УДК 004.8:004.932.72
DOI https://doi.org/10.32782/2521-6643-2026-1-71.8

Зінченко А. Ю., кандидат технічних наук, доцент,
доцент кафедри математичних методів системного аналізу
Національного технічного університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
ORCID: 0000-0003-1586-3645

ЕВОЛЮЦІЙНА ОПТИМІЗАЦІЯ АРХІТЕКТУРИ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ У ЗАДАЧАХ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ

У даній роботі розглядається підхід до автоматизованої оптимізації архітектури згорткових нейронних мереж на основі еволюційних обчислень. Основну увагу зосереджено на застосуванні генетичних алгоритмів як одного з найбільш ефективних класів еволюційних алгоритмів, здатних здійснювати глобальний пошук оптимальних рішень у складних багатовимірних просторах параметрів. Запропонований підхід орієнтований на оптимізацію структурних параметрів згорткової нейронної мережі, зокрема кількості згорткових і повноз'єднаних шарів, числа нейронів у кожному шарі, кількості та розміру згорткових фільтрів, що безпосередньо впливають на узагальнювальні властивості моделі.

У роботі проведено аналіз основних принципів функціонування згорткових нейронних мереж та їх застосування в задачах класифікації зображень. Розглянуто базові компоненти архітектури згорткових нейронних мереж, включаючи згорткові шари, шари субдискретизації та повноз'єднані шари, а також проаналізовано вплив їх параметрів на точність класифікації. На основі цього аналізу було реалізовано базову згорткову нейронну мережу з архітектурою, підібраною безпосередньо розробником, та проведено експериментальне дослідження її роботи на тестових і реальних даних.

Крім цього, на основі основних принципів генетичних алгоритмів було розроблено власний генетичний алгоритм, призначений для оптимізації архітектури згорткової нейронної мережі. У якості функції пристосованості використано показники точності класифікації, що дозволяє безпосередньо орієнтувати еволюційний пошук на підвищення якості роботи моделі.

Розроблений генетичний алгоритм здійснює автоматизований пошук оптимальної архітектури нейронної мережі шляхом еволюційної модифікації параметрів її структури. У процесі еволюції відбувається поступове покращення характеристик мережі за рахунок відбору найбільш ефективних конфігурацій та генерації нових архітектур на основі операцій схрещування та мутації. Експериментально показано, що вже протягом десяти поколінь еволюції можливо отримати архітектуру згорткової нейронної мережі, яка забезпечує істотно кращі результати порівняно з початковою, вручну спроєктованою моделлю.

Ключові слова: *згорткові нейронні мережі, еволюційні алгоритми, генетичний алгоритм, інтелектуальний аналіз даних, обробка зображень.*

Zinchenko A. Yu. Evolutionary optimization of convolutional neural network architectures in intelligent data analysis tasks

This paper presents an approach to the automated optimization of convolutional neural network architectures based on evolutionary computation. The main focus is placed on the application of genetic algorithms as one of the most effective classes of evolutionary algorithms capable of performing global search for optimal solutions in complex high-dimensional parameter spaces. The proposed approach is aimed at optimizing the structural parameters of a convolutional neural network, including the number of convolutional and fully connected layers, the number of neurons in each layer, as well as the number and size of convolutional filters, which directly affect the generalization properties of the model.

The paper analyzes the fundamental principles of convolutional neural networks and their application to image classification tasks. The basic components of convolutional neural network architectures are examined, including convolutional layers, subsampling layers, and fully connected layers, and the influence of their parameters on classification accuracy is investigated. Based on this analysis, a baseline convolutional neural network with an architecture manually selected by the developer was implemented, and an experimental evaluation of its performance on both test and real-world datasets was conducted.

In addition, a proprietary genetic algorithm was developed on the basis of the fundamental principles of genetic algorithms to optimize the architecture of the convolutional neural network. Classification accuracy was employed as the fitness function, allowing the evolutionary search process to be directly guided toward improving the model's performance.

The developed genetic algorithm performs an automated search for an optimal neural network architecture through evolutionary modification of its structural parameters. During the evolutionary process, a gradual improvement of network characteristics is achieved by selecting the most effective configurations and generating new architectures through crossover and mutation operations. Experimental results demonstrate that within ten generations of evolution, it is possible to obtain a convolutional neural network architecture that significantly outperforms the initial manually designed model.

Key words: *convolutional neural networks, evolutionary algorithms, genetic algorithm, intelligent data analysis, image processing.*

Постановка проблеми. У сучасних задачах комп'ютерного зору та інтелектуального аналізу зображень згорткові нейронні мережі посідають провідне місце завдяки здатності автоматично виділяти інформативні ознаки та забезпечувати високу точність класифікації. Разом із цим ефективність використання згорткових нейронних мереж значною мірою визначається вибором їх архітектури, зокрема кількістю та типом шарів, числом нейронів, параметрами згорткових фільтрів і механізмами агрегування. Процес проєктування архітектури таких мереж традиційно здійснюється на основі емпіричного досвіду розробника або шляхом багаторазового ручного підбору гіперпараметрів, що є трудомістким, суб'єктивним і не гарантує досягнення оптимальних результатів, особливо при роботі з реальними, зашумленими або нерівномірно розподіленими даними.

Зазначені обмеження особливо загострюються у випадках застосування згорткових нейронних мереж до аналізу реальних зображень, які характеризуються високим рівнем шуму, варіативністю масштабів об'єктів, неоднорідними умовами освітлення та значною різницею між навчальними і робочими вибірками. За таких умов архітектура мережі, що демонструє високу точність на тестових даних, часто виявляється недостатньо стійкою при роботі з реальними даними, що призводить до зниження узагальнювальної здатності та погіршення практичних результатів класифікації.

Крім того, зростання складності сучасних задач комп'ютерного зору супроводжується необхідністю одночасного врахування великої кількості структурних параметрів нейронної мережі, таких як глибина моделі, конфігурація згорткових шарів, кількість та розміри фільтрів, способи агрегації ознак і параметри повноз'єднаних шарів. Простір можливих архітектур при цьому має комбінаторний характер, що унеможливлює ефективний перебір усіх варіантів за допомогою класичних методів оптимізації або ручного налаштування.

Існуючі підходи до автоматизації вибору архітектури згорткових нейронних мереж, зокрема методи градієнтної оптимізації гіперпараметрів або шаблонні архітектури, не завжди забезпечують достатню гнучкість та адаптивність до специфіки конкретної задачі та структури даних. У зв'язку з цим актуальною є проблема розроблення таких методів оптимізації архітектури нейронних мереж, які б дозволяли здійснювати глобальний пошук ефективних структурних рішень у складних багатовимірних просторах параметрів без жорстких припущень щодо форми функції якості.

Одним із перспективних напрямів розв'язання зазначеної проблеми є використання еволюційних алгоритмів, зокрема генетичних алгоритмів, які базуються на принципах природного добору та дозволяють ефективно досліджувати простір можливих архітектур нейронних мереж. Застосування генетичних алгоритмів створює передумови для автоматизованого синтезу архітектури згорткових нейронних мереж, адаптованої до конкретних умов задачі класифікації зображень, з урахуванням вимог до точності, стійкості та узагальнювальної здатності моделей.

Таким чином, актуальною є науково-прикладна задача розроблення та дослідження генетичного алгоритму для оптимізації архітектури згорткових нейронних мереж, здатного підвищити ефективність класифікації зображень, особливо при роботі з реальними даними, та зменшити залежність якості результатів від суб'єктивного вибору параметрів розробником.

Аналіз останніх досліджень і публікацій. Аналіз останніх досліджень і публікацій свідчить, що впродовж останніх років значна увага науковців зосереджена на розвитку методів глибинного навчання для задач комп'ютерного зору, зокрема класифікації та розпізнавання зображень. Базові принципи побудови згорткових нейронних мереж і їх ефективність у широкому спектрі прикладних задач детально висвітлені в фундаментальній роботі LeCun та ін. [1], де показано, що глибинні нейронні мережі здатні автоматично формувати ієрархічні ознаки з даних, забезпечуючи високу точність класифікації.

Подальший розвиток згорткових нейронних мереж пов'язаний із роботою Krizhevsky та ін. [2], у якій вперше було продемонстровано суттєвий прорив у задачі класифікації зображень за допомогою глибокої згорткової архітектури. Результати цього дослідження підтвердили визначальну роль архітектури нейронної мережі у досягненні високої точності та стимулювали активні дослідження, спрямовані на пошук ефективних структурних рішень для різних класів задач.

Разом із тим, проєктування архітектури нейронних мереж традиційно залишається складним і трудомістким процесом. Класичні підходи до оптимізації параметрів мережі часто ґрунтуються на ручному підборі або локальних методах оптимізації, що не дозволяє ефективно досліджувати простір можливих архітектур. У цьому контексті особливу увагу привертають еволюційні алгоритми, теоретичні основи яких були закладені в роботі Holland [3]. Генетичні алгоритми, що імітують механізми природного добору, продемонстрували здатність знаходити ефективні рішення в умовах складних багатовимірних просторів параметрів.

Подальший розвиток ідей еволюційної оптимізації нейронних мереж представлено в роботі Stanley та Miikkulainen [4], де запропоновано підхід до еволюційного синтезу топологій нейронних мереж. Авторами показано, що еволюційне нарощування структури мережі дозволяє отримувати архітектури з покращеними узагальнювальними властивостями без жорстких обмежень на початкову конфігурацію.

Безпосереднє поєднання згорткових нейронних мереж і генетичних алгоритмів розглянуто в роботі Xie та Yuille [5], де запропоновано підхід Genetic CNN, що дозволяє автоматизувати пошук архітектур згорткових нейронних мереж. Результати дослідження демонструють ефективність еволюційних методів для

оптимізації структури мереж у задачах комп'ютерного зору та підтверджують перспективність такого підходу порівняно з ручним проєктуванням архітектур.

Узагальнення сучасних підходів до автоматизованого проєктування нейронних мереж наведено в оглядовій роботі Elsken та ін. [6], де розглядаються методи Neural Architecture Search, включаючи еволюційні, градієнтні та комбіновані підходи. Автори відзначають, що еволюційні алгоритми залишаються одним із найбільш універсальних інструментів для оптимізації архітектури нейронних мереж, особливо у випадках, коли цільова функція є складною або недиференційовною.

В роботі [7] запропонований підхід для покращення розпізнавання структурованого тексту, шляхом інтеграції нейронної мережі YOLO з технологією OCR, що значно підвищило ефективність системи розпізнавання тексту, забезпечуючи більш точне виявлення текстових об'єктів для їх подальшого розпізнавання.

Таким чином, аналіз сучасних наукових публікацій показує, що хоча згорткові нейронні мережі широко застосовуються в задачах класифікації зображень, проблема автоматизованої оптимізації їх архітектури залишається актуальною. Це обумовлює доцільність подальших досліджень, спрямованих на розроблення генетичних алгоритмів для оптимізації архітектури згорткових нейронних мереж з метою підвищення їх ефективності при роботі з реальними даними.

Мета статті: розроблення та дослідження генетичного алгоритму для автоматизованої оптимізації архітектури згорткових нейронних мереж у задачах класифікації зображень, а також у проведенні експериментальної оцінки впливу еволюційної оптимізації структурних параметрів нейронної мережі на точність класифікації при роботі з тестовими та реальними даними.

Виклад основного матеріалу. Особливості застосування згорткових нейронних мереж для інтелектуального аналізу зображень. Згорткові нейронні мережі є одним із базових інструментів сучасного комп'ютерного зору та інтелектуального аналізу зображень. Їх широке застосування зумовлене здатністю автоматично виявляти просторові та ієрархічні ознаки без необхідності ручного формування дескрипторів. На відміну від класичних методів обробки зображень, що базуються на фіксованих фільтрах і евристичних правилах, згорткові нейронні мережі забезпечують адаптивне навчання ознак безпосередньо з даних, що дозволяє ефективно працювати з різноманітними типами зображень та складними умовами зйомки.

Інтелектуальний аналіз зображень передбачає вирішення таких задач, як класифікація, сегментація, виявлення об'єктів та розпізнавання візуальних патернів. У межах цих задач згорткові нейронні мережі використовують принцип локальних рецептивних полів, відповідно до якого кожен нейрон обробляє лише обмежену ділянку вхідного зображення. Такий підхід дозволяє значно зменшити кількість параметрів моделі, знизити обчислювальну складність та забезпечити інваріантність до зсувів і масштабування об'єктів.

Типова архітектура згорткової нейронної мережі складається зі згорткових шарів, шарів агрегації (subsampling або pooling) та повноз'єднаних шарів. Згорткові шари відповідають за виділення локальних ознак, таких як краї, контури та текстури, тоді як на більш глибоких рівнях формуються складніші абстрактні представлення, що відповідають окремим об'єктам або їх частинам. Шари агрегації зменшують просторову розмірність ознак, підвищують стійкість мережі до шумів і зменшують ризик перенавчання. Повноз'єднані шари інтегрують виділені ознаки та формують фінальне рішення щодо класу зображення.

Ефективність застосування згорткових нейронних мереж значною мірою залежить від вибору їх архітектури. Кількість шарів, число нейронів у кожному шарі, розміри згорткових фільтрів і способи агрегації безпосередньо впливають на здатність мережі до узагальнення та її продуктивність при роботі з реальними даними. Невдало підібрана архітектура може призводити до перенавчання або, навпаки, до недостатньої складності моделі, що знижує точність класифікації.

Особливої актуальності ця проблема набуває в задачах аналізу реальних зображень, які характеризуються наявністю шумів, варіативністю освітлення, різними масштабами об'єктів та неповнотою навчальних вибірок. У таких умовах архітектура, що демонструє високі показники на тестових даних, може втрачати ефективність при практичному застосуванні. Це зумовлює необхідність використання методів автоматизованої оптимізації архітектури згорткових нейронних мереж.

Одним із перспективних підходів до вирішення цієї задачі є застосування еволюційних алгоритмів, які дозволяють здійснювати глобальний пошук ефективних архітектур у багатовимірному просторі параметрів. Поєднання згорткових нейронних мереж з генетичними алгоритмами створює можливість автоматизованого синтезу структур мереж, адаптованих до конкретних умов задачі інтелектуального аналізу зображень, що підвищує точність класифікації та стійкість моделей до змін вхідних даних.

Нижче наведено приклад реалізації згорткової нейронної мережі для класифікації зображень з використанням мови програмування Python та високорівневого інтерфейсу Keras, який функціонує поверх обчислювального фреймворку TensorFlow:

Опис генетичного алгоритму оптимізації архітектури. Для автоматизованого пошуку ефективної архітектури згорткової нейронної мережі у даному дослідженні використовується генетичний алгоритм, який дозволяє здійснювати еволюційну оптимізацію структурних параметрів моделі без необхідності аналітичного опису залежності між архітектурою та якістю класифікації.

```
In [23]: import numpy as np
         import random
         import matplotlib.pyplot as plt
         from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten
```

Завантаження набору даних для тренування моделі

```
In [7]: X_train = np.loadtxt('input.csv', delimiter = ',')
        Y_train = np.loadtxt('labels.csv', delimiter = ',')

        X_test = np.loadtxt('input_test.csv', delimiter = ',')
        Y_test = np.loadtxt('labels_test.csv', delimiter = ',')
```

```
In [20]: X_train = X_train.reshape(len(X_train), 100, 100, 3)
         Y_train = Y_train.reshape(len(Y_train), 1)

         X_test = X_test.reshape(len(X_test), 100, 100, 3)
         Y_test = Y_test.reshape(len(Y_test), 1)

         X_train = X_train/255.0
         X_test = X_test/255.0
```

```
In [14]: print("Shape of X_train: ", X_train.shape)
         print("Shape of Y_train: ", Y_train.shape)
         print("Shape of X_test: ", X_test.shape)
         print("Shape of Y_test: ", Y_test.shape)
```

```
Shape of X_train:  (2000, 100, 100, 3)
Shape of Y_train:  (2000, 1)
Shape of X_test:  (400, 100, 100, 3)
Shape of Y_test:  (400, 1)
```

```
In [28]: idx = random.randint(0, len(X_train))
         plt.imshow(X_train[idx, :])
         plt.show()
```

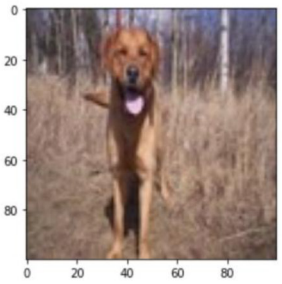


Рис. 1. Реалізація згорткової нейронної мережі для класифікації зображень

Генетичний алгоритм розглядає множину можливих архітектур нейронних мереж як популяцію особин $P = \{a_1, a_2, \ldots, a_N\}$, де кожна особина a_i відповідає окремій архітектурі згорткової нейронної мережі, а N – розмір популяції.

Архітектура нейронної мережі подається у вигляді хромосоми, яка складається з набору генів: $a_i = (g_1, g_2, \ldots, g_k)$, де кожен ген g_j кодує певний структурний параметр мережі, зокрема:

- кількість згорткових шарів;
- кількість фільтрів у кожному згортковому шарі;
- розмір ядра згортки;
- параметри шару субдискретизації;
- кількість нейронів у повноз'єднаному шарі.

Таке подання забезпечує можливість еволюційної модифікації архітектури при збереженні її структурної коректності.

Для оцінювання якості кожної особини використовується функція пристосованості, що базується на точності класифікації нейронної мережі після навчання на навчальній вибірці. Формально функція пристосованості визначається як $F(a_i) = Acc(a_i)$, де $Acc(a_i)$ – точність класифікації мережі з архітектурою a_i на валідаційній або тестовій вибірці.

У процесі еволюції максимізується значення функції пристосованості $a^* = \arg\max_{a_i \in P} F(a_i)$. .

На кожному поколінні генетичного алгоритму здійснюється відбір особин із найвищими значеннями функції пристосованості. Відібрані архітектури використовуються для формування нового покоління за допомогою операторів схрещування та мутації.

Конфігурація та тренування моделі

In [30]:
```
model = Sequential([
    Conv2D(32, (3,3), activation = 'relu', input_shape = (100, 100, 3)),
    MaxPooling2D((2,2)),

    Conv2D(32, (3,3), activation = 'relu'),
    MaxPooling2D((2,2)),

    Flatten(),
    Dense(64, activation = 'relu'),
    Dense(1, activation = 'sigmoid')
])
```

In [31]:
```
model = Sequential()

model.add(Conv2D(32, (3,3), activation = 'relu', input_shape = (100, 100, 3)))
model.add(MaxPooling2D((2,2)))

model.add(Conv2D(32, (3,3), activation = 'relu'))
model.add(MaxPooling2D((2,2)))

model.add(Flatten())
model.add(Dense(64, activation = 'relu'))
model.add(Dense(1, activation = 'sigmoid'))
```

In [32]:
```
model.compile(loss = 'binary_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
```

In [32]:
```
model.compile(loss = 'binary_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
```

In [34]:
```
model.fit(X_train, Y_train, epochs = 5, batch_size = 64)
```

```
Epoch 1/5
32/32 [==============================] - 30s 934ms/step - loss: 0.3599 - accuracy: 0.8475
Epoch 2/5
32/32 [==============================] - 29s 911ms/step - loss: 0.2853 - accuracy: 0.8800
Epoch 3/5
32/32 [==============================] - 30s 946ms/step - loss: 0.2196 - accuracy: 0.9160
Epoch 4/5
32/32 [==============================] - 32s 994ms/step - loss: 0.1694 - accuracy: 0.9395
Epoch 5/5
32/32 [==============================] - 30s 931ms/step - loss: 0.1247 - accuracy: 0.9610
```

Out[34]:
```
<keras.callbacks.History at 0x1de37b90850>
```

In [35]:
```
model.evaluate(X_test, Y_test)
```

```
13/13 [==============================] - 4s 149ms/step - loss: 0.9762 - accuracy: 0.6550
```

Out[35]:
```
[0.9762333631515503, 0.6549999713897705]
```

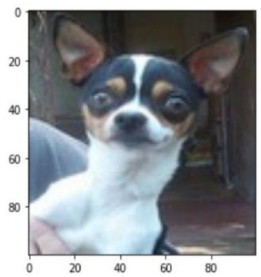
Тест моделі на реальних даних

In [43]:
```
idx2 = random.randint(0, len(Y_test))
plt.imshow(X_test[idx2, :])
plt.show()

y_pred = model.predict(X_test[idx2, :].reshape(1, 100, 100, 3))
y_pred = y_pred > 0.5

if(y_pred == 0):
    pred = 'dog'
else:
    pred = 'cat'

print("Our model says it is a :", pred)
```




```
Our model says it is a : dog
```

Рис. 2. Конфігурація, тренування та тестування моделі на реальних даних

Оператор схрещування реалізується шляхом обміну частинами хромосом між двома батьківськими особинами: $a_{child} = Crossover(a_{parent_1}, a_{parent_2})$, що дозволяє поєднувати структурні характеристики різних архітектур.

Оператор мутації полягає у випадковій зміні значень окремих генів з малою ймовірністю p_m: $g'_j = \begin{cases} rand(G_j), \text{ імовірність} - p_m \\ g_j, \text{ інакше} \end{cases}$, де $rand(G_j)$ – випадкове допустиме значення відповідного параметра.

Мутація сприяє підтриманню різноманітності популяції та зменшує ймовірність передчасної збіжності алгоритму до локального оптимуму.

Еволюційний процес триває протягом фіксованої кількості поколінь або до досягнення заданого значення функції пристосованості. У результаті роботи генетичного алгоритму формується архітектура згорткової нейронної мережі, яка демонструє покращені показники точності класифікації порівняно з початковими або вручну спроєктованими моделями.

Наведемо реалізацію генетичного алгоритму для оптимізації архітектури згорткових нейронних мереж.

```
In [7]: class CNN(Sequential):
            def __init__(self,nfilters,sfilters):
                super().__init__()
                tensorflow.random.set_seed(0)
                self.add(Conv2D(nfilters[0],kernel_size=(sfilters[0],sfilters[0]),padding='same',activation='relu',
                                input_shape=(112,92,1)))
                self.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
                self.add(Conv2D(nfilters[1],kernel_size=(sfilters[1],sfilters[1]),padding='same',activation='relu'))
                self.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
                self.add(Conv2D(nfilters[2],kernel_size=(sfilters[2],sfilters[2]),padding='same',activation='relu'))
                self.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
                self.add(Flatten())
                self.add(Dropout(0.3))
                self.add(Dense(512,activation='relu'))
                self.add(Dense(40,activation='softmax'))
                self.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

Рис. 3. Лістинг коду архітектури згорткової нейронної мережі

По завершенню виконання алгоритму можемо оцінити точність розпізнавання зображень:

Генетичний алгоритм запускався з такими параметрами:

Враховуючи стохастичний характер ініціалізації параметрів нейронної мережі у середовищі TensorFlow, максимальна точність класифікації на реальних даних становила приблизно 97,5 %, що проілюстровано на графіку.

На рис. 7 зображено динаміку зміни функції пристосованості залежно від номера покоління генетичного алгоритму. В якості показника пристосованості використано точність класифікації. Отримані результати демонструють ефективність еволюційного пошуку, оскільки зі збільшенням кількості поколінь спостерігається стале зростання значення fitness та досягнення близького до оптимального рівня вже протягом перших десяти поколінь.

Після десяти поколінь еволюційної оптимізації генетичним алгоритмом було знайдено архітектуру згорткової нейронної мережі з шістьма оптимізованими шарами (без урахування вхідного шару), для яких кількість нейронів дорівнювала 66, 40, 88, 3, 1 та 1 відповідно.

Висновки. У роботі реалізовано згорткову нейронну мережу для задачі класифікації зображень та проведено експериментальну оцінку її ефективності на тестових і реальних даних. Для базової моделі архітектура нейронної мережі була сформована розробником на основі емпіричних міркувань, що дозволило досягти високої точності класифікації на тестовій вибірці, однак виявило суттєве зниження точності при роботі з реальними даними.

У межах дослідження реалізовано генетичний алгоритм для автоматизованої оптимізації архітектури згорткової нейронної мережі. Запропонований алгоритм забезпечує еволюційний пошук ефективних структурних конфігурацій шляхом оптимізації кількості згорткових і повноз'єднаних шарів, числа нейронів у кожному шарі, а також кількості та розмірів згорткових фільтрів. У процесі еволюції протягом десяти поколінь було отримано архітектуру згорткової нейронної мережі з покращеними узагальнювальними властивостями.

Порівняльний аналіз результатів класифікації показав, що згорткова нейронна мережа з архітектурою, оптимізованою за допомогою розробленого генетичного алгоритму, забезпечує суттєве підвищення точності при роботі з реальними даними. Зокрема, точність класифікації зросла з 65 % для базової архітектури до 97–99 % для оптимізованої моделі, що свідчить про високу ефективність застосування еволюційної оптимізації структури нейронної мережі.

Подальші дослідження доцільно спрямувати на розширення функціональних можливостей розробленого генетичного алгоритму шляхом включення багатокритеріальної оптимізації, зокрема з урахуванням

```
In [22]: class Genetic:

             def __init__(self,pop_size,nlayers,max_nfilters,max_sfilters):
                 self.pop_size = pop_size
                 self.nlayers = nlayers
                 self.max_nfilters = max_nfilters
                 self.max_sfilters = max_sfilters
                 self.max_acc = 0
                 self.best_arch = np.zeros((1,6))
                 self.gen_acc = []

             def generate_population(self):
                 np.random.seed(0)
                 pop_nlayers = np.random.randint(1,self.max_nfilters,(self.pop_size,self.nlayers))
                 pop_sfilters = np.random.randint(1,self.max_sfilters,(self.pop_size,self.nlayers))
                 pop_total = np.concatenate((pop_nlayers,pop_sfilters),axis=1)
                 return pop_total

             def select_parents(self,pop,nparents,fitness):
                 parents = np.zeros((nparents,pop.shape[1]))
                 for i in range(nparents):
                     best = np.argmax(fitness)
                     parents[i] = pop[best]
                     fitness[best] = -99999
                 return parents
```

```
def crossover(self,parents):
    nchild = self.pop_size - parents.shape[0]
    nparents = parents.shape[0]
    child = np.zeros((nchild,parents.shape[1]))
    for i in range(nchild):
        first = i % nparents
        second = (i+1) % nparents
        child[i,:2] = parents[first][:2]
        child[i,2] = parents[second][2]
        child[i,3:5] = parents[first][3:5]
        child[i,5] = parents[second][5]
    return child

def mutation(self,child):
    for i in range(child.shape[0]):
        val = np.random.randint(1,6)
        ind = np.random.randint(1,4) - 1
        if child[i][ind] + val > 100:
            child[i][ind] -= val
        else:
            child[i][ind] += val
        val = np.random.randint(1,4)
        ind = np.random.randint(4,7) - 1
        if child[i][ind] + val > 20:
            child[i][ind] -= val
        else:
            child[i][ind] += val
    return child
```

```
def fitness(self,pop,X,Y,epochs):
    pop_acc = []
    for i in range(pop.shape[0]):
        nfilters = pop[i][0:3]
        sfilters = pop[i][3:]
        model = CNN(nfilters,sfilters)
        H = model.fit(X,Y,batch_size=32,epochs=epochs)
        acc = H.history['accuracy']
        pop_acc.append(max(acc)*100)
    if max(pop_acc) > self.max_acc:
        self.max_acc = max(pop_acc)
        self.best_arch = pop[np.argmax(pop_acc)]
    self.gen_acc.append(max(pop_acc))
    return pop_acc
```

Рис. 4. Реалізація генетичного алгоритму для оптимізації архітектури згорткової нейронної мережі

не лише точності класифікації, але й обчислювальної складності, часу навчання та обсягу використаних ресурсів.

Перспективним напрямом є адаптація запропонованого підходу до інших задач комп'ютерного зору, таких як виявлення об'єктів і сегментація зображень, а також дослідження його ефективності при роботі з більшими та більш різнорідними наборами реальних даних.

Доцільним є також дослідження можливості поєднання генетичного алгоритму з іншими методами автоматизованого проєктування нейронних мереж, зокрема підходами Neural Architecture Search, що може забезпечити подальше підвищення ефективності та стабільності результатів

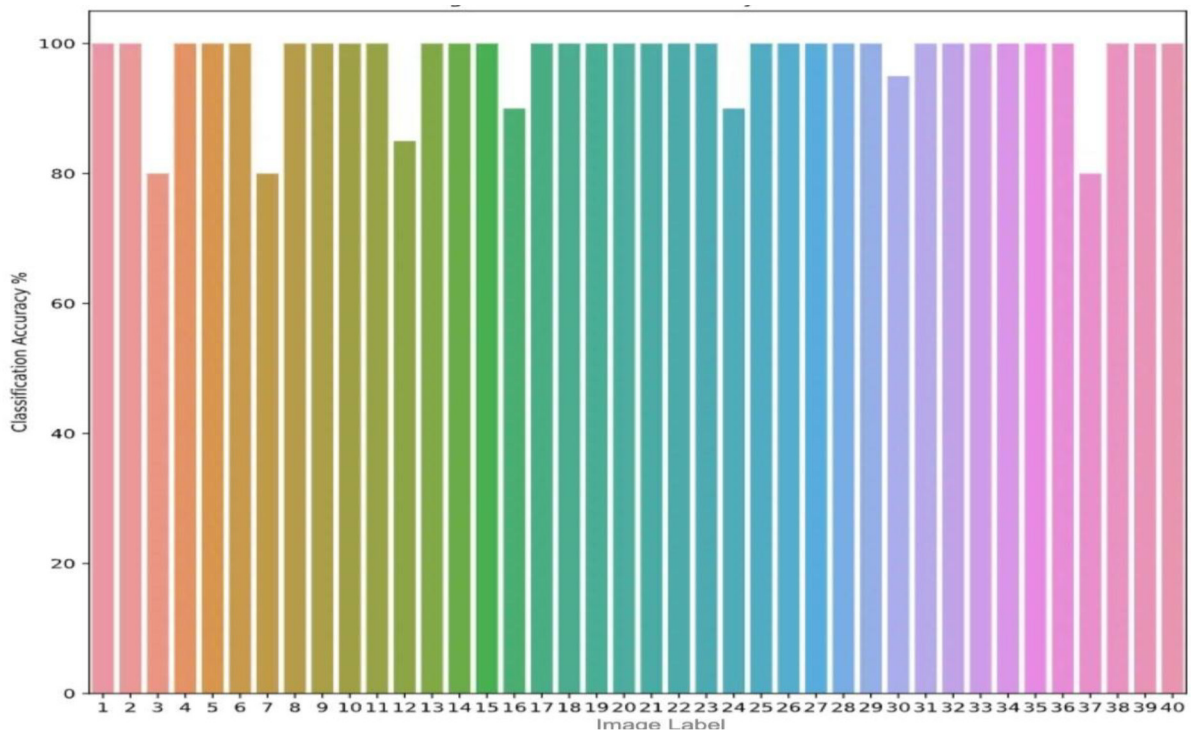


Рис. 5. Точність розпізнавання зображень оптимізованою згортковою нейронною мережею за допомогою генетичного алгоритму

Individuals Per Population	10
Number of Generations	10
Maximum No. Of Filters	100
Maximum size of Kernels	20
No. Of Parents and Children	5 each

Рис. 6. Параметри ініціалізації генетичного алгоритму

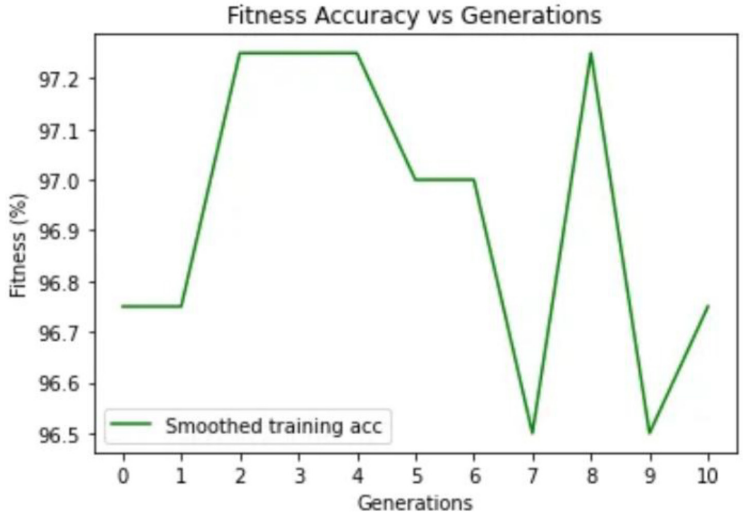


Рис. 7. Динаміку зміни функції пристосованості залежно від номера покоління генетичного алгоритму

Список використаних джерел:

1. LeCun Y., Bengio Y., Hinton G. Deep Learning. *Nature*. 2015. Vol. 521. P. 436–444. DOI: https://doi.org/10.1038/nature14539 (дата звернення: 05.01.2026)

2. Krizhevsky A., Sutskever I., Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*. 2012. Vol. 25. P. 1097–1105. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (дата звернення: 05.01.2026)

3. Holland J. H. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Ann Arbor: *University of Michigan Press*, 1992. 183 p. DOI: https://doi.org/10.7551/mitpress/1090.001.0001 (дата звернення: 05.01.2026)

4. Stanley K. O., Miikkulainen R. Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation*. 2002. Vol. 10, No. 2. P. 99–127. DOI: https://doi.org/10.1162/106365602320169811 (дата звернення: 05.01.2026)

5. Xie L., Yuille A. Genetic CNN. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017. P. 1379–1388. DOI: https://doi.org/10.1109/ICCV.2017.154 (дата звернення: 05.01.2026)

6. Elsken T., Metzen J. H., Hutter F. Neural Architecture Search: A Survey. *Journal of Machine Learning Research*. 2019. Vol. 20, No. 55. P. 1–21. URL: https://www.jmlr.org/papers/volume20/18-598/18-598.pdf (дата звернення: 05.01.2026)

7. Зінченко, А. Ю., Хайдуров, В. В. (2024). Покращення розпізнавання структурованого тексту нейронною мережею YOLO. *Системи та технології*, вип. 68 (2), 2024, с. 23–31. DOI: https://doi.org/10.32782/2521-6643-2024-2-68.3 (дата звернення: 05.01.2026)

References:

1. LeCun Y., Bengio Y., Hinton G. Deep Learning. *Nature*. 2015. Vol. 521. P. 436–444. DOI: https://doi.org/10.1038/nature14539 (accessed on: 05.01.2026).

2. Krizhevsky A., Sutskever I., Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*. 2012. Vol. 25. P. 1097–1105. Retrieved from: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (accessed on: 05.01.2026).

3. Holland J. H. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Ann Arbor: *University of Michigan Press*, 1992. 183 p. DOI: https://doi.org/10.7551/mitpress/1090.001.0001 (accessed on: 05.01.2026).

4. Stanley K. O., Miikkulainen R. Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation*. 2002. Vol. 10, No. 2. P. 99–127. DOI: https://doi.org/10.1162/106365602320169811 (accessed on: 05.01.2026).

5. Xie L., Yuille A. Genetic CNN. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017. P. 1379–1388. DOI: https://doi.org/10.1109/ICCV.2017.154 (accessed on: 05.01.2026).

6. Elsken T., Metzen J. H., Hutter F. Neural Architecture Search: A Survey. *Journal of Machine Learning Research*. 2019. Vol. 20, No. 55. P. 1–21. Retrieved from: https://www.jmlr.org/papers/volume20/18-598/18-598.pdf (accessed on: 05.01.2026).

7. Zinchenko A. Yu., Khaidurov, V. V. Pokrashchennia rozpiznavannia strukturovanoho tekstu neironnoiu merezheiu YOLO [Improving structured text recognition with the YOLO neural network]. *Systemy ta tekhnolohii*. vyp. 68 (2), 2024, s. 23–31 DOI: https://doi.org/10.32782/2521-6643-2024-2-68.3 (accessed on: 05.01.2026).

Дата першого надходження статті до видання: 30.11.2025
Дата прийняття статті до друку після рецензування: 22.12.2025
Опубліковано: 00.00.2025