Харківський національний університет імені В.Н. Каразіна

Факультет математики і інформатики

Кафедра прикладної математики

Кваліфікаційна робота

на тему **«Застосування штучної нейронної мережі до розв'язання одної задачі з теорії керування»**

Виконала: студентка групи МП41 IV курсу,
спеціальності 113
Прикладна математика
Брусенцева О.Є.

Керівник: доктор фіз.-мат. наук
професор кафедри
прикладної математики
Ігнатович С.Ю.

Рецензент: викладач кафедри
прикладної математики
Карєва В.В.

Харків — 2023 рік

Анотації

Брусенцева О.Є. Застосування штучної нейронної мережі для розв'язку однієї задачі теорії керування.

У даній роботі запропоновано нейрону мережу для розв'язку лінійної задачі швидкодії. Аналітичний розв'язок, запропонований В.І.Коробовим, Г.М.Скляром, узято за основу логіки побудови нейронної мережі. Архітектура алгоритму реалізована за допомогою мови програмування Python. Використано бібліотеки numpy, tensorFlow та Pytorch. Нейронна мережа є прикладом алгоритму, що навчається із "вчителем". Для навчання були підготовлені два набори даних: тренувальний та валідаційний. Тренувальний набір містить два датасети. Перший датасет складається з початкових точок x^0, тоді як другий містить моменти перемикань та тип керування. Валідаційний набір даних має аналогічну до тренувального структуру.

Brusentseva O.Ye. Using an artificial neural network to solve one problem from the control theory.

In this study, a neural network is proposed for solving a linear problem of time-optimal control. The analytical solution described by V.I. Korobov, G.M. Sklyar forms the basis for the logic of constructing the neural network. The algorithm architecture is implemented using the Python programming language, utilizing the libraries numpy, TensorFlow, and Pytorch. The neural network serves as an example of a supervised learning algorithm. Two data sets were prepared for training: the training set and the validation set. The training set consists of two datasets. The first data set comprises initial points, while the second data set contains switching moments, and the control type. The validation data set has a structure similar to the training data set.

Зміст

Вступ

Останні роки популярності набувають моделі штучного інтелекту для розв'язку завдань різноманітного характеру. У першу чергу ефект обумовлений тим, що такий спосіб може забезпечити низку вдосконалень традиційних методів, зокрема автоматизацію розв'язання повторюваних задач, підвищення ефективності та точності рішень, персоналізацію та вдосконалення досвіду користувачів, зниження витрат та багато іншого. Однією з переваг алгоритмів штучного інтелекту є обчислювальна простота та гнучкість. Наприклад, нейронній мережі достатньо один раз навчитися розв'язувати клас задач, щоб потім давати миттєві рішення вже з іншими початковими умовами.

Нейронна мережа є альтернативним способом розв'язку задач, у яких аналітичний алгоритм пошуку рішення є обчислювально занадто складним. Необхідно пам'ятати, що навіть гарно натренована мережа надає *апроксимований* результат, хоча й дуже близький до справжнього. Такий спосіб пошуку рішень може використовуватися для моделювання складних фізичних процесів, де інші методи є надто складними для застосування. У цьому контексті, використання нейронної мережі для задачі швидкодії може стати альтернативою традиційним методам розв'язку.

У даній роботі ми хочемо представити нейрону мережу, як спосіб розв'язку задачі швидкодії наступного вигляду:

$$\dot{x_1} = u, \quad \dot{x_k} = x_{k-1}, \quad k = 2, ..., n,$$

$$|u(t)| \leq 1, \quad x(0) = x^0, \quad x(\theta) = 0, \quad \theta \to min.$$

Пошук альтернативних способів розв'язку для задачі оптимального за часом керування є важливим, адже дана проблема є яскравим прикладом такої, де розв'язок значно поскладнюється зі збільшенням розмірності.

Розв'язок задачі для $n = 2$ природнім чином випливає із принципу максимуму Понтрягіна, проте для $n \geq 3$ зростає кількість необхідних обмежень. До 1987 року задача оптимального керування для $n \geq 4$ вважалася нерозв'язною в аналітичниму сенсі та основний акцент у дослідженнях робився на чисельних методах знаходження рішення.

Однак у 1988 році [1] В.І.Коробов та Г.М.Скляр описали аналітичний розв'язок $n-$мірної лінійної задачі швидкодії. Потрібно зауважити, що спосіб розв'язку тісно пов'язаний із проблемою моментів Маркова. Вперше запропонував розглядати задачу теорії керування з точки зору проблеми моментів Н.Н.Красовський у 1968 році [2].

Не зважаючи на те, що аналітичний розв'язок вже існує, обчислювальна складність алгоритму не дає можливості швидко знаходити моменти перемикання та тип керування для великих розмірностей. В першу чергу це пов'язано із степенем полінома, що утворюється в результаті підрахунку визначника $\Delta_n^{\pm}(x^0, \theta)$. Для непарних n степінь полінома є $\frac{(n+1)^2}{4}$, для парних $\frac{n(n+2)}{4}$. Це означає, що для розмірності $n = 4$ степінь полінома $\Delta_n^{\pm}(x^0, \theta)$ дорівнює 6, для $n = 10$ – 30, а при $n = 20$ вже 110.

Задача пошуку коренів полінома є обчислювально доволі складною та займає чимало часу, тож нейронна мережа є альтернативою в плані швидкості пошуку рішення. Застосування алгоритму мережі може бути корисним в багатьох галузях таких як фізика, інженерія, економіка тощо.

Для прикладу нейронні мережі мають потенціал для ефективного моделювання складних систем керування з великою кількістю змінних та обме-

жень. Це відкриває можливості для покращення швидкості, точності та енергоефективності процесів у різних інженерних областях, таких як автоматизоване керування, робототехніка, авіаційна та космічна техніка, енергетика, транспортні системи та багато інших.

Застосування нейронних мереж для розв'язання задач оптимального керування в інженерній діяльності може привести до нових можливостей в проектуванні та управлінні складними системами, покращення продуктивності та зниження витрат ресурсів.

Розділ 1. Огляд літератури

1.1. Постановка задачі

Розглянемо наступну лінійну задачу оптимального керування в сенсі швидкодії:

$$\begin{aligned} \dot{x_1} &= u, \\ \dot{x_k} &= x_{k-1}, \quad k = 2, ..., n, \\ |u(t)| &\leq 1, \quad x(0) = x^0, \quad x(\theta) = 0, \quad \theta \to min. \end{aligned} \tag{1.1}$$

За допомогою тренувального та валідаційного наборів даних, ми ставимо собі на меті навчити нейронну мережу визначати точки перемикання та тип керування систем розмірності $n = 4$ та $n = 10$.

Такі розмірності обрані з наступних міркувань: на прикладі $n = 4$ ми продемонструємо точність алгоритму отриманої нейронної мережі, провівши експеримент із $n = 10$ порівняємо швидкість знаходження результату із класичним методом розв'язу даної задачі.

1.2. Задача швидкодії

Практично кожен процес, що відбувається у нашому житті, можна розглядати як керований. Проблеми, пов'язані з теорією керування, виникають у різних наукових галузях, таких як математика, фізика, інженерія,

економіка, біологія тощо. Природна потреба полягає у досягненні найкращих результатів з точки зору швидкості, енергоефективності, економії ресурсів та інших критеріїв. Математичні моделі для таких проблем зазвичай являють собою задачі варіаційного числення. Однак, у задачі вигляду (1.1) присутнє обмеження вигляду $|u(t)| \leq 1$. Математичного апарату для розв'язку таких *некласичних задач* раніше не було. Тому протягом певного часу ряд задач вважався нерозв'язним.

Однак у роботі "Математична теорія оптимальних процесів" [7] Л.С.Понтрягін вперше ввів поняття *принципу максимуму*, який став основою для розв'язання більшості специфічних задач того часу. Принцип дає необхідні умови для досягнення оптимальності у керованій системі.

Наведемо теорему про принцип максимума. Розглянемо наступну систему:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, u), \tag{1.2}$$

де $\mathbf{f}(\mathbf{x}, u)$ є вектор-функція з наступними координатами

$$f^1(x,u), f^2(x,u), ..., f^n(x,u), \quad x \in X, u \in U.$$

Функції є неперервними за сукупністю параметрів $(x^1, x^2, x^3, ..., x^n, u)$ та неперервно диференційовними за $(x^1, x^2, x^3, ..., x^n)$.

Нехай задано ще одну функцію $f^0(x^1, x^2, x^3, ..., x^n, u) = f^0(x, u)$, що визначена та неперервна з усіма частковими похідними $\dfrac{\partial f^0}{\partial x^i}$, $i = 1, 2, ..., n$, на просторі $X \times U$. Тоді можна сформулювати задачу оптимального керування наступним чином. *Серед усіх допустимих керувань $u = u(t)$ у фазовому просторі, що переводять фазову точку з положення x_0 в положення x_1, знайти таке, для якого функціонал $J = \int_{t_0}^{t_1} f^0(x(t), u(t))dt$ приймає найменше значення. Тут $x(t)$ розв'язок рівняння з початковою умовою $x(t_0) = x_0$, що відповідає керуванню $u(t)$, t_1 це момент проходження цього рішення через точку x_1.* Якщо розглянути частковий випадок,

коли $f^0(x,u) = 1$, тоді функціонал приймає вид $J = t_1 - t_0$. Така задача називається *задачею про оптимальну швидкодію.*

Випишемо наступні системи:

$$\frac{dx^i}{dt} = f^i(x,u), \quad i = 0,1,2,...,n, \tag{1.3}$$

$$\frac{d\psi_i}{dt} = -\sum_{\alpha=0}^{n} \frac{\partial f^\alpha(x(t),u(t))}{\partial x^i}\psi_\alpha, \quad i = 0,1,...,n, \tag{1.4}$$

для допустимого керування $u(t), t_0 \le t \le t_1$ та відповідної фазової траєкторії $x(t)$ з початковою умовою $x(t_0) = x_0$. Остання система є однорідною, тому для будь-яких початкових умовах $\psi_i(t_0)$ вона має єдиний розв'язок.

Запишемо наступні функції:

$$H = (\psi, x, u) = (\psi, f(x,u)) = \sum_{\alpha=0}^{n} \psi_\alpha f^\alpha(x,u), \tag{1.5}$$

$$M(\psi, x) = sup_{u\in U} H(\psi, x, u) \tag{1.6}$$

Теорема 1.1 (Принцип максимуму [7]). *Нехай $u(t), t_0 \le t \le t_1$ допустиме керування, $x(t)$ відповідна траєкторія, що виходить з точки x_0 в момент t_0. Щоб $u(t)$ та $x(t)$ були оптимальними, необхідне існування такої ненульової неперерервної вектор-функції $\psi(t) = (\psi_0(t), \psi_1(t), .., \psi_n(t))$, що задовольняє систему:*

$$\frac{dx^i}{dt} = \frac{\partial H}{\partial \psi_i}, \quad i = 0,1,...,n, \tag{1.7}$$

$$\frac{d\psi_i}{dt} = -\frac{\partial H}{\partial x^i}, \quad i = 0,1,...,n, \tag{1.8}$$

причому

- *майже за будь-якого $t, t_0 \le t \le t_1$, функція $H(\psi(t), x(t), u)$ змінної $u \in U$ досягає в точці $u = u(t)$ максимума*

$$H(\psi(t), x(t), u(t)) = M(\psi(t), x(t)); \tag{1.9}$$

- *у кінцевий момент часу t_1 виконуються співвідношення*

$$\psi_0(t_1) \leq 0, \quad M(\psi(t_1), x(t_1)) = 0. \tag{1.10}$$

Таким чином, основна концепція базується на ідеї максимізації функції Гамільтона-Понтрягіна, яка визначається для керованої системи. Ця функція включає у себе як динамічну модель системи, так і критерій якості, що характеризує бажаний результат. Принцип максимуму стверджує, що для оптимальної траєкторії системи повинна виконуватись умова, за якої функція Гамільтона-Понтрягіна досягає максимуму в кожному моменті часу.

Використання принципу Понтрягіна відкрило нові можливості для вирішення задач оптимального керування в різних галузях науки і технологіях, оскільки багато проблем зводяться до задач оптимального керування. Він застосовується у робототехніці, фінансах, автоматизації виробництва, системах управління транспортом тощо.

Розуміння та використання принципу максимуму Понтрягіна дозволяє вирішувати складні задачі оптимального керування та досягати ефективних результатів у різних сферах діяльності.

Тоді для часткового випадку задачі (1.1) отримуємо наступні результати [7]:

(i) для будь-якої початкової умови $x^0 \in \mathbb{R}$ оптимальне за часом керування існує та єдине;

(ii) керування є кусково-постійною функцією та приймає значення ± 1, більш того система має не більше, ніж $n-1$ перемикань, де n це розмірність системи.

(iii) якщо $u = u(t), t \in [0, \theta]$ є кусково-постійною функцією, що приймає значення ± 1, має не більше, ніж $n-1$ переключень та переводить x^0

в початок координат за час θ, тоді ця функція є оптимальним керуванням, а θ є оптимальним часом.

Тож початкова задача теорії керування є скінченновимірною та полягає в тому, щоб знайти таке $p \leq n-1$ та $0 < t_1 < ... < t_p < \theta$, що керування $u(t) = \pm 1$ переводить початкову точку x^0 в 0 за час θ при моментах переключення t_i. [7]

1.3. Проблема моментів

Дана задача є прикладом такої, що значно поскладнюється зі збільшенням розмірності. Довгий час проблема для $n \geq 4$ вважалася нерозв'язною, однак у 1987 аналітичний розв'язок було знайдено [1]. Вагомим внеском у розвиток задачі швидкодії стали ідеї класичної теорії проблеми моментів А.А.Маркова.

Підхід до розв'язку задач теорії керування з точки зору проблеми моментів вперше було запропоновано Н.Н.Красовським [2]. Значний вклад в пошук аналітичного розв'язку задачі швидкодії зробили В.І. Коробов та Г.М.Скляр, коли розглянули задачу теорії керування як проблему моментів Маркова на мінімально можливому інтервалі.

Розглянемо лінійну керовану систему наступного виду [5]:

$$\dot{x} = A(t)x + b(t)u, \quad x \in \mathbb{R}^n, \quad u \in \mathbb{R}, \tag{1.11}$$

де $A(t)$ це матриця nxn з неперервними елементами, $b(t)$ $n-$мірний вектор, $u = u(t)$ є керування, що переводить початкову точку $x(0) = x^0$ в початок координат $x(T) = 0$ за час T. Нехай $\phi(t)$ фундаментальна матриця розв'язків системи $\dot{x} = A(t)x$, більш того така, що $\phi(0) = I$. Згідно формули Коші маємо наступне рівняння:

$$x(T) = 0 = \phi(T)x^0 + \phi(T)\int_0^T \phi^{-1}(t)b(t)u(t)dt. \tag{1.12}$$

Тоді ми можемо сказати, що виконується така моментна рівність:

$$x_k^0 = \int_0^T g_k(t)u(t)dt, \quad k = 1, ..., n \tag{1.13}$$

для $u(t)$, де $g(t) = (g_1(t), ..., g_n(t)) = -\phi^{-1}(t)b(t)$.

За обмеження $|u(t)| \leq 1$ задача швидкодії є проблемою Маркова на інтервалі $(-1, 1)$ [3], тобто:

$$s_k = \int_0^T g_k(t)u(t)dt, \quad k = 1, ..., n, \quad u(t) \in [-1, 1]. \tag{1.14}$$

За теорією проблеми моментів задача Маркова має єдиний розв'язок тільки за найменшого додатнього числа T, що задовольняє умові (1.13). Більш того, $u^0(t) = \pm 1$ та має не більше, ніж $n-1$ точку розриву, якщо функцій $g_1(t), ..., g_n(t)$ утворюють систему Чебишева. З точки зору теорії керування T є мінімальним часом, за який можливо перевести систему в початок координат, отже, T і є оптимальним часом, а $u^0(t)$ є оптимальним керуванням.

Цей висновок є одним з ключових у процесі розв'язку задачі швидкодії, адже саме він визначає, природу оптимального часу та кількість можливих перемикань в ході переведення системи з початкової точки x_0 в початок координат.

Тепер В.І.Коробов, Г.М.Скляр [4] пропонують розглянути частинний випадок проблеми моментів Маркова на найменшому можливому інтервалі (*Markov moment min-problem*), де $g_k(t) = t^{k-1}, k = 1, ..., n$:

$$s_k = \int_0^\theta t^{k-1}u(t)dt, \quad k = 1, ..., n, \quad |u(t)| \leq 1, \quad \theta \to min. \tag{1.15}$$

На мові теорії керування задача виглядає наступним чином:

$$\begin{aligned} \dot{x_1} &= u, \quad \dot{x_k} = x_{k-1}, \quad k = 2, ..., n, \\ x(0) &= x^0, \quad x(\theta) = 0, \\ |u(t)| &\leq 1, \quad \theta \to min, \end{aligned} \tag{1.16}$$

де $x_k^0 = \dfrac{(-1)^k s_k}{(k-1)!}, k = 1, ..., n$. Якщо $(\theta_s, u_s(t))$ є розв'язком проблеми моментів (1.15), тоді функція $u_s(t)$ набуває значень ± 1 і має $n-1$ точок розриву. Отже, виконується наступна рівність:

$$\sum_{j=1}^{n-1}(-1)^{n-j+1}t_j^k = c_k^{\pm}(\theta_s, s), \quad k = 1, ..., n, \tag{1.17}$$

де $c_k^{\pm}(\theta, s) = \dfrac{1}{2}(\theta^k \mp k s_k)$ знак c_k вказує на знак $u_s(t)$ на останньому інтервалі.

Далі для випадку парного та непарного n вводяться раціональні функції наступним чином:

$$R(z) = \frac{\prod_{j=1}^{m}(z - t_{2j})}{\prod_{j=1}^{m}(z - t_{2j-1})} = 1 - \sum_{k=1}^{\infty}\frac{\gamma_k}{z^k}, \quad n = 2m+1, \tag{1.18}$$

$$R(z) = \frac{\prod_{j=1}^{m}(z - t_{2j-1})}{z\prod_{j=1}^{m-1}(z - t_{2j})} = 1 - \sum_{k=1}^{\infty}\frac{\gamma_k}{z^k}, \quad n = 2m. \tag{1.19}$$

Розклавши $R(z)$ в ряд Лорана, можна зробити висновок [5], що $\gamma_1, ..., \gamma_n$ можна виразити через $c_k = c_k^{\pm}(\theta_s, s), k = 1, ..., n$ наступним чином:

$$\gamma_k = \frac{(-1)^{k-1}}{k!}\begin{vmatrix} c_1 & 1 & 0 & ... & 0 \\ c_2 & c_1 & 2 & ... & 0 \\ ... & ... & ... & & \\ c_{k-1} & c_{k-2} & ... & c_1 & k-1 \\ c_k & c_{k-1} & ... & c_2 & c_1 \end{vmatrix}, \quad q = 2p+1, \tag{1.20}$$

$$\overline{\gamma_k} = \frac{(-1)^{k-1}}{k!}\begin{vmatrix} -c_1 & 1 & 0 & ... & 0 \\ -c_2 & -c_1 & 2 & ... & 0 \\ ... & ... & ... & & \\ -c_{k-1} & -c_{k-2} & ... & -c_1 & k-1 \\ -c_k & -c_{k-1} & ... & -c_2 & -c_1 \end{vmatrix}, \quad q = 2p, \tag{1.21}$$

Основні результати сформульовані в наступній теоремі:

Теорема 1.2 (Korobov Sklyar [1]). *Для будь-якого $s \in \mathbb{R}^n$, розв'язок $(\theta_s, u_s(t))$ проблеми Маркова на найменшому інтервалі (1.15) можна знайти за наступними кроками:*

- *θ_s дорівнює максимальному дійсному кореню рівняння*

$$\Delta_n^+(\theta) \cdot \Delta_n^-(\theta) = 0;$$

- *За наступними умовами можна однозначно знайти точки розриву функції $u_s(t)$:*

$$\Delta_q^+(\theta_s) \cdot \Delta_q^-(\theta_s) = 0,$$
$$(\Delta_q^+(\theta_s))^2 + (\Delta_q^-(\theta_s))^2 \neq 0;$$

- *Якщо $\Delta_q^+(\theta_s) = 0$, то $u_s(\theta_s - 0) = 1$. Аналогічно у випадку $\Delta_q^-(\theta_s) = 0$;*

- *Усі точки розриву $0 < t_1 < ... < t_{q-1} < \theta_s$ функції $u_s(t)$ є коренями наступного рівняння:*

$$\begin{vmatrix} \gamma_2 & \gamma_3 & \cdots & \gamma_{p+1} \\ \cdots & \cdots & \cdots & \cdots \\ \gamma_p & \gamma_{p+1} & \cdots & \gamma_{2p-1} \\ 1 & z & \ldots & z^{p-1} \end{vmatrix} \cdot \begin{vmatrix} -1 & \overline{\gamma_1} & \ldots & \overline{\gamma_p} \\ \overline{\gamma_1} & \overline{\gamma_2} & \ldots & \overline{\gamma_{p+1}} \\ \ldots & \ldots & \ldots & \ldots \\ \overline{\gamma_{p-1}} & \overline{\gamma_p} & \cdots & \overline{\gamma_{2p-1}} \\ 1 & z & \ldots & z^p \end{vmatrix} = 0, \quad q = 2p,$$

або

$$\begin{vmatrix} \overline{\gamma_1} & \overline{\gamma_2} & \ldots & \overline{\gamma_p} \\ \ldots & \ldots & \ldots & \ldots \\ \overline{\gamma_{p-1}} & \overline{\gamma_p} & \cdots & \overline{\gamma_{2p-2}} \\ 1 & z & \ldots & z^{p-1} \end{vmatrix} \cdot \begin{vmatrix} \gamma_1 & \gamma_2 & \ldots & \gamma_p \\ \ldots & \ldots & \ldots & \ldots \\ \gamma_{p-1} & \gamma_p & \cdots & \overline{\gamma_{2p-2}} \\ 1 & z & \ldots & z^{p-1)} \end{vmatrix} = 0, \quad q = 2p-1,$$

де $\gamma_k = \gamma_k^{\pm}(\theta_s)$ та $\overline{\gamma_k} = \overline{\gamma_k}^{\pm}(\theta_s)$ й знак $\pm$ відповідає знаку $u_s(\theta_s - 0)$, визначеного вище.

1.4. Нейронні мережі

Штучна нейронна мережа є математичним аналогом біологічного процесу функціонування нейронних зв'язків у мозку людини. У 1943 році Уоррен Мак-Каллок та Волтер Піттс [8] створили першу математичну модель штучного нейрона, яка відображала базові принципи роботи біологічного нейрона. Модель Мак-Каллока та Піттса описує простий нейрон як вузол, який отримує вхідні сигнали, обробляє їх і генерує вихідний сигнал. Вхідні сигнали мають ваги, які відповідають сили зв'язку між нейронами. Нейрон обчислює зважену суму вхідних сигналів і застосовує нелинійну функцію активації до цієї суми. Вихідний сигнал нейрона передається до інших нейронів як вхідний сигнал.

У 1957році Френк Розенблатт розробив перцептрон [9] - першу нейронну мережу, здатну до навчання. Вона використовувала принципи зворотного розповсюдження помилки для коригування ваг і досягала успішних результатів у вирішенні простих задач класифікації.

Завдяки роботі Джеффрі Гінтона [10] та інших дослідників, зворотне розповсюдження помилки стало основним методом тренування нейронних мереж. Це призвело до збільшення застосування нейронних мереж у багатьох галузях, включаючи комп'ютерне зорове сприйняття та обробку природної мови.

Переломним моментом в історії розвитку теми нейронних мереж став 1998 рік, коли Ян Лекун запропонував поняття [11] "глибокі нейронні мережі маючи на увазі використання нейронних мереж зі значним числом шарів для розв'язання складних завдань.

Останні роки завдяки великому обсягу даних, потужності обчислень і покращенню алгоритмів навчання нейронних мереж, стався значний ривок в галузі глибокого навчання. Мережі, такі як згорткові нейронні мережі

(CNN) і рекурентні нейронні мережі (RNN), здобули визнання в багатьох задачах комп'ютерного зору, обробки мови, голосового і рекомендаційного аналізу тощо.

Введемо декілька необхідних визначень. Однією з найголовніших складових нейронної мережі є шар (layer). Шари бувають трьох типів:

- *Вхід* На цьому етапі в структуру нейронної мережі вводяться початкові дані. Зазвичай цей шар у векторній формі.
- *Прихований* На цьому етапі відбувається процес тренування мережі. На $k-$му шарі $i-$й вузол(нейрон) може бути представлений наступним чином:

$$x_m^k = \theta(s_m^k) = \theta(\sum_{i=0}^{d^{k-1}} w_{im}^k x_i^{k-1}), \tag{1.22}$$

  де θ це функція активації, а w ваги.
- *Вихід* Зазвичай, це вектор, що є результатом роботи функції активації. Всі елементи приймають значення 0 або 1.

Зв'язком між усіма прихованими шарами є *ваги*. Їх основна задача це оцінювати, чи є попередній результат важливим. Функція ваг визначається наступним чином:

$$W_{im}^k = \begin{cases} 1 \le k \le K, & \text{шари} \\ 0 \le i \le d^{k-1}, & \text{входи} \\ 1 \le m \le d^k, & \text{виходи} \end{cases} \tag{1.23}$$

Важливим етапом побудови нейронної мережі є вибір функції активації. Така функція на кожній ітерації вирішує, чи потрібно запускати алгоритм нейронної мережі. Фактично, процес прийняття рішення є способом боротьби з лінійністю моделі. Найбільш поширеними є наступні функції активації:

- *Sigmoid* $S(x) = \dfrac{e^x}{1 + e^x}$
- *tanh* $\tanh(x) = \dfrac{e^{2x} - 1}{e^{2x} + 1}$
- *ReLU* $ReLU(x) = \max(0, z)$

Поняття *епохи* вводиться як показник кількості оновлення вагів. *Функція втрат* оцінює наближення до реального результату. Зазвичай вона розраховується як середня квадратична помилка.

Існує кілька типів нейронних мереж, кожен з яких призначено для конкретних задач.Найпростіший тип це *нейронні мережі прямого зв'язку (FNN)*, де інформація обробляється лише в одному напрямку, від вхідного рівня до вихідного. Вони зазвичай використовуються для таких завдань, як класифікація та регресія. *Згорточні нейронні мережі (CNN)* використовуються для завдань обробки зображень і відео. Вони складаються зі згорткових шарів, які застосовують фільтри для вилучення відповідних функцій із вхідних даних. CNN відомі своєю здатністю вловлювати просторові та ієрархічні шаблони в даних. Повторювані нейронні мережі (RNN) призначені для обробки послідовних даних, де вихідні дані попереднього кроку повертаються як вхідні дані для поточного кроку. Цей механізм зворотного зв'язку дозволяє RNN фіксувати часові залежності в даних, що робить їх придатними для таких завдань, як моделювання мови, розпізнавання мови та аналіз часових рядів. *Мережі довгострокової короткочасної пам'яті (LSTM)* це особливий тип RNN, який може ефективно вивчати довгострокові залежності.

1.5. Застосування нейронних мереж для розв'язку класичних математичних задач

Зараз застосування алгоритмів машинного навчання для розв'язку математичних задач стає все популярнішим. Нейронні мережі мають значний потенціал у розумінні складних математичних залежностей та здатності до навчання на великих обсягах даних. Алгоритми мереж застосовуються для задач різного характеру, наприклад:

Класифікація:Нейронні мережі можуть використовуватись для класифікації об'єктів на підставі їх властивостей або характеристик. Одне з найпоширеніших застосувань це використання мережі для розпізнавання рукописних цифр або класифікації зображень на кілька категорій.

Моделювання:Нейронні мережі можуть бути використані для прогнозування числових значень на основі вхідних даних. Наприклад, прогнозування цін на нерухомість або прогнозування попиту на товари.

Оптимізація: Ряд оптимізаційних задач, таких як мінімізація функцій великої кількості змінних або знаходження оптимального шляху в графах, може бути розв'язаний за допомогою алгоритмів нейронних мереж.

Апроксимація функцій: Нейронні мережі можуть бути використані для апроксимації складних математичних функцій або наближення функцій, для яких немає аналітичного виразу.

1.6. Застосування нейронних мереж для розв'язку задач теорії керування

Інтеграція нейронних мереж у процес розв'язку задач теорії керування була предметом цікавості протягом кількох десятиліть. Проте, такі методи були популярні серед дослідників, що займались теорією керування в інженерному сенсі. Вперше ідея машинного навчання була застосована до області адаптивного керування.

Однією з перших спроб застосування нейронних мереж до розв'язку задач теорії керування було з ціллю апроксимації функцій в адаптивних системах керування [12]. Автори зробили значний внесок у галузі, включаючи розробку алгоритмів адаптивного керування на основі нейронної мережі. Вони запропонували такий метод, як ADP (адаптивне динамічне програмування).

Л.Люнг зробив внесок у сферу ідентифікації систем та адаптивного керування. Він із командою запропонували використання нейронних мереж для ідентифікації та керування системи [13]. Робота була зосереджена на використанні нейронних мереж для оцінки динаміки складних систем і розробки адаптивних законів керування на основі визначених моделей.

С.Хайкін зробив внесок у розробку адаптивних алгоритмів керування на основі нейронних мереж. Він є автором книги «Нейронні мережі: всебічна основа» [14], в якій обговорюється інтеграція нейронних мереж і адаптивного керування, включаючи застосування до різних проблем керування.

Н.Наір застосував ідею навчання з підкріпленням до роботи з адаптивного контролю. Він розробив алгоритми, що поєднують нейронні мережі та методи адаптивного керування, зокрема в області еталонного адаптивного керування моделлю [15].

Розділ 2. Аналітичний алгоритм

Задача, що описана вище, вже є розв'язаною для довільного n. Більш того, сформовано алгоритм для пошуку оптимального часу, моментів перемикань та типу керування [6]. Зазначимо, що ми кажемо, що кусково-неперервна функція $u = u(t) = \pm 1$ при $t \in [0, \theta]$ має *перший* тип, якщо $u(\theta - 0) = -1$ та $u(\theta - 0) = 1$ у випадку керування *другого* типу.

1. Побудуємо наступні поліноми:

$$c_k^{\pm} = c_k^{\pm}(x^0, \theta) = \frac{\pm(-1)^k k! x_k^0 + \theta^k}{2}, k = 1, ..., n, \tag{2.1}$$

$$\gamma_k^{\pm} = \gamma_k^{\pm}(x^0, \theta) = \frac{1}{k}(c_k^{\pm} - \sum_{i=1}^{k-1} \gamma_{k-i}^{\pm} c_i^{\pm}), k = 1, ..., n, \tag{2.2}$$

$$\Delta_{2k+1}^{\pm} = \Delta_{2k+1}^{\pm}(x^0, \theta) = det\{\gamma_{i+j+1}^{\pm}\}_{i,j=0}^{k}, \quad 0 \le k \le \frac{n-1}{2}, \tag{2.3}$$

$$\Delta_{2k}^{\pm} = \Delta_{2k}^{\pm}(x^0, \theta) = det\{\gamma_{i+j}^{\pm}\}_{i,j=1}^{k}, \quad 1 \le k \le \frac{n}{2}. \tag{2.4}$$

2. Знайдемо дійсні корені θ^+, θ^- рівнянь $\Delta_n^+(x^0, \theta) = 0$ $\Delta_n^-(x^0, \theta) = 0$ відповідно. Тоді оптимальний час θ_0 буде максимальним значенням серед θ^+, θ^-, тобто $\theta_0 = max(\theta^+, \theta^-)$

3. Визначимо таке $0 \le p \le n-1$, що

$$\Delta_{p+1}^+(x^0, \theta_0) \cdot \Delta_{p+1}^-(x^0, \theta_0) = 0, \tag{2.5}$$

$$(\Delta_{p+1}^+(x^0, \theta_0))^2 + (\Delta_{p+1}^-(x^0, \theta_0))^2 > 0. \tag{2.6}$$

4. Якщо $\theta^+ > \theta^-$ тоді керування першого типу. У випадку $\theta^+ < \theta^-$ керування другого типу.

5. Останнім кроком є моменти переключень. Ми можемо їх знайти за наступними формулами. У випадку $p = 2s$:

$$\begin{vmatrix} \gamma_1 & \gamma_2 & \cdots & \gamma_{s+1} \\ \gamma_2 & \gamma_3 & \cdots & \gamma_{s+2} \\ \cdots & \cdots & \cdots & \cdots \\ \gamma_s & \gamma_{s+1} & \cdots & \gamma_{2s} \\ 1 & z & \ldots & z^s \end{vmatrix} \cdot \begin{vmatrix} \gamma_1 & \gamma_2 & \cdots & \gamma_{s+1} \\ \gamma_2 & \gamma_3 & \cdots & \gamma_{s+2} \\ \cdots & \cdots & \cdots & \cdots \\ \gamma_s & \gamma_{s+1} & \cdots & \gamma_{2s} \\ w_0(z) & w_1(z) & \ldots & w_s(z) \end{vmatrix} = 0 \qquad (2.7)$$

якщо $p = 2s - 1$

$$\begin{vmatrix} \gamma_2 & \gamma_3 & \cdots & \gamma_{s+1} \\ \gamma_3 & \gamma_4 & \cdots & \gamma_{s+2} \\ \cdots & \cdots & \cdots & \cdots \\ \gamma_s & \gamma_{s+1} & \cdots & \gamma_{2s-1} \\ 1 & z & \ldots & z^{s-1} \end{vmatrix} \cdot \begin{vmatrix} \gamma_2 & \gamma_3 & \cdots & \gamma_{s+1} \\ \gamma_3 & \gamma_4 & \cdots & \gamma_{s+2} \\ \cdots & \cdots & \cdots & \cdots \\ \gamma_s & \gamma_{s+1} & \cdots & \gamma_{2s-1} \\ w_1(z) & w_2(z) & \ldots & w_s(z) \end{vmatrix} = 0 \qquad (2.8)$$

при $w_j(z) = z^j - \sum_{i=0}^{j-1} \gamma_{j-i} z^i, j = 0, ..., s$

Корені рівняння (2.7) або (2.8) є точками перемикання.

Розділ 3. Структура алгоритму для розв'язку задач теорії керування

Існує два основних типи машинного навчання: контрольоване (supervised) та неконтрольоване (unsupervised). Вони відрізняються доступністю позначених навчальних даних.

Контрольоване навчання передбачає навчання моделі з використанням позначених прикладів, де кожен приклад складається з вхідних даних і відповідних вихідних міток. Мета полягає в тому, щоб вивчити функцію відображення, яка може точно передбачати мітки для нових, невидимих вхідних даних. Модель вивчає дані з мітками, мінімізуючи розбіжності між прогнозованими результатами та справжніми мітками.

Неконтрольоване навчання має справу з даними без міток, де вхідні дані не мають відповідних вихідних міток. Замість того, щоб передбачати конкретні мітки, мета полягає в тому, щоб виявити закономірності, структури або зв'язки в даних. Алгоритми неконтрольованого навчання спрямовані на вилучення значущих представлень або кластерів із даних без попереднього знання вихідних міток.

Виходячи з того, що покроковий алгоритм розв'язку задачі швидкодії існує, нам доцільніше використовувати контроловний тип машинного навчання.

3.1. Створення необхідних наборів даних

Було створено дві пари датасетів: тренувальний та валідаційний. Кожен з них має набір початкових точок x^0 розмірності $(10000 \times n)$ та набір точок перемикання і типу керування $(10000 \times n)$ (така розмірність обумовлена кількістю перемикань $n-1$ та ще однією колонкою, кожен елемент якої приймає значення 0, якщо керування *першого* типу та 1, якщо *другого*). Аналогічну структуру мають валідаційні дані.

Набори даних заповнювались функцією, що була написана за аналітичним алгоритмом розв'язку задачі швидкодії на мові програмування Python. Дана функція на вхід приймає початкову точку x_0, а результатом надає $n-$мірний вектор із точками перемикання $t_i, i = 1, ..., n-1$ та типом керування (перший або другий). За допомогою даної функції створюємо набори тренувальних та валідаційних даних як на Рис.3.1.

3.2. Алгоритм нейронної мережі

Для розв'язку задачі було використано нейронну мережу прямого зв'язку з кількома прихованими шарами.

Функції активації: приховані шари використовують функцію активації гіперболічного тангенса (tanh), а передостанній шар використовує функцію активації випрямленої лінійної одиниці (ReLU). Останній шар використовує функцію лінійної активації.

Ініціалізація ваги (Weight initialization): Матриці ядра (ваги) шарів ініціалізуються за допомогою різних стратегій випадкової ініціалізації: «random normal» для шарів активації tanh і «random uniform» для шару активації ReLU.

Функція втрат і оптимізатор: мережа скомпільована з використанням рі-

Index	1	2	3	4
0	-1.77254	-1.86624	-0.27028	2.91505
1	0.470871	-0.248332	2.32949	2.59632
2	1.63551	2.79344	-2.0837	0.556533
3	0.866967	-1.96146	-1.93672	-2.96898
4	-0.377059	2.22018	-1.42673	-2.98046
5	-0.574639	-0.321158	1.48285	2.63693

(а) Початкові умови для тренувального датасету

Index	1	2	3	control_type
0	10.5251	15.1787	4.70736	1
1	4.04653	7.30976	0.863193	0
2	7.59279	14.4014	0.973154	0
3	5.44336	8.63111	1.3047	1
4	1.67232	4.33691	0.15216	1
5	3.61104	5.63783	1.53287	0

(б) Моменти переключення та тип керування тренувального датасету

Рис. 3.1: Тренувальний набір даних

зних функцій втрат і оптимізаторів. Перша компіляція використовує оптимізатор RMSprop із функцією втрат середньоквадратичної помилки (MSE). Друга компіляція також використовує оптимізатор RMSprop, але з функцією втрати середньої абсолютної відсоткової помилки (MAPE). Третя компіляція використовує оптимізатор Адама з функцією втрат середньоквадратичної помилки (MSE).

Під час тренування функції рахується функція витрат, що є середньою квадратичною похибкою між очікуваним результатом та прогнозом моделі. Для нашої задачі динаміка функції витрат упродовж тренування моделі виглядає як показано на Рис.3.2.

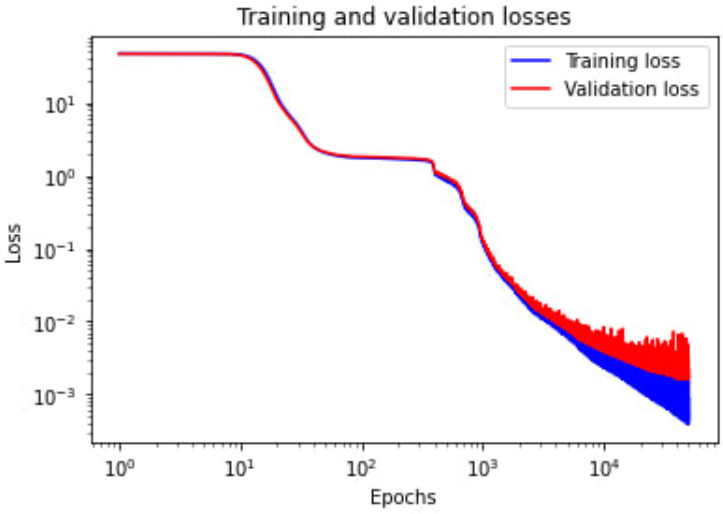

Рис. 3.2: Функція витрат тренувального й валідаційного датасетів.

Можна відмітити, що функція втрат для тренувального та валідаційного наборів даних веде себе майже однаково. Також явно помітні два плато на проміжках $(10 - 100)$ й $(1000 - 10000)$. Це може бути пов'язано з накопиченням обчислювальних похибок під час підбору коєфіцієнтів вагів.

Найкраща модель зберігається та в подальшому використовується для швидкого пошуку розв'язків.

Розділ 4. Випробовування мережі на прикладі

Після того як модель натренувалася (підібрано усі ваги (*weights*)) та найкраща модель збережена, можна перевірити якість алгоритму на тестових даних.

Для генерації тестувального набору даних використовується такий самий принцип, що й при формуванні тренувального та валідаційного. Тобто створюється дві матриці $(10000 \times n)$. Кожен рядок першої відповідає початковій точці x_0. Три колонки другої матриці відповідають моментам переключення системи, а остання визначає тип керування (перший чи другий). Для перевірки точності моделі також було утворено створено датасет із правильними результатами.

Під час випробовування матриці на тестових даних ми отримали наступний результат (Рис. 4.1). Можемо побачити, що результати є дуже близькими, але якщо подивитися на графік похибок, то виявиться, що існує декілька піків, де помилка майже досягає значення 1. Цей ефект має прямий зв'язок з ефектом плато на графіку похибок. Для випробовування нейронної мережі також було узяту початкову точку, що була запропонована авторами алгоритму у своєму прикладі [6]. Класичний алгоритм пошуку точок перемикань та виду керування в задачі керування для початкової точки $x_0 = (1, -1, 1, -1)$ розмірності $n = 4$ дає результат $t_1 = 0.1185222964, t_2 = 1.539741406, t_3 = 2.425990476$ (керування *першого* ти-

0	1	2	3
3.46465	7.25667	0.839763	0.00347899
6.17579	11.4303	0.691161	-0.0123059
4.18163	8.34481	0.697296	-0.000755381
3.50526	5.43292	0.800513	0.0322051
10.9242	20.4802	1.27542	0.0143519
7.01774	8.39238	4.62512	-0.0200595

(а) Результат нейронної мережі

0	1	2	3
3.55379	7.32109	0.89996	0
6.14046	11.4537	0.643245	0
4.17407	8.33057	0.678243	0
3.4436	5.40354	0.720298	0
11.0008	20.5279	1.31296	0
6.99354	8.4532	4.4904	0

(б) Реальний результат

Рис. 4.1: Тестування нейронної мережі

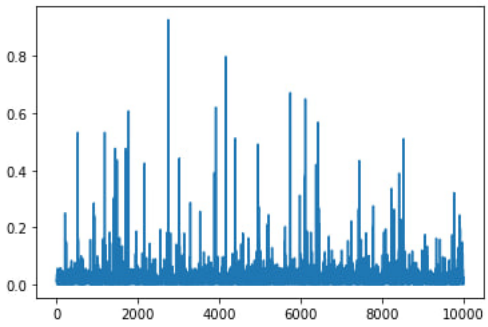

Рис. 4.2: Графік похибок тестових даних

пу $\theta^+ > \theta^-$) за 0.039976 секунди [6]. Натренована нами нейронна мережа надає результат $t_1 = 0.16607359, t_2 = 2.4586797, t_3 = 1.5499189$ (керування *першого* типу Рис.4) за 0.0396279 секунди. Також алгоритм перевірено

```
[1.5499489  2.4586797  0.16607359 0.9626119 ]
[1.53974141 2.42599048 0.1185223  1.        ]
```

Рис. 4.3: Порівняння результатів

для $n = 10$. Для цього було створено тренувальний та валідаційний набори даних аналогічно випадку $n = 4$. Таку розмірність було обрано з декількох причин. По-перше, з точки зору утворення необхідних дата сетів. Як

вже було зазначено, складнісь розв'язку задачі суттєво збільшується із підвищенням розмірності. Тож утворення необхідного набору даних вже для $n = 10$ та кількості експериментів (рядків) 10000 займає чимало часу (точна кількість часу). Однак, у той самий час $n = 10$ вже вистачає для того, щоб показати переваги використання нейронної мережі на високих розмірностях, адже класичному алгоритму у процесі пошуку рішеня доводиться формувати поліноми степеня 30 і знаходити його корені.

Створення чотирьох наборів даних розмірності (30x10000) в сумі зайняло 8 годин, що більше, ніж для розмірності $n = 4$ майже у вісім разів (у випадку $n = 4$ необхідні дата сети було сформовано за 1 годину 20 хвилин). Процес тренуання нейронної мережі зайняв дві години. Поведінка функції втрат виглядає наступним чином:

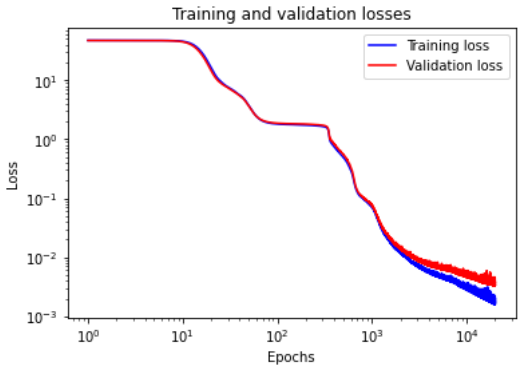

Рис. 4.4: функція втрат для $n = 10$

Результатом класичного алгоритму для початкової точки

$$x_0 = (1, -1, 1, -1, 1, -1, 1, -1, 1, -1)$$

є наступні моменти перемикань та другий тип керування (програма показує 0 для першого типу керування, та 1 для другого):

$$t_1 = 7.26, \quad t_2 = 5.18, \quad t_3 = 2.78, \quad t_4 = 1.,$$
$$t_5 = 8.01, \quad t_6 = 6.32, \quad t_7 = 3.93, \quad t_8 = 1.59,$$
$$t_9 = 0.04, \quad type = 1.$$

Результатом роботи нейронної мережі є

$$t_1 = 7.7, \quad t_2 = 5.63, \quad t_3 = 2.85, \quad t_4 = 1.16,$$
$$t_5 = 8.03, \quad t_6 = 6.68, \quad t_7 = 4.2, \quad t_8 = 1.86,$$
$$t_9 = 0.44, \quad type = 1.$$

Похибка виглядає наступним чином:

$$res = (0.56, 0.21, 0.14, 0.42, 0.02, 0.17, 0.35, 0.34, 0.02, 0.24).$$

Тож можна зробити висновок, що аналогічний алгоритм для розмірності $n = 10$ дає значно більшу похибку. Це пов'язано у першу чергу із накопиченням помилок упродовж трансформації вагів векторів.

Зважаючи, на низьку точність роботи мережі на високій розмірності, було проведено додатковий експеримент. У ньому на меті поставлено пошук не усіх моментів перемикання, а тільки оптимального часу θ та типу керування. Відповідні набори даних було створено на основі попередніх для розмірності $n = 10$. Через те, що попередній набір даних включав в себе оптимальний час θ(максимум серед усіх $t_i, i = 1, 2, ..., n-1$) та тип керування, нові датасети утворювались простим переносом та вибором максимального значення.

Процес тренування нейронної мережі зайняв 1 годину 30 хвилин та надав наступні результати для початкової точки

$$x_0 = (1, -1, 1, -1, 1, -1, 1, -1, 1, -1):$$

```
network: [8.1143, 1.0264]
actual: [8.1011, 1]
```

Рис. 4.5: θ, тип керування

Середня квадратична похибка дорівнює *0.021*.

Таким чином, можна зробити висновок, що на високих розмірностях доцільніше шукати лише значення θ, а не моменти перемикань. Більш того

процес застосування нейронної мережі до пошуку рішення займає 0.142276 секунди, для такої ж початкової точки класичний алгоритм рахує відповідь 3.07секунди.

Зауважимо, що найбільш складною частиною розв'язання задачі швидкодії є саме побудова поліномів (2.3), (2.4) для пошуку θ, адже необхідно формувати визначник матриці, кожен елемент якої є поліномом. Однак, якщо θ відоме, то моменти перемикання можна знайти за формулами (2.7), (2.8), розклавши визначники за останніми рядками.

Таким чином, для більшої точності на високих розмірностях задачу можна поділити на дві великі частини: пошук оптимального часу θ та пошук моментів перемикання $t_i, i = 1, ..., n-1$. Першу з яких розв'язувати за допомогою алгоритму нейронної мережі, а другу відповідною частиною класичного алгоритму.

Висновки

Отже, використання нейронних мереж для вирішення задачі швидкодії є важливим та ефективним методом апроксимації моментів перемикання та пошуку типу керування. Отримані результати показують, що натренована мережа досягає точності, яка відрізняється від справжньої величини на порядок 10^{-2}, при цьому витрачаючи менше часу, ніж традиційні алгоритми.

Цей висновок є значущим з наукової точки зору, оскільки підкреслює потенціал нейронних мереж у вирішенні задач оптимізації та швидкодії. Застосування таких моделей може виявитись корисним у різних сферах, включаючи автоматизацію, транспорт, робототехніку та інші області, де важливо досягти високої точності та ефективності при обробці складних задач.

Проте, варто враховувати, що дослідження може бути обмежене деякими факторами. Наприклад, потрібно враховувати розмір та складність навчального набору даних, а також параметри самої нейронної мережі, такі як кількість шарів та розмір кожного шару. Додатковою роботою може бути дослідження різних архітектур мереж та методів навчання з метою поліпшення результатів.

Можливі напрями подальших досліджень

Наступним кроком варто розглянути застосування алгоритму нейронної мережі до розв'язку нелінійної задачі швидкодії наступного вигляду:

$$\dot{x} = a(t,x) + b(t,x)u, \quad a(t,0) \equiv 0, \quad u \in \mathbb{R}, \tag{4.1}$$

де $a(t,x), b(t,x)$ це вектор-функції в околі нуля. Умова $a(t,0) \equiv 0$ означає, що початок координат є точкою покою для системи. Обмеження керування $|u(t)| \leq 1$ лишається з лінійної задачі.

Навчання із "вчителем" виявляється недоцільним для розмірностей, що перевищують $n = 10$, через високі обчислювальні вимоги та значний час, необхідний для створення великих тренувальних та валідаційних наборів даних. Однак, можна розглянути альтернативний підхід, що базується на зворотній задачі. Конкретніше, ми можемо спробувати визначити початкову точку системи, знаючи фінальну точку та оптимальний час, необхідний для переведення системи в початок координат.

Одним з можливих підходів є формування функції втрат як середньоквадратичної похибки системи від оптимального керування та фінальної точки. У рамках цього підходу можна застосувати оптимізаційні методи для мінімізації функції втрат та знаходження оптимальних параметрів моделі. Наприклад, можна використовувати градієнтний спуск або його варіації для ефективного оновлення ваг моделі. Також можна розглянути використання різноманітних алгоритмів оптимізації, таких як методи другого порядку або методи стохастичного градієнтного спуску.

Також можливим підходом є використання архітектур нейронних мереж, що підтримують паралельні обчислення, такі як глибокі згорткові мережі (CNN) або рекурентні нейронні мережі (RNN). Ці архітектури можуть допомогти впоратися зі складними завданнями та робити обчислення більш ефективними.

Список використаних джерел

[1] V. I. Korobov, G. M. Sklyar, Time-optimality and the power moment problem. *Math. USSR-Sb.* **62** (1989), pp. 185-206.

[2] N. N. Krasovskiĭ, Concerning the theory of optimal control. *Automation and Remote Control*, **18** (1957), pp. 1005–1016.

[3] M. G. Kreĭn, A. A. Nudel'man, The Markov moment problem and extremal problems. Ideas and problems of P. L. Cebysev and A. A. Markov and their further development. Translations of Mathematical Monographs **50**, AMS, Providence, R.I., 1977, 417 p.

[4] V. I. Korobov, G. M. Sklyar, The Markov moment min-problem and time optimality. *Siberian Math. J.* **32** (1991), pp. 46-55.

[5] G. M. Sklyar, S. Yu. Ignatovich, Development of the Markov moment problem approach in the optimal control theory, *Methods of Functional Analysis and Topology* **13** (2007), no. 4, pp. 386–400.

[6] V. I. Korobov, G. M. Sklyar, S.Yu.Ignatovich Solving ot the polynomial systems arising in the linear time-optimal control. *Communications in Mathematical Analysis*, Conference 03 (2011), pp. 153-171.

[7] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, E. F. Mishchenko, The mathematical theory of optimal processes. Interscience Publishers John Wiley & Sons, Inc., New York-London, 1962, 360 p.

[8] W.McCulloch, W.Pitts, A Logical Calculus of the Ideas Immanent in Nervous Activity, *Bulletin of Mathematical Biology* **52** (1990), No. 1/2, pp. 99-115. Reprinted from the Bulletin of Mathematical Biophysics, Vol. 5, pp. 115-133 (1943).

[9] F. Rosenblatt, The perceptron a perceiving and recognizing automation, 1957, Cornel Aeronautical Labolatory Inc. Buffalo, New York

[10] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, *Nature*, **323** (1986), pp. 533–536.

[11] P.Simard, L.Bottou, P.Haffner, Y.LeCun, Boxlets: A Fast Convolution Algorithm for Signal Processing and Neural Networks. Advances in Neural Information Processing Systems, 1998, MIT Press, pp. 571-577.

[12] Ssu-Hsin Yu, A.M.Annaswamy, Adaptive control of nonlinear dynamic systems using θ-adaptive neural networks, Automatica, pp. 1975-1995, 1997; doi.org/10.1016/s0005-1098(97)00130-1

[13] J.Oberg, Q.Zhang, L.Ljung,A.Benveniste, B.Delyon, P.-Y. Glorennec, H.Hjalmarsson, A. Juditsky, Nonlinear black-box modeling in system identification: a unified overview,Automatica, pp.1691-1724, 1995

[14] S.Haykin, Neural Networks: A Comprehensive Foundation. Prentice Hall, 1994, 842 p.

[15] S.-L.Chang, S.S.Nair, A New Neural Network Control Architecture for a Class of Nonlinear Dynamic Systems, 1993 American Control Conference, IEEE, 1993