КІБЕРБЕЗПЕКА: освіта, наука, техніка

CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE

№ 3 (11), 2021

ISSN 2663 - 4023

DOI 10.28925/2663-4023.2021.11.110123

УДК 004.89: 004.656

**Кузніченко Світлана Дмитрівна**
к.геогр.н., доц., доцент
місце роботи: Одеський державний екологічний університет, Одеса, Україна
ORCID ID: 0000-0001-7982-1298
*skuznichenko@gmail.com*

**Терещенко Тетяна Михайлівна**
к.т.н., доц., доцент
місце роботи: Одеський державний екологічний університет, Одеса, Україна
ORCID ID: 0000-0001-7691-6996
*tereshchenko.odessa@gmail.com*

**Бучинська Ірина Вікторівна**
PhD, асистент
місце роботи: Одеський державний екологічний університет, Одеса, Україна
ORCID ID: 0000-0002-0393-2781
*buchinskayaira@gmail.com*

**Клепатська Вікторія Вікторівна**
асистент
місце роботи: Одеський державний екологічний університет, Одеса, Україна
ORCID ID: 0000-0001-5613-6546
*victoria.klepatska@gmail.com*

# ПРОГРАМНА СИСТЕМА КЛАСИФІКАЦІЇ ПАРАМЕТРІВ ЕЛЕКТРОННОГО БЛОКУ КЕРУВАННЯ ДВИГУНОМ НА ОСНОВІ ХАРАКТЕРИЗАТОРІВ ТА БАЗИ ЗНАНЬ

**Анотація.** У статті розглядаються питання підвищення ефективності процесу класифікації карт електронних блоків керування двигуном автомобіля. Проведено аналіз існуючого програмного забезпечення для редагування калібрувальних таблиць в електронних блоках керування, яке має інструменти визначення калібровок і розпізнавання даних. Обмеження використання таких програмних продуктів пов'язані з невеликою кількістю заданих класів калібрувальних таблиць та низькою швидкістю обробки даних. Аналіз результатів проведеного тестування методів класифікації за допомогою спектрального розкладу показав, що система на основі такого методу вимагає складних перетворень результатів спектрального розкладу. Застосування спектрального розкладу для вирішення задачі класифікації можливо в разі визначення деяких характеристик вхідних даних і використанні їх в якості даних для класифікації. Було розроблено алгоритм класифікації даних, який використовує характеризатори для обчислення чітко визначеної характеристики вхідної матриці. Програмний комплекс для реалізації розробленого алгоритму було побудовано з використанням платформи .NET Framework та мови програмування C #. Тестування працездатності системи класифікації проводилося за допомогою розробленої програмної системи на невеликій вибірці карт. Результати попереднього тестування показали, що система вірно визначає клас наданої карти після навчання. Подальше тестування на сімействі блоків автомобілів Mercedes-Benz Bosch EDC16C31/EDC16CP31 показало, що у випадках з великою кількістю навчальних образів результат задовольняє вимогам. Проведене тестування дозволило визначити оптимальну кількість образів для навчання та необхідний для цього час.

**Ключові слова:** класифікація з навчанням; розпізнавання образів; база знань; програмна система; електронний блок керування двигуном.

ISSN 2663 - 4023

## ВСТУП

**Постановка проблеми.** Інформаційні технології широко використовуються в різних сферах людської діяльності. Сьогодні складно уявити промисловість, медицину, освіту, науку без використання інформаційних систем, комп'ютерних мереж і програмного забезпечення. Технічний прогрес і стрімкий розвиток промисловості ставлять перед сучасним суспільством завдання збереження і раціонального використання природних ресурсів, забезпечення екологічної безпеки людства та захисту навколишнього середовища від наслідків антропогенної діяльності. Підвищення рівня життя в розвинених країнах призвело до різкого зростання кількості автомобільного транспорту. Державні органи країн і громадські організації постійно працюють в напрямку зміни екологічних стандартів, мотивують і змушують автомобільну промисловість розвивати і впроваджувати технології нейтралізації відпрацьованих газів [1]. У цих умовах заводи-виробники змушені випускати велику кількість варіантів програмного забезпечення, яке стандартизується під різні умови експлуатації автомобільного транспорту. При цьому врахувати всі можливі умови досить складно. Тому що кількість варіантів, які можливо встановити в настройках програми при випуску автомобіля, обмежена. Це стало причиною того, що сторонні компанії почали розвивати способи зміни калібрувальних параметрів систем уприскування для більш точного налаштування двигуна, для зняття обмежень, накладених заводом-виробником на ходові характеристики двигуна. Безліч компаній по всьому світу займаються розробкою програмного забезпечення та інструментів для коригування програмного забезпечення електронних блоків управління автомобіля.

**Аналіз останніх досліджень і публікацій.** Існує кілька програмних продуктів, які створені для редагування калібрувальних таблиць в електронних блоках керування автомобілем. Можна виділити два з них, в які вбудована система визначення калібровок і розпізнавання даних: WinOLS і Swiftec [2], [3]. WinOLS є лідером на ринку програмного забезпечення для редагування калібровок і орієнтована тільки для досвідчених фахівців [3]. Це програмне забезпечення не надає інформації про калібрування, а надає математичні алгоритми для пошуку невідомих наборів калібровок для деяких виробників. Такий підхід дещо прискорює роботу, але при цьому швидкість роботи автоматичного пошуку калібровок досить низька. Swiftec більш новий продукт на ринку і має ряд переваг в порівнянні з WinOLS. В нього вбудовані аналогічні алгоритми пошуку невідомих наборів калібровок для деяких виробників електронних блоків керування, а також реалізована класифікація калібровок для окремих поколінь блоків. Результати роботи системи класифікації досить неточні і дуже прив'язані до конкретного виду блоку керування. У разі помилки вибору блоку система може не визначити нічого або визначити зовсім не те, що необхідно. Кількість калібрувань визначається заздалегідь написаними словниками, які розробники додають в оновленнях. Аналіз існуючого програмного забезпечення показав, що системи такого типу не є адаптивними і такими, що здатні до навчання. Класифікація відбувається за чітко визначеними алгоритмами або словниками, які створюються розробниками. Можна виділити наступні недоліки сучасних систем розпізнавання карт, які представлені на ринку:

– відсутність самонавчання, адаптивності;
– відсутність можливості самостійно створювати словники;
– низька швидкість роботи;
– невелика кількість наданих класів калібрувальних таблиць.

Аналіз публікацій в цій галузі досліджень показав, що найбільш поширеною системою класифікації в подібних системах є штучні нейронні мережі (ШНМ) [4-7]. Використання нейронних мереж передбачає, що на вхід подаються підготовлені дані. Обробка цих даних вимагає складних обчислювальних операцій, що помітно знижує швидкість виконання класифікації. При цьому ШНМ складно піддаються навчанню, результат навчання залежить від кількості і якості образів в навчальній вибірці, складно вирішуються завдання класифікації, і здатні видавати образи одного класу за образи іншого класу, а також швидко втрачати накопичений досвід при суперечливих вхідних даних. В представленому дослідженні використання ШНМ в якості основного механізму розпізнавання вимагає створення своєї топології мережі з додатковими перетвореннями вхідних даних, а також розробку власного механізму навчання мережі. Це вимагає великих трудовитрат, при цьому результат не задовольняє вимогам розширюваності і самонавчання системи в процесі роботи. Крім ШНМ в процесах розпізнавання образів використовують спектральний розклад сигналу і подальший аналіз спектрів сигналу з метою отримання даних про характеристики сигналу. Існує кілька стандартних математичних методів розкладу сигналу на спектри, такі як перетворення Фур'є і вейвлет-перетворення [8-10]. Аналіз результатів тестування показав, що система на основі подібних методів вимагає складних перетворень результатів спектрального розкладу. Застосування спектрального розкладу для вирішення задачі класифікації можливо в разі визначення деяких характеристик вхідних даних і використанні їх в якості даних для класифікації. Використання тільки спектрального розкладу в задачі, що розглядається, вимагає складних обчислень, і має непередбачуваний результат.

**Метою статті** є розробка системи класифікації параметрів електронного блоку керування двигуном автомобіля, яка забезпечує високу швидкість обробки вхідних даних, має можливості для навчання і серіалізації даних навчання в окремі словники бази знань.

## ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ

В результаті читання програми з пам'яті блоку керування двигуном вилучається бінарний файл – дамп, який містить в собі область калібрувань, дані з якої необхідно розпізнати. На рис. 1 кожне значення в дампі представлено у вигляді точок на двомірному полі, де по горизонталі відкладені адреси в пам'яті, по вертикалі – значення.

При цьому візуально проглядаються карти калібровок. Але навіть досвідченому фахівцю не завжди просто визначити, де саме починається кожна з карт, яких вона розмірів, і яка це характеристика. Для визначення всіх цих показників необхідна система класифікації.

Перед тим, як класифікувати карти, необхідно надати на вхід системи сформовану матрицю значень. Для цього необхідно розбити дамп на таблиці. Це завдання вирішується досить складно, за винятком декількох випадків, коли інформація про структуру області калібрувань знаходиться безпосередньо в цьому дампі. Наприклад, у випадку з системами фірми Delphi в дампі міститися адреси всіх карт з розмірами в одній таблиці – завдання знайти таку таблицю автоматично і отримати з неї дані. Електроний блок управління фірми Bosch також нескладно розбиваються на карти. Після того, як перший етап обробки дампа завершено, отримуємо набір готових для

ISSN 2663 - 4023

класифікації карт, які представляють собою таблицю значень. Вхідними даними для системи є матриці різних розмірностей *M*.

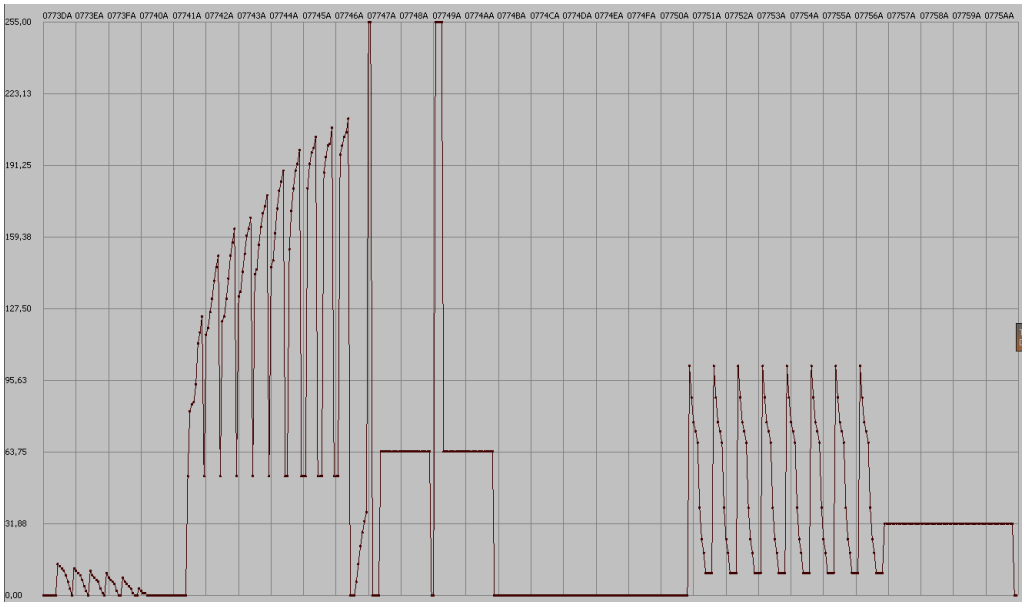


*Рис. 1. Графічне представлення дампа*

**Постановка задачі класифікації.** Аналіз властивостей вхідних матриць *M* потребує попереднього етапу для «розпізнавання» та надання істотних властивостей у вигляді вектора для кожного класу. Розглянемо постановку задачі класифікації. Класифікація – це задача віднесення вихідної матриці *m** до деякого класу властивостей. При цьому задані можливі класи *C*, доступна множина інших матриць *M*, а також відомі (чи не відомі) істотні ознаки *F*, що характеризують ці матриці: $F = \varnothing$ або $F \neq \varnothing$. Загальна постановка задачі класифікації властивостей визначається у вигляді:

- дана множина матриць *M*. Множина матриць може бути представлена підмножиною схожих матриць, які називаються класами *C*;
- існує або витягується: інформація про класи *C*, інформація про вихідну матрицю *m**, приналежність якої до певного класу невідома;
- потрібно за описом матриці *m** побудувати алгоритм ψ для визначення приналежності m* до деякого класу *c**.

Формальна постановка задачі класифікації з невідомими властивостями *F*, які потрібно попередньо витягти: дано $M \neq \varnothing$, $C \neq \varnothing$, $M=\{m_i\}$, $i = 1, 2,\ldots, M_k$; $m^* \notin M$, $C=\{c_j\}$, $j = 1, 2, \ldots, C_k$; відомо відношення $R \subset M \times C$ та деякий критерій схожості *dist*. Треба побудувати алгоритм ψ: $m^* \to c^*$, $c^* \in C$, такий що

$$\begin{gathered} c^* = \psi(m^*, M, R1, C, F, dist),\ \ \psi = <\psi 1, \psi 2>, \\ \psi 1: M \to F,\ \ \psi 1: m^* \to f^*,\ \text{де } F = \{f_n\},\ n = 1,2,\ldots,Nk;\ f^* \in F; \\ \psi 2: f^* \to c^*,\ \ c^* \in C,\ \ R1 \subset F \times C. \end{gathered} \tag{1}$$

Наприклад, алгоритм ψ може бути представлений у наступному вигляді:

ISSN 2663 - 4023

$$c^* = c_z, \quad z = \arg\min_i(g_i);$$
$$g_i = dist(f^*, f_i),$$
$$f_i = \psi 1(m_i),$$
$$f^* = \psi 1(m^*), \quad i = 1, 2, ..., Mk.$$

**Алгоритм класифікації.** Блок класифікації, складається з набору простих характеризаторів. Характеризатором будемо називати окремий елемент, який визначає певну властивість вхідної матриці за власним алгоритмом $\psi 1$ і порівнює її значення з еталонними даними (отриманими раніше в процесі навчання) за алгоритмом $\psi 2$.

Алгоритм класифікації представлений на рис. 2. При класифікації тест-блок отримує набір еталонних ознак для кожного з характеризаторів, що відповідає певному класу. Якщо хоч один з тестів дає негативну відповідь, відбувається перехід до наступного класу для перевірки. Якщо перевірка на відповідність не дає позитивний результат для всього набору класів, то клас вважається невизначеним.

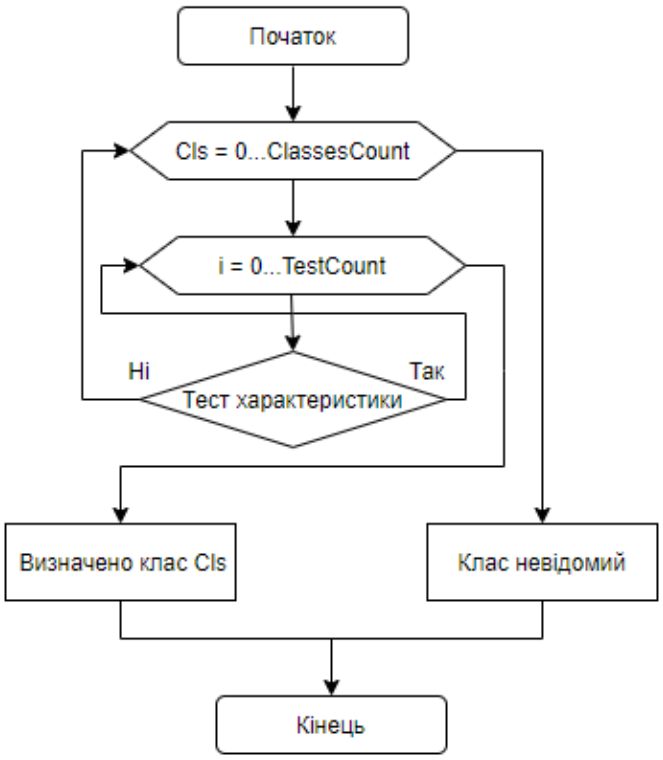


*Рис. 2. Блок-схема алгоритму класифікації*

Представлений на рис. 2 алгоритм, містить подвійний цикл, на верхньому рівні відбувається ітеративний прохід по відомим класам, які були навчені. На цьому рівні алгоритм працює зі словником карт, йому важливий тільки список класів. Якщо реалізувати алгоритм в такому вигляді кількість характеризаторів буде:

$$N_\psi = n \cdot m, \tag{2}$$

де $n$ – кількість характеристик;

$m$ – кількість відомих класів.

При такому підході реалізація системи буде вимагати великого обсягу пам'яті, навіть якщо характеризатори будуть досить простими, і містити мало навчальних даних.

Для скорочення кількості пам'яті, а також для прискорення роботи алгоритму, існує можливість створювати всього один набір характеризаторів, тому що він принципово однаковий для всіх класів карт. На етапі перевірки конкретного класу, кожному характеризатору передаються дані навчання для даного класу, після чого відбувається перевірка. Цей спосіб досить легко реалізується, виходячи з того, що дані зберігаються всі разом в одному словнику, і значно покращує роботу системи. Друга можлива оптимізація ґрунтується на алгоритмі перевірки. Адже відповідь для кожного класу це:

$$R = \wedge_{i=0}^{N} C_i(M), \tag{3}$$

тобто загальний результат $R$ для даного класу карт є кон'юнкція результатів кожного з характеризаторів $C_i(M)$. Якщо хоча б один з них дасть негативну відповідь, то можна переходити відразу до наступного класу карт (рис. 2). Однією з можливостей оптимізації є пріоритет самих характеризаторів. При цьому можливі два варіанти:

1) В результаті ретельного тестування визначити найбільш підходящу послідовність характеризаторів.

2) Реалізувати для кожного класу пріоритет характеризаторів, тобто для кожного класу послідовність буде різною.

Для оптимізації процесу тестування був використаний перший варіант. Однак можна припустити, що з ростом кількості класів необхідність реалізації другого варіанту зросте. Для того, щоб передбачити перехід до другого варіанту, кожному характеризатору було визначено додаткову властивість – пріоритет.

**Види характеризаторів**. Характеризатор – це самостійний блок, що обчислює характеристику карти, за яку він відповідає за допомогою власного алгоритму $\psi_i$. У загальному вигляді характеризатор являє собою програмний модуль, який:

- обчислює визначену ним характеристику карти;
- надає алгоритм зберігання навчених даних;
- вміє завантажити збережені навчені дані;
- відповідає на питання, чи підходить карта під навчені дані, встановлені їй заздалегідь.

Крім цього, кожен характерізатор має пріоритет – натуральне число, що привласнюється йому ззовні при ініціалізації тест блоку.

Ознакою для кожного характеризатора є набір дійсних чисел, які обчислюються за заданим алгоритмом $\psi_i$ всередині цього характеризатора. Таким чином, кожен характеризатор має свій власний формат зберігання еталонних ознак. У загальному випадку ознака може бути представлена вектором $P = \{p_1, p_2, \ldots, p_n\}$. Набори еталонних ознак, які навчені для кожного класу, є словником. Зберігання дійсних чисел досить просто реалізується, тому питання серіалізації даних навчання можна вважати принципово вирішеним.

Характеризатор визначається наступним набором властивостей:

$$\Psi = \left\langle P, W, f_{\psi}(m^*), f_t(m^*) \right\rangle, \tag{4}$$

ISSN 2663 - 4023

де $P$ – формат ознаки; W – алгоритм навчання та зберігання даних навчання; $f_\psi(m^*)$ – функція обчислення ознак за вхідною матрицею $m^*$; $f_t(m^*)$ – функція тестування матриці $m^*$ на основі навчених даних.

Наведемо стислий опис деяких, розроблених для пропонованої системи, видів характеризаторів.

*Характеризатор розмірів* визначає розміри вхідної таблиці та перевіряє допустимість розмірів в даному класі. Формат ознаки: $P = \{K, L\}$, де $K$ – кількість рядків, $L$ – кількість стовпців. Алгоритм навчання засновано на визначення найменшого та найбільшого з розмірів. Так для набору матриць $M = \{m_i | \ i = 1,2,\ldots,n\}$ вектори ознак розраховуються наступним чином:

$$\begin{aligned} P_{\min} &= \arg\min_i(size(m_i)); \\ P_{\max} &= \arg\max_i(size(m_i)); \end{aligned} \tag{5}$$

Алгоритм тестування виконує перевірку наявності розміру в списку та визначає чи входить розмір в область допустимих значень: $(size(m^*) \geq P_{\min})$ AND $(size(m^*) \leq P_{\max})$.

*Характеризатор області значень* визначає мінімальне і максимальне допустиме значення в калібрувальній таблиці. Формат ознаки: $P = \{\min, \max\}$. Навчання засновано на пошуку мінімального $P_{\min}$ і максимального $P_{\max}$ значення з усіх матриць навчальної вибірки. Тест перевіряє умову потрапляння знайденого мінімального та максимального значення в діапазон, збережений на етапі навчання: $(min(m^*) \geq P_{\min})$ AND $(max(m^*) \leq P_{\max})$.

*Характеризатор кінцевих різниць* [11] обраного фрагмента. При ініціалізації задається функція вибору елементів з матриці, які визначають обраний фрагмент $D(m) = \{x_1, x_2,\ldots,x_n\}$. Обчислюються кінцеві різниці обраного фрагмента $\Delta x_i = x_{i+1} - x_i$ і порівнюються з навченими, виходячи з допустимого вектора нев'язки.

**Алгоритм визначення міри схожості послідовностей.** Для визначення міри схожості векторів ознак використовувався алгоритм, заснований на розрахунку коефіцієнтів апроксимації. Вхідні дані визначаються як набір послідовностей: $X_i = \{x_0, x_1, x_2,\ldots, x_n\}$ і наводяться до середнього масштабу за наступним алгоритмом масштабування. Визначається середній діапазон значень для усіх перевірених послідовностей:

$$\begin{aligned} P_{\min} &= \frac{1}{n}\sum_{i=0}^{n}\min(X_i); \\ P_{\max} &= \frac{1}{n}\sum_{i=0}^{n}\max(X_i). \end{aligned} \tag{5}$$

Розраховується коефіцієнт пропорційності:

$$k = \frac{P_{\max}^i - P_{\min}^i}{P_{\max} - P_{\min}}. \tag{6}$$

Розраховується значення кожного елемента послідовності:

$$x'_{ij} = k(x_{ij} - P^i_{\min}) + P_{\min}, \quad (7)$$

де $P^i_{\min}$ – мінімальне значення і-ой послідовності, $P_{\min}$ – мінімальне значення середнього діапазону.

Для масштабованих послідовностей виконується апроксимація [12] кожної послідовності на основі багаточлену: $y = ax^2 + bx + c$. Для апроксимації в системі був використаний МНК. Коефіцієнти $a$, $b$, $c$ – основа для дослідження схожості. У даному випадку формат ознаки: $P = \{a, b, c\}$ або $\Delta P = \{\Delta a, \Delta b, \Delta c\}$, де $P$ – середнє значення вектора апроксимуючих коефіцієнтів, а $\Delta P$–максимальне відхилення $a$, $b$, $c$ від середнього (вектора $P$). Для простоти подання вектори можуть бути зведені в один 6-тімерний вектор, що містить 3 коефіцієнта і 3 їх максимальних відхилення.

Використання апроксимації для дослідження схожості показало хороший результат, враховуючи досить високу швидкість його роботи.

Для роботи деяких характеризаторів, необхідно зберігати дані навчання більш компактно. Для цього при навчанні на вибірці з різними розмірами ознак, вибираються ключові розміри, а інші приводяться до таких ключових розмірів. Для приведення послідовності до ключового (або вузлового) розміру, необхідно не просто провести інтерполяцію, а масштабування з урахуванням того, що вузлові точки не збігаються. Для цього пропонується алгоритм інтерполяції з плаваючими вузловими точками. На рис. 3 представлений приклад інтерполяції з плаваючими вузловими точками, де сині точки – це вузли вихідної функції, яку потрібно привести до функції, яка показана червоними точками.

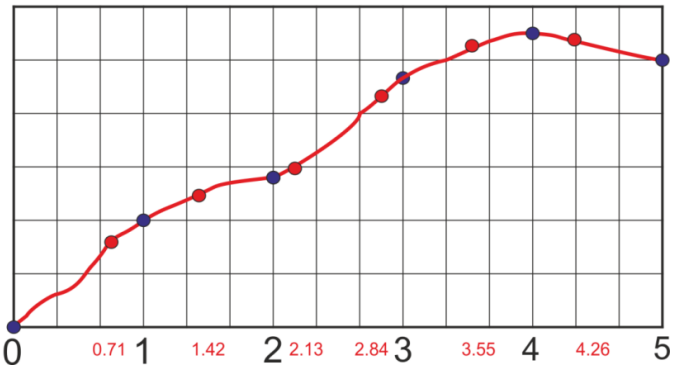


*Рис. 3. Інтерполяція з плаваючими вузловими точками*

Тільки абсциси першого і останнього вузлів співпадають. Отже для такого роду інтерполяції необхідно знати тільки нові значення осі абсцис. Алгоритм складається з наступних етапів:

1) Обчислюється масштаб, який дорівнює відношенню кількості проміжків вихідної функції до кількості проміжків в цільової функції $k = (n\text{-}1)/(m\text{-}1)$, де $n$ – кількість точок у вихідній функції, $m$ – необхідна кількість точок.

2) Визначається значення абсциси виходячи з масштабу, множенням номеру вузла на знайдений масштаб $X_i = i \times k$.

3) Для кожного нового значення абсциси обчислюється значення функції за допомогою інтерполяції між сусідніми точками вихідної функції. Для цього може бути використовано будь-який з існуючих алгоритмів інтерполяції (у даному випадку використовувався алгоритм лінійної інтерполяції).

ISSN 2663 - 4023

## РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Система містить два компонента: тест блок, в якому знаходяться всі необхідні характеризатори та менеджер словників, який керує тестовим блоком. Система побудована з компонентів за принципом єдиної відповідальності, коли кожен компонент відповідає тільки за одну дію. В даному випадку завдання тест блоку – відповісти чи відповідає образ поточному класу, який перевіряється, на основі наданих тест-блоку навчених еталонних ознак. Роль іншого компонента – управління словником навчених даних і виконання завдання класифікації.

Програмна система класифікації розроблена з використанням платформи .NET Framework. Мова програмування C#. Для тестування працездатності системи розроблено консольний застосунок, а також зроблена невелика вибірка карт – 3-5 для кожного класу, і вибрано всього 3 різних класи. Карти були розсортовані по папках, в одній папці – окремий клас карт, у назві файлу відзначені розміри карт. Спочатку створюється тестовий блок, в якому відзначені наступні характеризатори:

– аналізатор розмірів;
– аналізатор спектру;
– аналізатор кінцевих різниць першого рядка;
– аналізатор кінцевих різниць останнього рядка;
– аналізатор схожості послідовностей.

Далі, для кожен класу в масив завантажується матриця і додається в словник, попередньо отримавши навчені дані. Після ініціалізації даних завантажується одна випадкова карта з наданою вибірки і перевіряється здатність системи визначити її клас. Для налагодження системи виводяться значення кінцевих різниць і вектора нев'язок (рис. 4). Результати попереднього тестування представлені в табл.4.1.

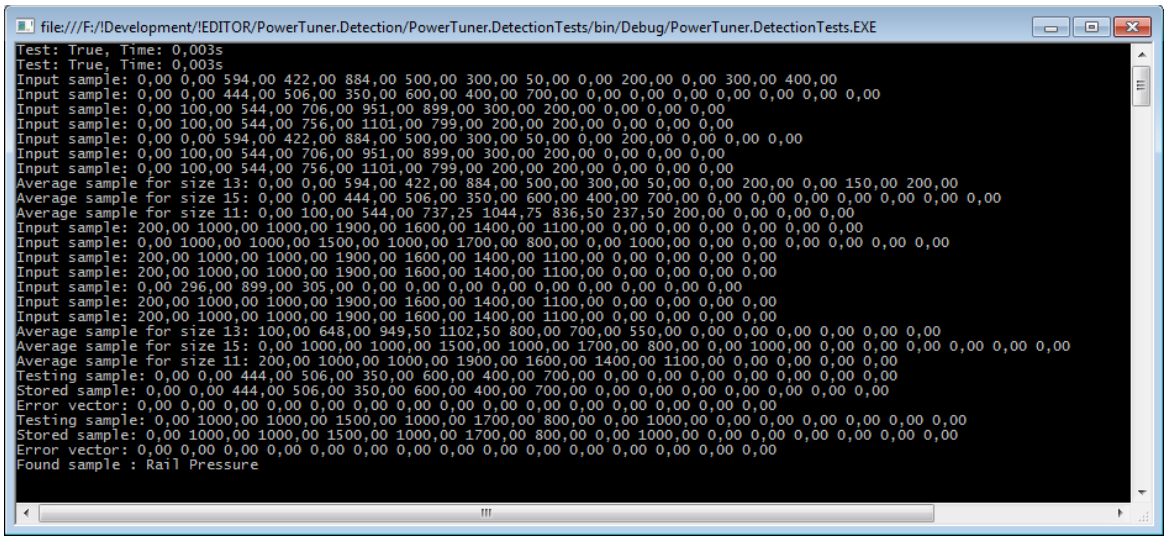


*Рис. 4. Результат попереднього тестування*

*Таблиця 1*

**Швидкість навчання на попередньому тестуванні**

| № | 1 вибірка | 1000 вибірок | 1000000 вибірок |
|---|---|---|---|
| 1 | 3 мс | 28 мс | 49 мс |
| 2 | 4 мс | 32 мс | 55 мс |

Аналіз результатів попереднього тестування показав, що система побудована задовільно. Наступний етап - це побудова системи тестування і аналіз ефективності системи. Для даного завдання було прийнято рішення обмежитися модульним тестуванням. Цей програмний продукт є окремим модулем, який повинен бути вбудований в інше програмне забезпечення для поліпшення його роботи. Тому проводилося тестування цієї системи на предмет відмовостійкості, ефективності і продуктивності. Тестування відбувалося в режимі «білого ящика», оскільки при впровадженні системи в програмне забезпечення для редагування необхідно буде дописувати адаптаційні програмні модулі.

Для тестування вибрано сімейство блоків автомобілів Mercedes-Benz Bosch EDC16C31/EDC16CP31. Для спрощення набору еталонних карт було прийнято рішення створити модуль, який автоматично буде зберігати карти в окремі файли і сортувати їх за такими ознаками: марка автомобіля; фірма-виробник блоку керування; сімейство блоку. Помічник для роботи з картами автоматично зберігає обрану карту в окремий бінарний файл. Для простоти завантаження даних було створено дерево каталогів. Для навчання окремого словника вибирається відповідна йому папка з навчальними еталонними матрицями, імена яких несуть в собі інформацію про те, як саме завантажувати карту, і до якого класу образів вона відноситься. Для швидкого завантаження карт був розроблений алгоритм на основі регулярних виразів, який вже на етапі перегляду файлу надає відразу всю інформацію, що зберігається в назві файлу. Такий метод, дозволяє також відкинути файли, які потрапили в папку випадково і мають не той формат імені файлу, який передбачено для еталонних матриць. Дані навчання зберігаються в словнику, де встановлюється відповідність між описом класу карти (в даному випадку це вектор) і набором еталонних матриць для навчання цього класу образів.

На етапі навчання була створена навчальна вибірка, що складається з понад 100 різних карт, які були збережені з 6 різних файлів калібровок одного і того ж сімейства блоків керування двигуном. У цій навчальній вибірці представлено 10 основних класів карт, тобто приблизно по 10 навчальних еталонів на кожен клас. Після чого на кожен клас було вибрано по кілька карт для тестування точності.

У табл. 2 наведені результати тестування. На рис. 5 представлені графіки результатів тестування.

*Таблиця 2*

**Результати тестування точності розпізнавання системи**

| № | Клас карти | Кількість еталонів | Кількість тестів | Час, мс | Результат тестування |
|---|---|---|---|---|---|
| 1 | Advance | 19 | 5 | 55 | 100% |
| 2 | Boost Limiter via PAtm. | 8 | 2 | 14 | 50% |
| 3 | Boost Limiter via Temp. | 6 | 1 | 8 | 100% |
| 4 | Boost Pressure | 16 | 6 | 82 | 100% |
| 5 | Driver’s Wish | 10 | 4 | 228 | 60% |
| 6 | Geometry Control | 14 | 5 | 293 | 60% |
| 7 | Rail Pressure | 20 | 5 | 310 | 100% |
| 8 | Smoke limiter via Boost | 8 | 3 | 195 | 60% |
| 9 | Smoke Limiter | 12 | 2 | 133 | 100% |
| | Разом | 118 | 33 | 1318 | 74% |

ISSN 2663 - 4023

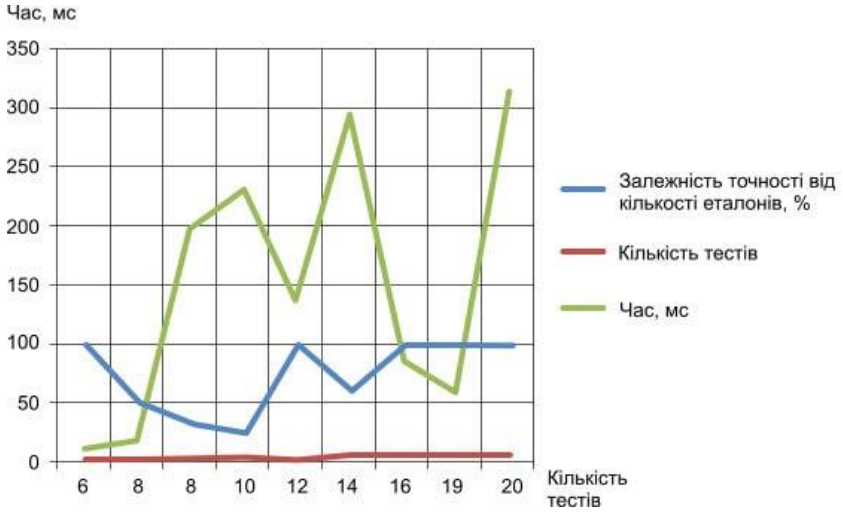


*Рис. 5. Графічне відображення результатів тестування*

## ВИСНОВКИ ТА ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

По результатам тестування програмної системи класифікації параметрів електронного блоку керування двигуном можна відмітити наступне: у випадках, коли кількість навчальних образів велика (12 та більше) – результат задовольняє вимогам, для малої кількість образів – класи починають перетинатися.

В середньому на 5 тестів витрачається 150 мс. Тести були проведені в режимі налагодження (Debug), тому швидкісні показники завищені. При запуску програми тестування з EXE файлу в режим Release показники часу будуть нижчими.

Точність класифікації після налагодження системи склала 75 - 82 %.

Вектор подальших досліджень може бути спрямований на розв’язання проблеми підвищення швидкості та точності класифікації за рахунок реалізації пріоритетів характеризаторів, а також введення додаткових ознак і видів характеризаторів для усунення ситуацій перетину класів у випадку класифікації за невеликою кількістю навчальних образів.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 European Parliament. (2009). Directive 2009/33/EC of the European Parliament and of the Council of 23 April 2009 on the promotion of clean and energy-efficient road transport vehicles.

2 *Swiftec - Forget The Limits! :: Products :: Swiftec :: Overview*. (б. д.). Swiftec - Forget The Limits! :: Products. https://www.vcpowerteam.com/products/swiftec/

3 *Software WinOLS*. (б. д.). EVC electronic - The Tools For Chiptuning. https://www.evc.de/en/product/ols/software/

4 de Nola, F., Giardiello, G., Gimelli, A., Molteni, A., Muccillo, M., & Picariello, R. (2019). Volumetric efficiency estimation based on neural networks to reduce the experimental effort in engine base calibration. *Fuel*, *244*, 31–39. https://doi.org/10.1016/j.fuel.2019.01.182

*5* Макаров А. (2011). Алгоритмы контроля и диагностики систем управления авиационными ГТД на основе нейросетевых моделей и нечеткой логики (кандидат технических наук). *Уфимский государственный авиационный технический университет.*

6 Wu, J.-D., & Liu, C.-H. (2009). An expert system for fault diagnosis in internal combustion engines using wavelet packet transform and neural network. *Expert Systems with Applications*, *36*(3), 4278–4286. https://doi.org/10.1016/j.eswa.2008.03.008

7 Luján, J. M., Climent, H., García-Cuevas, L. M., & Moratal, A. (2017). Volumetric efficiency modelling of internal combustion engines based on a novel adaptive learning algorithm of artificial neural networks. *Applied Thermal Engineering*, *123*, 625–634. https://doi.org/10.1016/j.applthermaleng.2017.05.087

8 Shatnawi, Y., & Al-khassaweneh, M. (2013). Fault Diagnosis in Internal Combustion Engines Using Extension Neural Network. *IEEE Transactions on Industrial Electronics*, (*61)*, 1434–1443.

9 Jian-Da, W., Peng-Hsin, Ch., Yo-Wei, Ch., Yao-jung, Sh. (2008). An expert system for fault diagnosis in internal combustion engines using probability neural network. *Expert Systems with Applications*, (34), 2704-2713.

10 Jian-Da, W., Chen, J.-Ch. (2006). Continuous wavelet transform technique for fault signal diagnosis of internal combustion engines. *NDT & E International*, (39), 304-311.

11 Соловейчик, Ю.Г., Рояк, М.Э., & Персова, М.Г. (2007). *Метод конечных элементов для скалярных и векторных задач*. НГТУ.

12 Лоран, П.Ж. (1975). *Аппроксимация и оптимизация*. Мир.

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 3 (11), 2021
ISSN 2663 - 4023

**Svitlana D. Kuznichenko**
PhD in Geographical Science, Associate Professor of Department of Information Technology
Odessa State Environmental University, Odessa, Ukraine
ORCID ID: 0000-0001-7982-1298
*skuznichenko@gmail.com*

**Tetiana M. Tereshchenko**
PhD in Technical Science, Associate Professor of Department of Information Technology
Odessa State Environmental University, Odessa, Ukraine
ORCID ID: 0000-0001-7691-6996
*tereshchenko.odessa@gmail.com*

**Iryna V. Buchynska**
PhD, assistant of Department of Information Technology
Odessa State Environmental University, Odessa, Ukraine
ORCID ID: 0000-0002-0393-2781
*buchinskayaira@gmail.com*

**Viktoriia V. Klepatska**
assistant of Department of Information Technology
Odessa State Environmental University, Odessa, Ukraine
ORCID ID: 0000-0001-5613-6546
*victoria.klepatska@gmail.com*

# PARAMETER CLASSIFICATION SOFTWARE BASED ON CHARACTERIZERS AND KNOWLEDGE BASE FOR ELECTRONIC ENGINE CONTROL UNIT

**Abstract.** The article discusses the issues of increasing the efficiency of the classification process of cards of electronic control units of a car engine. The analysis of the existing software for editing calibration tables in electronic engine control unit, which has tools for determining calibrations and data recognition, was carried out. The limits of use of such software products are conditioned by a small number of specified classes of calibration tables and low data processing speed. The analysis of testing results of classification methods using spectral decomposition demonstrated that a system based on this method requires complex transformations of the results of spectral decomposition. The use of spectral decomposition as a solution of the classification problem is possible if some characteristics of the input data are determined and used as data for classification. It was developed a data classification algorithm that uses characterizers to compute a clearly identified characteristic of the input matrix. The software package for the implementation of the developed algorithm was carried out by using the .NET Framework and the C # programming language. The testing of the classification system performance performed by using the developed software system on a small sample of maps. The results of preliminary testing showed that the system determines correctly the class of the provided card after training. Further testing on the Mercedes-Benz Bosch EDC16C31 / EDC16CP31 car block family showed that in cases of a large number of training images, the result meets the requirements. The performed tests allowed us to determine the optimal number of images for training and the time required for this.

**Keywords:** classification with learning; pattern recognition; knowledge base; software system; electronic engine control unit.

## REFERENCES (TRANSLATED AND TRANSLITERATED)

1 European Parliament. (2009). Directive 2009/33/EC of the European Parliament and of the Council of 23 April 2009 on the promotion of clean and energy-efficient road transport vehicles.

2 *Swiftec - Forget The Limits! :: Products :: Swiftec :: Overview*. (б. д.). Swiftec - Forget The Limits! :: Products. https://www.vcpowerteam.com/products/swiftec/

3 *Software WinOLS*. (б. д.). EVC electronic - The Tools For Chiptuning. https://www.evc.de/en/product/ols/software/

4 de Nola, F., Giardiello, G., Gimelli, A., Molteni, A., Muccillo, M., & Picariello, R. (2019). Volumetric efficiency estimation based on neural networks to reduce the experimental effort in engine base calibration. *Fuel*, *244*, 31–39. https://doi.org/10.1016/j.fuel.2019.01.182

5 Makarov, A. (2011). *Algoritmyi kontrolya i diagnostiki sistem upravleniya aviatsionnyimi GTD na osnove neyrosetevyih modeley i nechetkoy logiki (kandidat tehnicheskih nauk).* Ufimskiy gosudarstvennyiy aviatsionnyiy tehnicheskiy universitet.

6 Wu, J.-D., & Liu, C.-H. (2009). An expert system for fault diagnosis in internal combustion engines using wavelet packet transform and neural network. *Expert Systems with Applications*, *36*(3), 4278–4286. https://doi.org/10.1016/j.eswa.2008.03.008

7 Luján, J. M., Climent, H., García-Cuevas, L. M., & Moratal, A. (2017). Volumetric efficiency modelling of internal combustion engines based on a novel adaptive learning algorithm of artificial neural networks. *Applied Thermal Engineering*, *123*, 625–634. https://doi.org/10.1016/j.applthermaleng.2017.05.087

8 Shatnawi, Y., & Al-khassaweneh, M. (2013). Fault Diagnosis in Internal Combustion Engines Using Extension Neural Network. *IEEE Transactions on Industrial Electronics*, (*61)*, 1434–1443.

9 Jian-Da, W., Peng-Hsin, Ch., Yo-Wei, Ch., Yao-jung, Sh. (2008). An expert system for fault diagnosis in internal combustion engines using probability neural network. *Expert Systems with Applications*, (34), 2704-2713.

10 Jian-Da, W., Chen, J.-Ch. (2006). Continuous wavelet transform technique for fault signal diagnosis of internal combustion engines. *NDT & E International*, (39), 304-311.

11 Soloveychik, Yu.G., Royak, M.E., & Persova, M.G. (2007). *Metod konechnyih elementov dlya skalyarnyih i vektornyih zadach.* NGTU.

12 Loran, P.Zh. (1975). *Approksimatsiya i optimizatsiya.* Mir.