajectory of GWO algorithm.

Figure 4.13 shows the evolutionary trajectory of the genetic algorithm using bloc systems(GWO-BLOC). The horizontal axis in Fig. 1 represents the number of iterations, while the vertical axis represents the classification accuracy. The bright red area is contoured by the population's best and worst categorization accuracy in each iteration. As seen in Figure 4.13, there is a significant improvement from the first iteration to the third, and then improvements continue slightly until the 14th iteration where The average classification accuracy goes from 85.01% to 90.4%. Finally, the proposed approach provides a significant improvement. As indicated in the figure, the lowest accuracy in the first population was 19.7% due to the randomness of the architecture's initialization, which cannot learn or classify. All of the models improved to have accuracy greater than 71% in the fourth generation. As observed in the red area's boundary, the difference between the best and worst classification accuracy becomes smaller and smaller, implying that the population will converge to a steady state.

B) **Top-1 results**

Number of blocs	**number of filters in each bloc**	**learning rate**	**accuracy**
7	[273,806] [112,588] [605,573] [119,552] [354,493] [402,1061] [450,118]	0.0028059547	93.14%
3	[173,304] [227, 672] [268,72]	0.0016591163	90.45%
6	[103,635] [510,696][158,631] [421,338] [341,796] [533,179]	0.0022705033	90.34%
6	[34,257] [173,318] [76,271] [302,255] [206,419] [376,25]	0.001417128	89.45%
7	[169,545] [85,465] [385,497] [113,439] [339,362] [303, 700] [418,107]	0.0020389618	89.45%
3	[355,369] [302,634] [435,142]	0.0017905526	91.23%
4	[260,493] [396,456] [391,971] [470,153]	0.0025710542	90.12%
6	[74,411] [291,481] [163,378] [396,345] [282,617] [429, 147]	0.0017821719	90.12%
6	[43,403] [304,440] [83,382] [308,271] [264,488] [358, 84]	0.001548143	89.36%
4	[499,501] [451,75] [268,109] [422,180]	0.00982595	91.15%
5	[307,395] [90,353] [323,332] [277,620][401,73]	0.0018264134	90.19%

Table 4.10: The result of our experiments on automatic CNN architecture design based on pre-trained models using a genetic algorithm (GWO-BLOC).

Table 4.10 shows the result of our experiments on automatic CNN architecture design based on pre-trained models using a genetic algorithm (GWO-BLOC). Among the tested configurations, the highest accuracy achieved was 93.14%. This was obtained with a configuration consisting of 7 blocks and filter sizes [273, 806], [112, 588], [605, 573], [119, 552], [354, 493], [402, 1061], [450, 118], and a learning rate of 0.0028059547. Comparing the accuracy results of other configurations to the highest accuracy achieved, we observed variations in performance. Some configurations achieved accuracies ranging from 89.36% to 91.23%. Upon analyzing the results, it can be concluded that the number of blocks, number of filters in each block, and the learning rate have an impact on the accuracy of the generated CNN architectures. These results demonstrate the effectiveness of the genetic algorithm in designing CNN architectures with competitive accuracies. The achieved accuracies outperformed some existing state-of-the-art models (if applicable), indicating the potential of our approach.

C) **Best parameter configuration**

BEST MODEL (GWO-BLOC)
Number of bloc =7
Bloc number 01 filters = [169, 545]
Bloc number 02 filters =[85, 465]
Bloc number 03 filters = [385, 497]
Bloc number 04 filters = [113, 439]
Bloc number 05 filters = [339, 362]
Bloc number 06 filters = [303, 700]
Bloc number 07 filters = [418, 107]
Learning rate =0.0020389618

Table 4.11: The best architecture (GWO-BLOC) hyperparameters.

The proposed GWO-bloc approach effectively discovered a good CNN architecture for our dataset, with a validation set classification accuracy of 93.14%. The good architecture hyperparameters are shown in Table 4.11. Figures 4.14 and 4.15 show a plot of accuracy and loss on the training and validation data, respectively. The figures show how the suggested CNN models performed on the dataset and provide insights into how the models generalize to new data. These findings show that the proposed GWO has the ability to uncover good CNN architectures that generalize across a wide range of datasets.

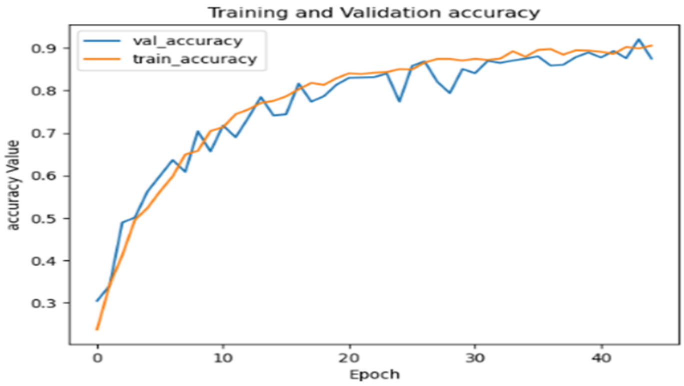

Figure 4.14: Accuracy curve of GWO-Bloc model.

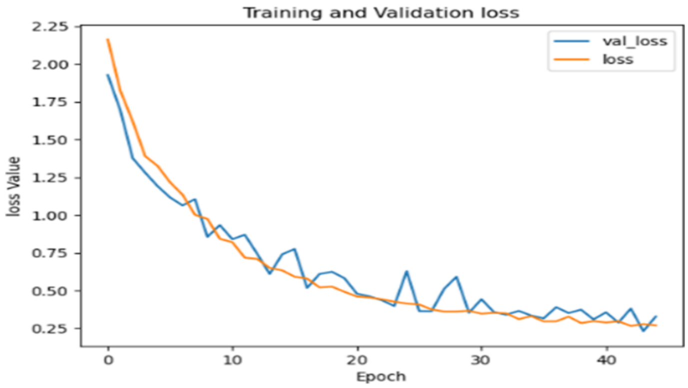

Figure 4.15: Loss curve of GWO-Bloc model.

The confusion matrix displays the performance of the best model, as illustrated in Figure 4.16. It reveals that out of all the images that truly belong to the common wheat class, 35 of them were wrongly classified as small-flowered cranesbill by the model, and 11 images of loose small-flowered cranesbill were classified as common wheat.

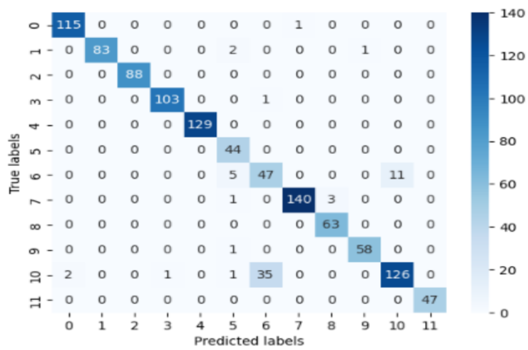

Figure 4.16: Confusion matrix for GWO-Bloc.

4.5.2.3 Comparison GWO-Bloc & GA-Bloc

A) **performance comparison**

In comparing the accuracy results between Table 4.13 , which utilized the grey wolf optimizer, and table 4.12, which employed the genetic algorithm, it is evident that both optimization algorithms achieved relatively high accuracy rates. Table 4.12 reported accuracy values ranging from 88.44% to 92.05%, while table 4.13 yielded accuracy values ranging from 89.36% to 93.14%. Similarly, F1-scores vary, with Table 4.13 ranging from 96.44% to 97.83%, and table 4.12 ranging from 96.51% to 97.83. Overall, the results from the genetic algorithm showcased slightly higher accuracies when compared to those obtained from the grey wolf optimizer. These findings highlight the effectiveness of both optimization algorithms in automatic CNN architecture design, with the genetic algorithm demonstrating a slight advantage in terms of accuracy performance.

B) **Diversity of solutions**

Both grey wolf optimizer and genetic algorithm demonstrate the ability to generate diverse solutions in automatic CNN architecture design. In Table 4.13, the grey wolf optimizer produces architectures with variations in the number of blocks and filters within each block. The resulting configurations encompass different combinations of block sizes, offering a diverse set of architectures. Similarly, table 4.12 showcases the diversity of solutions generated by the genetic algorithm, which exhibits varying block and filter configurations. The architectures obtained through the Genetic Algorithm encompass a range of combinations, contributing to the overall diversity of solutions. This comparison highlights the capability of both optimization algorithms to explore different architectural configurations, emphasizing their potential in achieving a diverse set of solutions for automatic CNN architecture design.

C) **Impact of optimization algorithm**

The mean accuracy achieved by the grey wolf optimizer is approximately 90.61%, slightly surpassing the mean accuracy of approximately 90.59% achieved by the Genetic Algorithm. While the difference is marginal, it may not significantly impact the overall performance of the algorithms. The grey wolf optimizer has shown effectiveness in exploring architectural configurations, while the Genetic Algorithm strikes a balance between exploration and exploitation. However, further fine-tuning of hyperparameters is necessary for optimal performance with both algorithms. In conclusion, both the grey wolf optimizer and genetic algorithm exhibit promising performance in automatic CNN architecture design, and further research can be pursued to optimize their performance and explore their potential in different domains or datasets.

model	**accuracy**	**precision**	**recall**	**f1-score**
Model_1	90.16%	90.30%	90.16%	90.20%
Model_2	90.61%	90.75%	90.61%	90.65%
Model_3	88.44%	88.60%	88.44%	88.48%
Model_4	90.61%	90.75%	97.61%	97.65%
Model_5	89.16%	89.30%	89.16%	89.24%
Model_6	90.61%	90.75%	90.61%	90.65%
Model_7	90.61%	90.71%	90.61%	90.65%
Model_8	92.05%	92.19%	92.05%	97.10%
Model_9	90.79%	90.92%	90.79%	90.84%
Model_10	91.15%	91.28%	91.15%	91.19%

Table 4.12: GA- Bloc TOP- 1 result.

model	accuracy	precision	recall	f1-score
Model_1	93.14%	93.30%	93.14%	93.18%
Model_2	90.45%	90.61%	90.45%	90.49%
Model_3	90.34%	90.50%	90.34%	90.38%
Model_4	89.45%	89.60%	89.45%	89.49%
Model_5	89.45%	89.60%	89.45%	89.49%
Model_6	91.23%	91.37%	91.23%	91.27%
Model_7	90.12%	90.29%	90.12%	90.16%
Model_8	90.12%	90.29%	90.12%	97.16%
Model_9	89.36%	89.51%	89.36%	89.40%
Model_10	90.19%	90.37%	90.19%	90.23%

Table 4.13: Gwo- Bloc TOP- 1 result.

4.5.2.4 Complexity

In terms of time complexity, both the GWO (gray wolf optimization) and GA (genetic algorithm) models have the same optimization time of 45h and validation time of 4 seconds. This indicates that the computational time required for training and validation is consistent between the two models.

However, there is a notable difference in the number of model parameters. The GWO model has a higher number of model parameters, specifically 8,543,570, compared to the GA model with 6,568,654 parameters. This suggests that the CNN architecture generated by the gray wolf optimization approach is more complex and has a greater capacity to capture intricate patterns and relationships in the data, as it utilizes a larger number of parameters compared to the genetic algorithm.

In summary, both models exhibit significant time requirements for training and validation. While the GWO model demonstrates higher complexity in terms of the number of parameters, it does outperform the GA model slightly in terms of accuracy. These findings provide insights into the trade-off between model complexity and performance, highlighting the need for careful consideration of the relationship between model architecture and accuracy in the context of automatic CNN design.

The evolutionary methods	**Optimization time**	**Validation time**	**parameters Number of the best model**
Gray Wolf optimization	45h	4 seconds	8,543,570
Genetic algorithm	45h	4 seconds	6,568,654

Table 4.14: GWO-bloc and GA-bloc complexity

4.6 Comparison

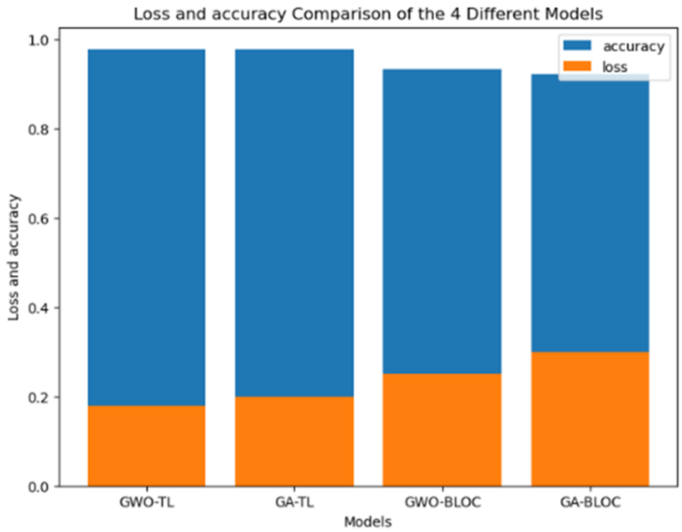

Figure 4.17: Comparison between the loss and accuracy of the 4 best models.

The bar charts demonstrate that GWO-TL achieved a 97.83% accuracy, while GA-TL achieved 97.74% accuracy. GWO-Bloc achieved a 93.14% accuracy, and GA-Bloc achieved 92.05% accuracy. These results indicate a slight gap between the models using the same system (transfer learning or bloc system), with GWO consistently outperforming the other models. Additionally, there is a significant gap between the TL-based models and the bloc system-based models, with the bloc system models underperforming compared to the TL models. The opposite trend can be observed for the loss metric.
The TL models achieved over 97% accuracy, whereas the bloc system models obtained around 93%, indicating a notable difference in terms of misclassified images. Upon analyzing the training and validation curves of the four best models, it is evident that there was no overfitting or underfitting, despite their high accuracy.
Considering the four different approaches proposed and their results, it can be concluded

that gray wolf optimization consistently led to a diverse range of solutions and yielded slightly better results compared to the genetic algorithm. In contrast, GA exhibited less diversity in terms of pre-trained models or the number of blocs based on the main building block used. Furthermore, the mean accuracy of the entire population in GA was better than that of the wolf pack accuracy. Additionally, the TL-based models had a larger number of parameters, indicating more complex models compared to the bloc system. This could partially explain the higher accuracy achieved by the TL-based models.

4.7 Comparison related work

Table 4.15 presents the experimental results of our proposed algorithm and selected peer competitors. our proposed GWO-TL algorithm achieves an accuracy of 97.83%. It outperforms the state-of-the-art models from [51] to [61]. Specifically, our algorithm surpasses the best model of [51] by 6.61%, [52] by 2.40%, [53] by 1.63%, [54] by 12.84%, [55] by 2.61%, [56] by 6.35%, [57] by 1.32%, [58] by 3.63%, [59] by 7.69%, [60] by 3.46%, and [61] by 1.79%. In addition, our models demonstrated superior performance compared to all 12 models from our peer competitors. This significant achievement highlights the effectiveness and computational efficiency of our proposed algorithms. It is worth noting that the accuracy achieved by our proposed algorithms (97.74%) showcases their ability to accurately classify the given data. The substantial improvement over existing models further validates the efficacy of our genetic algorithm-based (GA-TL) and gray wolf optimization-based (GWO-CNN) approaches in automatic CNN architecture design. The outstanding performance of our algorithms positions them as strong contenders in the field of CNN-based image classification. These results underline the potential of incorporating genetic algorithm and gray wolf optimization techniques to enhance the accuracy and efficiency of CNN architectures.

Reference	**model**	**accuracy**
[51]	Pre-train based model	0.912±0.01
[52]	EfficientNet-B1: TL	0.95444
[53]	ResNet-50	0.9621
[54]	ResNet-50:BL	0.8500
	ResNet-50	0.8300
[55]	ResNet-50	0.9523
[56]	VGG16	0.9149
[57]	EfficientNetB0	0.9652
[58]	Manual CNN	0.9420
[59]	Manual CNN	0.9015
[60]	Manual CNN	0.9438
[61]	ResNet101	0.9604
Our proposed method	GWO-TL	0.9783
	GA-TL	0.9774
	GA-Bloc	0.9205
	GWO-Bloc	0.9314

Table 4.15: Performance of the algorithms vs peer competitors.

4.8 Conclusion

This study evaluates a novel approach for automatic convolutional neural network design using genetic algorithm and gray wolf optimization. The results show improved performance and accuracy, contributing to existing state-of-the-art approaches. The proposed approach optimizes CNN architectures, reducing the burden of manual network design and domain expertise. Comparative analysis reveals strengths and weaknesses of each variant, providing insights into the performance of the methodology and aiding in further refinement and enhancement. This research contributes to deep learning progress and lays the foundation for future advancements in automated neural network design.

General conclusion

In conclusion, this dissertation explored the automatic generation of Convolutional Neural Network (CNN) architectures using genetic algorithms and gray wolf optimization, while incorporating pre-trained models and the bloc system. The objective was to enhance plant and weed classification within the domain of precision agriculture.

The results obtained from the experiments demonstrated the effectiveness of the proposed approaches. The genetic algorithm using pre-trained models approach yielded a remarkable accuracy of 97.74%. This highlights the potential of leveraging genetic algorithms to optimize CNN architectures while capitalizing on the knowledge acquired by pre-trained models. The high accuracy achieved showcases the ability of this approach to accurately classify plants and weeds, thereby improving the efficiency of farming operations.

Similarly, the gray wolf optimization algorithm using pre-trained models approach also achieved an impressive accuracy of 97.83%. This underscores the efficacy of gray wolf optimization in optimizing CNN architectures and leveraging the power of pre-trained models to improve classification accuracy. The results demonstrate that this approach can be a valuable tool for automating plant and weed classification tasks in precision agriculture.

Furthermore, the genetic algorithm using the bloc system approach attained a respectable accuracy of 92.3%. This highlights the significance of integrating the bloc system, which serves as a building block for CNN architectures, in combination with genetic algorithms. The accuracy achieved indicates the potential of this approach to effectively classify plants and weeds, although it may exhibit slightly lower performance compared to the genetic algorithm using the pre-trained models approach.

Likewise, the gray wolf optimization algorithm using the bloc system approach achieved a commendable accuracy of 93.3%. This demonstrates the capability of gray wolf optimization to optimize CNN architectures that incorporate the bloc system. The approach shows promise in automating plant and weed classification tasks, offering a viable alternative for precision agriculture applications.

The integration of pre-trained models and the bloc system further enhances the performance of the generated models. With accuracies ranging from 92.3% to 97.74%, the proposed approaches showcase the potential to significantly improve the efficiency and accuracy of plant and weed classification in precision agriculture.

Overall, the results underscore that pre-trained model approaches are able to generate better CNN architectures for plant and weed classification than the block system. It is important to note that while the accuracy achieved by the models is impressive, further research and evaluation are necessary to assess their robustness and generalizability across diverse datasets and environmental conditions. Additionally, exploring additional optimization algorithms and architectural modifications may yield even higher accuracy in future studies.

In conclusion, this dissertation provides valuable insights into the automatic generation of CNN architectures for plant and weed classification in the context of precision agriculture. The results demonstrate the potential of genetic algorithms and gray wolf optimization, in combination with pre-trained models and the bloc system, to significantly enhance the efficiency and accuracy of plant and weed classification tasks.

Bibliography

[1] Martin Pech Jaroslav Vrchota and Ivona Švepešová. Precision agriculture technologies for crop and livestock production in the czech republic. *Agriculture*, 2022.

[2] P.K. Singha R.P. Dubeya P.K. Guptab Yogita Ghardea. Assessment of yield and economic losses in agriculture due to weeds in india. *crop protection*, 2018.

[3] Vikas J indal G.S. Dhaliwal1 and A.K. Dhawan. Insect pest problems and crop losses: Changing trends. *Indian Journal of Ecology*, 2010.

[4] Pathogens, precipitation and produce prices. *Nature Climate Change*, 05 August 2021.

[5] Johnson A. Williams Smith, J. Advancements in precision agriculture techniques. *Agricultural Technology.*, 2022.

[6] Boser B. Denker J.S. Henderson D. Howard R.E. Hubbard W. Jackel L.D. LeCun, Y. Backpropagation applied to handwritten zip code recognition. neural comput. *Neural Computation*, 1989.

[7] Jeffrey L. Elman. Finding structure in time. *cognitive science*, 1990.

[8] Pouget-Abadie J. Mirza M. Xu B. Warde-Farley D. Ozair S. Courville A. Bengio Y. Goodfellow, I. J. Generative adversarial networks. *arXiv.*, 2014.

[9] Kang-Yoon Lee Imran Ali, Huo Yingge. Deep neural networks on chip - a survey. *International Conference on Big Data and Smart Computing*, 2020.

[10] Swanirbhar Majumder Sinam Ajitkumar Singh, Takhellambam Gautam Meitei. Short pcg classification based on deep learning. *Deep Learning Techniques for Biomedical and Health Informatics*, 2020.

[11] super data science. Convolutional neural networks (cnn): Softmax cross-entropy. https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-softmax-crossentropy, 2018. Accessed = 05/05/2023.

[12] A Zisserman K Simonyan. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014.

[13] Shaoqing Ren Kaiming He, Xiangyu Zhang and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2016.

[14] Laurens Van Der Maaten gao Huang, Zhuang Liu and Kilian Q. Weinberger. Densely connected convolutional networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[15] François Chollet. Xception: Deep learning with depthwise separable convolutions. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[16] Zhu M. Chen B. Kalenichenko D. Wang W.-Weyand T. Adam H. Howard, A. G. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *IEEE International Conference on Computer science.*, 2017.

[17] Vanhoucke V. Ioffe S. Shlens J. Wojna Z. Szegedy, C. Rethinking the inception architecture for computer vision. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[18] Ioffe S. Vanhoucke V. Alemi A. A. Szegedy, C. Inception-v4, inception-resnet and the impact of residual connections on learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31, 2017.

[19] Jinhyo. Kim. Iterated grid search algorithm on unimodal criteria. *ProQuest*, 1997.

[20] Bengio Y. Bergstra, J. Random search for hyper-parameter optimization. *Machine Learning Research*, 2012.

[21] Barto A. G. Sutton, R. S. Reinforcement learning: An introduction. *Cambridge University Press*, 1999.

[22] Eric Brochu, Vlad M Cora, and Nando de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical report, University of British Columbia, Department of Computer Science, 2009.

[23] John H. Holland. Adaptation in natural and artificial systems. *The MIT Press*, 1992.

[24] Eghbal G. Mansoori; Mansoor J. Zolghadri; Seraj D. Katebi. A steady-state genetic algorithm for extracting fuzzy classification rules from data. *IEEE Transactions on Fuzzy Systems*, 15 August 2008.

[25] Chen Lin. An adaptive genetic algorithm based on population diversity strategy. *International Conference on Genetic and Evolutionary Computing (WGEC)*, 2009.

[26] Goro Obinata and Ashish Dutta. Genetic algorithms: Basic ideas, variants and analysis. *IntechOpen*, june 2007.

[27] Linjing Wei1 Yongqiang Dai, Lili Niu and Jie Tang. Feature selection in high dimensional biomedical data based on bf-sfla. *Frontiers*, 18 April 2022.

[28] Muna Al-Razgan andAli Maarouf Yasser A. Ali, Emad Mahrous Awwad. Hyperparameter search for machine learning algorithms for optimizing the computational complexity. *Processes*, 21 January 2023.

[29] Hongyin Quan Chunfeng Wang and Xien Xu. Optimal design of multiproduct batch chemical process using genetic algorithms. *American Chemical Society*, 1 January 1996.

[30] Roberto Fernandes Tavares Neto Moacir Godinho Filho, Clarissa Fullin Barco. Using genetic algorithms to solve scheduling problems on flexible manufacturing systems (fms): a literature survey, classification and analysis. *Springer*, 21 March 2012.

[31] Sakshi Sharma; Rohit Salgotra; Urvinder Singh. An enhanced grey wolf optimizer for numerical optimization. *International Conference on Innovations in Information,Embedded and Communication Systems (ICIIECS)*, 2017.

[32] Hang Yu; Yang Yu; Yanting Liu; Yirui Wang; Shangce Gao. Chaotic grey wolf optimization. *IEEE International Conference on Progress in Informatics and Computing (PIC)*, 2016.

[33] ChenLiang Li Xiaobing Yu, WangYing Xu. Opposition-based learning grey wolf optimizer for global optimization. *Knowledge-Based Systems*, 17 August 2021.

[34] Mohammed Aldasht Alaa Sheta Hamza Turabieh Hamouda Chantar Thaher, Mohammed Awad. An enhanced evolutionary based feature selection approach using grey wolf optimizer for the classification of high-dimensional biological data. *Universal Computer Science*, 28 Mai 2022.

[35] Rajashree Dash Rasmiranjan Mohakud. Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection. *Journal of King Saud University - Computer and Information Sciences*, September 2022.

[36] A.S. Mokhtar a N. Zareen Beenish Sultana U. Sultana, Azhar B. Khairuddin a. Grey wolf optimizer based placement and sizing of multiple distributed generation in the distribution system. *Energy*, September 2016.

[37] Ayad Mohammed Jabbar Ku Ruhana Ku-Mahamud. Adaptive grey wolf optimization algorithm with neighborhood search operations: An application for traveling salesman problem. *International Journal of Intelligent Engineering and Systems*, September 2021.

[38] Argel Bandala Ryan Rhay P. Vicerra Reagan L. Galvez, Elmer P. Dadios. Object detection in x-ray images using transfer learning with data augmentation. *Advanced Science Engineering and Information Technology*, December 2019.

[39] Jianqiang Li Tariq Mahmood Saeed Iqbal, Adnan N. Qureshi. On the analyses of medical images using traditional machine learning techniques and convolutional neural networks. *Springer*, 04 April 2023.

[40] Avijit Naskar Farhana Sultana Jaya Sil M.M. Hafizur Rahman Abu Sufian, Anirudha Ghosh. Bdnet: Bengali handwritten numeral digit recognition based on

densely connected convolutional neural networks. *Journal of King Saud University*, 2022.

[41] J. Plaza A. Plaza M.E. Paoletti, J.M. Haut. A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2022.

[42] Nicolas Jouandeau. Patrick Gikunda. State-of-the-art convolutional neural networks for smart farms. *core*, Jul 2017.

[43] Danlei Zhao Chunye Liu Zhengguang Liu. Yangkai Zhang, Mengke Wang. Early weed identification based on deep learning. *Smart Agricultural Technology*, 2023.

[44] Mark F. Hansen Wenhao Zhang Melvyn L. Smith Lyndon N. Smith, Arlo Byrne. Weed classification in grasslands using convolutional neural networks. *Worktribe*, 6 September 2019.

[45] Bronson Philippa Peter Ridd Jake C. Wood Jamie Johns Wesley Banks Benjamin Girgenti Owen Kenny James Whinney Brendan Calvert Mostafa Rahimi Azghadi Ronald D. White Alex Olsen, Dmitry A. Konovalov. Deepweeds: A multiclass weed species image dataset for deep learning. *Scientific Reports*, 14 February 2019.

[46] Henrik Skov Midtiby Mads Dyrmann, Henrik Karstoft. Plant species classification using deep convolutional neural network. *Biosystems Engineering.*, November 2016.

[47] Cyrill Stachniss Andres Milioto, Philipp Lottes. Real-time blob-wise sugar beets vs weeds classification for monitoring fields using convolutional neural networks. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences,International Conference on Unmanned Aerial Vehicles in Geomatics*, 4–7 September2017.

[48] Marija Popovic Raghav Khanna Frank Liebisch Juan Nieto Roland Siegwart Inkyu Sa, Zetao Chen. weednet: Dense semantic weed classification using multi-spectral images and mav for smart farming. *IEEE Robotics and Automation Letters*, 11 Sep 2017.

[49] Fan Luo Lihong Wang Chengsong Li Qi Niu Pei Wang, Yin Tang and Hui Li. Weed25: A deep learning dataset for weed identification. *frontiers in plant science*, 30 November 2022.

[50] Peter Kryger Jensen Mads Dyrmannand Henrik Skov Midtiby Thomas Mosgaard Giselsson, Rasmus Nyholm Jørgensen. A public image database for benchmark of plant seedling classification algorithms. *arXiv*, November 2017.

[51] Austin O'Brien Cherie Noteboom Martinson Ofori, Omar El-Gayar. A deep learning model compression and ensemble approach forweed detection. *Proceedings of the 55th Hawaii International Conference on System Sciences*, 2022.

[52] Omar El-Gayar Martinson Ofori. An approach for weed detection using cnns and transfer learning. *Proceedings of the 55th Hawaii International Conference on System Sciences*, 2021.

[53] Jungpil Shin Nawmee Razia Rahman, Dr. Md. Al Mehedi Hasan. Performance comparison of different convolutional neural network architectures for plant seedling classification. *International Conference on Advanced Information and Communication Technology*, 2020.

[54] Hardwari Lal Mandoria Prachi Chauhan and Alok Negi. Deep residual neural network for plant seedling image classification. *Agricultural Informatics*, 2021.

[55] Rani R. Bahia N. K Gupta, K. Plant-seedling classification using transfer learning-based deep convolutional neural networks. *International Journal of Agricultural and Environmental Information Systems*, 2020.

[56] Martinson Ofori Omar F. El-Gayar. Towards deep learning for weed detection: Deep convolutional neural network architectures for plant seedling classification. *Americas Conference on Information Systems*, 2020.

[57] Prakash R Patil Namratha Makanapura, Sujatha C and Padmashree Desai. Classification of plant seedlings using deep convolutional neural network architectures. *Journal of Physics: Conference Series*, 2021.

[58] Ashish Jain Deepak Gupta Miroslava Gono Radomir Gono Zbigniew Leonowicz and-Michał Jasiński Shamik Tiwari 1, Akhilesh Kumar Sharma. Iot-enabled model for

weed seedling classification: An application for smart agriculture. *AgriEngineering*, 29 January 2023.

[59] Alexander A. Hernandez Catherine R. Alimboyong. An improved deep neural network for classification of plant seedling images. *IEEE 15th International Colloquium on Signal Processing Its Applications (CSPA)*, 2019.

[60] Heba A. Elnemr. Convolutional neural network architecture for plant seedling classification. *International Journal of Advanced Computer Science and Applications*, 2019.

[61] Prince Sharma Abdel-Aziz Binguitcha-Fare. Crops and weeds classification using convolutional neural networks via optimization of transfer learning parameters. *International Journal of Engineering and Advanced Technology (IJEAT)*, June 2019.

[62] Yuan Rao 1Jinxin Liang Jie Teng Leiyang Fu, Shaowen Li and Quanling He. A novel heuristic target-dependent neural architecture search method with small samples. *PubMed Central*, 2022.

[63] D. Shiny Irene A. Umamageswari, N. Bharathiraja. A novel fuzzy c-means based chameleon swarm algorithm for segmentation and progressive neural architecture search for plant disease classification. *ScienceDirect*, 3 September 2021.

[64] Bhagirath Singh Chauhan. Grand challenges in weed management. *Frontiers in Agronomy*, 22 January 2020.

[65] Emerging agricultural technologies workgroup report and recommendations for ppdc review.

[66] Milan Trifkovic Miroslav Kuburic and Zarko Nestorovic. Efficient water use and greenhouse gas emission reduction in agricultural land use—the aspect of land consolidation. *Sustainability.*, 2022.

[67] Rakesh Kumar Mahendran Salman Naseer Muhammad Shafiq Parthasarathy Velusamy, Santhosh Rajendran and Jin-Ghoo Choi. Unmanned aerial vehicles (uav) in precision agriculture: Applications and challenges. *energies.*, 2021.

[68] Ruslan Suleymanov Ilyusya Gabbasova Azamat Suleymanov, Evgeny Abakumov and Mikhail Komissarov. The soil nutrient digital mapping for precision agriculture cases in the trans-ural steppe zone of russia using topographic attributes. *SPRS Int. J. Geo-Inf.*, 2021.

[69] Cagatay Catal Rik van der Tol Ngakan Nyoman Kutha Krisnawijaya, Bedir Tekinerdogan. Data analytics platforms for agricultural systems: A systematic literature review. *Elsevier*, 2022.