**УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ**
**ЗАЛІЗНИЧНОГО ТРАНСПОРТУ**

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КЕРУЮЧИХ СИСТЕМ**
**ТА ТЕХНОЛОГІЙ**

**Кафедра інформаційних технологій**

# МЕТОДИЧНІ ВКАЗІВКИ

**до лабораторних робіт**

**з дисципліни**

**«*ШТУЧНІ НЕЙРОННІ МЕРЕЖІ*»**

**Харків — 2024**

Методичні вказівки розглянуто і рекомендовано до друку на засіданні кафедри інформаційних технологій 03 червня 2024 р., протокол № 10.

Методичні вказівки призначено для здобувачів вищої освіти першого (бакалаврського) рівня спеціальності 126 «Інформаційні системи та технології» усіх форм навчання, що вивчають дисципліну «Штучні нейронні мережі».

Укладач

старш. викл. О. І. Іванюк

Рецензент

доц. Н. А. Корольова

**ЗМІСТ**

# ВСТУП

Метою методичних вказівок є надання здобувачам теоретичних знань і практичних навичок, необхідних для розуміння та застосування штучних нейронних мереж у різних галузях. Це охоплює вивчення архітектур повнозв'язних, згорткових і рекурентних нейронних мереж, а також їхнє практичне застосування для вирішення завдань класифікації, регресії, обробки зображень та послідовних даних.

Лабораторні роботи спрямовані на розвиток умінь самостійного дослідження нейронних мереж, виконання експериментів з різними параметрами й архітектурами, а також аналізу та інтерпретації результатів. Здобувачі працюватимуть із популярними фреймворками для машинного навчання, такими як Keras, що дасть змогу їм набути сучасних практичних навичок.

Завдання лабораторних робіт містять чіткі інструкції, приклади виконання завдань та запитання для самоконтролю, що допоможе здобувачам глибше зрозуміти матеріал і підготуватися до захисту робіт.

## Лабораторна робота 1
# ДОСЛІДЖЕННЯ ТА ЗАСТОСУВАННЯ ПОВНОЗВ'ЯЗНИХ НЕЙРОННИХ МЕРЕЖ ПРЯМОГО ПОШИРЕННЯ

### Мета роботи

Дослідити архітектуру та принципи роботи повнозв'язних нейронних мереж (НМ) прямого поширення. Набути навички застосування повнозв'язних нейронних мереж прямого поширення для аналізу структурованих даних.

### Порядок виконання роботи

1 За матеріалами лекцій вивчити теоретичні відомості.
2 Виконати практичні завдання.
3 Оформити звіт з лабораторної роботи.
4 Захистити роботу у викладача.

### Практичні завдання

### Завдання 1.1

### Постановка завдання

Виконати обчислення відповіді нейрона на набір прикладів, використовуючи вихідні дані за варіантом. Заокруглення виконувати до двох знаків після коми.

Архітектуру нейронної мережі наведено на рисунку 1.1.

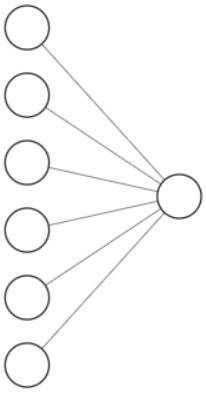

Рисунок 1.1 — Архітектура НМ для завдання 1

**Вихідні дані**

Варіанти вихідних даних до завдання доступні за посиланням [7].

**Приклад виконання завдання**

Приклад є спрощеною версією завдання; завдання за варіантами передбачає більший обсяг розрахунків.

Архітектуру нейронної мережі наведено на рисунку 1.2.

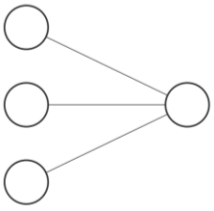

Рисунок 1.2 — Архітектура НМ для прикладу до завдання 1.1

Вихідні дані:

– матриця навчальних прикладів:

$$X = \begin{bmatrix} 3 & -9 \\ -5 & -3 \\ -6 & 7 \end{bmatrix};$$

– вектор вагових коефіцієнтів:

$$w = \begin{bmatrix} 0.9 \\ -0.1 \\ 1 \end{bmatrix};$$

– значення зміщення:

$$b = -0.2;$$

– активаційна функція:

$$\varphi = \tanh(z).$$

Обчислення відповіді нейрона.

Визначимо значення суматорної функції нейрона:

$$\begin{aligned} z = w^T X + b &= \begin{bmatrix} 0.9 \\ -0.1 \\ 1 \end{bmatrix}^T \cdot \begin{bmatrix} 3 & -9 \\ -5 & -3 \\ -6 & 7 \end{bmatrix} + (-0.2) = \\ &= \begin{bmatrix} 0.9 & -0.1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 & -9 \\ -5 & -3 \\ -6 & 7 \end{bmatrix} + (-0.2) = \\ &= \begin{bmatrix} 0.9 \cdot 3 + (-0.1) \cdot (-5) + 1 \cdot (-6) & 0.9 \cdot (-9) + (-0.1) \cdot (-3) + 1 \cdot 7 \end{bmatrix} - 0.2 = \\ &= [-2.8 \quad -0.8] - 0.2 = [-3 \quad -1]. \end{aligned} \tag{1.1}$$

Визначимо значення активації, що і є відповіддю нейрона на набір прикладів:

$$
\begin{aligned}
&a = \varphi(z) = \tanh(z) = \tanh\left(\begin{bmatrix} -3 & -1 \end{bmatrix}\right) = \\
&= \begin{bmatrix} \tanh(-3) & \tanh(-1) \end{bmatrix} = \begin{bmatrix} \dfrac{e^{-3} - e^{3}}{e^{-3} + e^{3}} & \dfrac{e^{-1} - e^{1}}{e^{-1} + e^{1}} \end{bmatrix} = \\
&= \begin{bmatrix} \dfrac{0.05 - 20.09}{0.05 + 20.09} & \dfrac{0.37 - 2.72}{0.37 + 2.72} \end{bmatrix} = \begin{bmatrix} \dfrac{-20.04}{20.14} & \dfrac{-2.35}{3.09} \end{bmatrix} = \\
&= \begin{bmatrix} -1 & -0.76 \end{bmatrix}.
\end{aligned}
\tag{1.2}
$$

**Завдання 1.2**

**Постановка завдання**

Виконати обчислення прямого поширення у нейронній мережі, використовуючи вихідні дані за варіантом. Заокруглення виконувати до двох знаків після коми.

Архітектуру нейронної мережі наведено на рисунку 1.3.

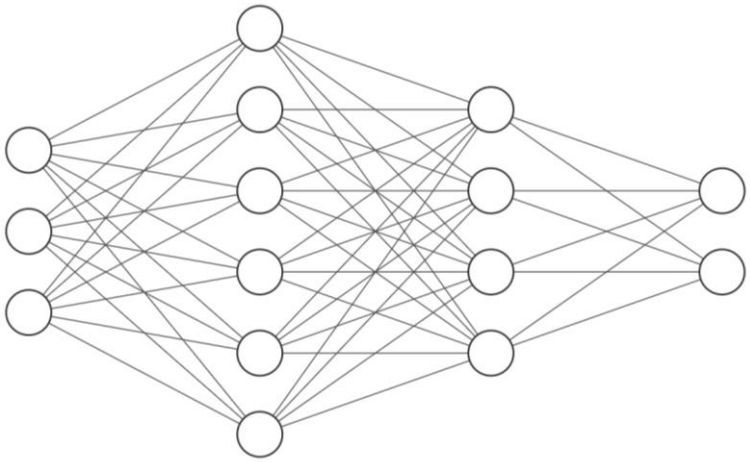

Рисунок 1.3 — Архітектура НМ для завдання 2

**Вихідні дані**

Варіанти вихідних даних до завдання доступні за посиланням [8].

**Приклад виконання завдання**

Приклад є спрощеною версією завдання; завдання за варіантами передбачає більший обсяг розрахунків.

Архітектуру нейронної мережі наведено на рисунку 1.4.

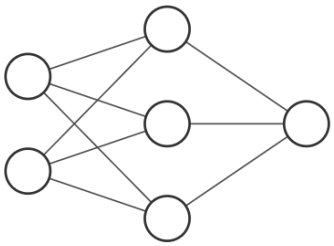

Рисунок 1.4 — Архітектура НМ для прикладу до завдання 1.2

Вихідні дані:

– матриця вхідних прикладів:

$$X = \begin{bmatrix} -2 \\ 3 \end{bmatrix};$$

– матриця вагових коефіцієнтів прихованого шару:

$$W^{[1]} = \begin{bmatrix} -1 & 9 \\ 0 & 4 \\ 6 & -3 \end{bmatrix};$$

– вектор зміщень прихованого шару:

$$b^{[1]} = \begin{bmatrix} -8 \\ -3 \\ 6 \end{bmatrix};$$

– матриця вагових коефіцієнтів вихідного шару:

$$W^{[2]} = \begin{bmatrix} -7 & 2 & -3 \end{bmatrix};$$

– вектор зміщень вихідного шару:

$$b^{[2]} = \begin{bmatrix} -6 \end{bmatrix};$$

– активаційна функція прихованого шару:

$$\varphi^{[1]}(z) = \text{relu}(z) = \max(0, z);$$

– активаційна функція вихідного шару:

$$\varphi^{[2]}(z) = \sigma(z).$$

Обчислення прямого поширення.

Визначимо значення суматорної функції першого шару нейронної мережі:

$$Z^{[1]} = W^{[1]}X + b^{[1]} = \begin{bmatrix} -1 & 9 \\ 0 & 4 \\ 6 & -3 \end{bmatrix} \cdot \begin{bmatrix} -2 \\ 3 \end{bmatrix} + \begin{bmatrix} -8 \\ -3 \\ 6 \end{bmatrix} = \begin{bmatrix} 21 \\ 9 \\ -15 \end{bmatrix}. \qquad (1.3)$$

Визначимо активацію першого шару нейронної мережі:

$$A^{[1]} = \text{relu}(Z^{[1]}) = \text{relu}\left(\begin{bmatrix} 21 \\ 9 \\ -15 \end{bmatrix}\right) = \begin{bmatrix} 21 \\ 9 \\ 0 \end{bmatrix}. \quad (1.4)$$

Визначимо значення суматорної функції вихідного шару нейронної мережі:

$$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]} = \begin{bmatrix} -7 & 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 21 \\ 9 \\ 0 \end{bmatrix} + \begin{bmatrix} -6 \end{bmatrix} = -135. \quad (1.5)$$

Визначимо активацію вихідного шару нейронної мережі, що і є відповіддю усієї мережі на набір прикладів:

$$\hat{y} = A^{[2]} = \sigma(Z^{[2]}) = \sigma(-135) = 0. \quad (1.6)$$

**Завдання 1.3**

**Постановка завдання**

Виконати оновлення вагових коефіцієнтів нейронної мережі за методом зворотного поширення помилки, використовуючи вихідні дані за варіантом. Обчислити функцію вартості до та після оновлення вагових коефіцієнтів та порівняти ці значення. Заокруглення виконувати до двох знаків після коми.

Архітектуру нейронної мережі наведено на рисунку 1.5.

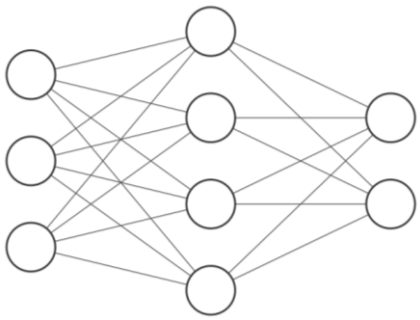

Рисунок 1.5 — Архітектура НМ для завдання 1.3

**Вихідні дані**

Варіанти вихідних даних до завдання доступні за посиланням [9].

**Приклад виконання завдання**

Приклад є спрощеною версією завдання; завдання за варіантами передбачає більший обсяг розрахунків.

Архітектуру нейронної мережі наведено на рисунку 1.6.

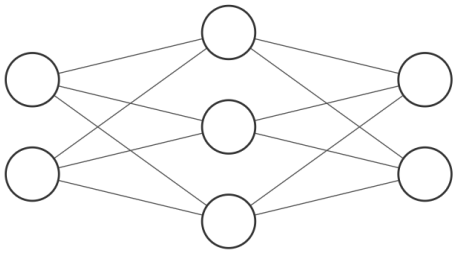

Рисунок 1.6 — Архітектура НМ для прикладу до завдання 1.3

Вихідні дані:

– матриця вхідних прикладів:

$$X = \begin{bmatrix} -3 \\ 1 \end{bmatrix};$$

– вектор правильних відповідей:

$$y = \begin{bmatrix} -2 \\ 4 \end{bmatrix};$$

– матриця вагових коефіцієнтів прихованого шару:

$$W^{[1]} = \begin{bmatrix} 1 & 2 \\ -1 & -2 \\ -3 & 1 \end{bmatrix};$$

– вектор зміщень прихованого шару:

$$b^{[1]} = \begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix};$$

– матриця вагових коефіцієнтів вихідного шару:

$$W^{[2]} = \begin{bmatrix} -2 & -2 & 2 \\ 1 & -1 & -3 \end{bmatrix};$$

– вектор зміщень вихідного шару:

$$b^{[2]} = \begin{bmatrix} 2 \\ -1 \end{bmatrix};$$

– активаційна функція прихованого шару:

$$\varphi^{[1]}(z) = \sigma(z);$$

– активаційна функція вихідного шару:

$$\varphi^{[2]}(z) = \sigma(z);$$

– функція втрат:

$$L(\hat{y}, y) = \frac{1}{2}(y - \hat{y})^2;$$

– коефіцієнт швидкості навчання:

$$\alpha = 0.2.$$

Для оновлення вагових коефіцієнтів за методом зворотного поширення помилки насамперед необхідно виконати обчислення прямого поширення.

Обчислення прямого поширення. Визначимо значення суматорної функції першого шару нейронної мережі:

$$Z^{[1]} = W^{[1]}X + b^{[1]} = \begin{bmatrix} 1 & 2 \\ -1 & -2 \\ -3 & 1 \end{bmatrix} \cdot \begin{bmatrix} -3 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 10 \end{bmatrix} + \begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 13 \end{bmatrix}. \quad (1.7)$$

Визначимо активацію першого шару нейронної мережі:

$$A^{[1]} = \sigma(Z^{[1]}) = \sigma\left(\begin{bmatrix} 0 \\ 0 \\ 13 \end{bmatrix}\right) = \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \end{bmatrix}. \quad (1.8)$$

Визначимо значення суматорної функції вихідного шару нейронної мережі:

$$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]} = \begin{bmatrix} -2 & -2 & 2 \\ 1 & -1 & -3 \end{bmatrix} \cdot \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \end{bmatrix} + \begin{bmatrix} 2 \\ -1 \end{bmatrix} = \\ = \begin{bmatrix} 0 \\ -3 \end{bmatrix} + \begin{bmatrix} 2 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ -4 \end{bmatrix}. \tag{1.9}$$

Визначимо активацію вихідного шару нейронної мережі, що і є відповіддю усієї мережі на набір прикладів:

$$\hat{y} = A^{[2]} = \sigma(Z^{[2]}) = \sigma\left(\begin{bmatrix} 2 \\ -4 \end{bmatrix}\right) = \begin{bmatrix} 0.88 \\ 0.02 \end{bmatrix}. \tag{1.10}$$

Після обчислення прямого поширення визначимо значення функції вартості. Зазначимо, що матриця прикладів $X$ містить лише один стовпець, тобто один приклад, а отже $m = 1$.

$$J = \frac{1}{m}\sum_{i,j} L(\hat{y}, y) = \frac{1}{1} \cdot \frac{1}{2}\sum_{i,j}\left(\begin{bmatrix} -2 \\ 4 \end{bmatrix} - \begin{bmatrix} 0.88 \\ 0.02 \end{bmatrix}\right)^2 = \frac{1}{2}\left(\begin{bmatrix} -2.88 \\ 3.98 \end{bmatrix}\right)^2 = \\ = \frac{1}{2}\sum_{i,j}\left(\begin{bmatrix} 8.29 \\ 15.84 \end{bmatrix}\right) = \frac{1}{2} \cdot 24.13 = 12.07. \tag{1.11}$$

Обчислення зворотного поширення.

$$dZ^{[2]} = L'(\hat{y}, y) \circ \varphi^{[2]\prime}(Z^{[2]}) = -(y - \hat{y}) \circ \sigma(Z^{[2]}) \circ (1 - \sigma(Z^{[2]})) =$$
$$= -\left(\begin{bmatrix} -2 \\ 4 \end{bmatrix} - \begin{bmatrix} 0.88 \\ 0.02 \end{bmatrix}\right) \circ \sigma\left(\begin{bmatrix} 2 \\ -4 \end{bmatrix}\right) \circ \left(1 - \sigma\left(\begin{bmatrix} 2 \\ -4 \end{bmatrix}\right)\right) =$$
$$= -\begin{bmatrix} -2.88 \\ 3.98 \end{bmatrix} \circ \begin{bmatrix} 0.88 \\ 0.02 \end{bmatrix} \circ \left(1 - \begin{bmatrix} 0.88 \\ 0.02 \end{bmatrix}\right) = \begin{bmatrix} 2.88 \\ -3.98 \end{bmatrix} \circ \begin{bmatrix} 0.88 \\ 0.02 \end{bmatrix} \circ \begin{bmatrix} 0.12 \\ 0.98 \end{bmatrix} = \quad (1.12)$$
$$= \begin{bmatrix} 0.3 \\ -0.08 \end{bmatrix}.$$

$$dW^{[2]} = \frac{1}{m} dZ^{[2]} \cdot (A^{[1]})^T = \frac{1}{1} \begin{bmatrix} 0.3 \\ -0.08 \end{bmatrix} \cdot \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \end{bmatrix}^T = \quad (1.13)$$
$$= \begin{bmatrix} 0.3 \\ -0.08 \end{bmatrix} \cdot \begin{bmatrix} 0.5 & 0.5 & 1 \end{bmatrix} = \begin{bmatrix} 0.15 & 0.15 & 0.3 \\ -0.04 & -0.04 & -0.08 \end{bmatrix}.$$

$$db^{[2]} = \frac{1}{m} \sum_{i=1}^{m} dZ^{[2]} = \begin{bmatrix} 0.3 \\ -0.08 \end{bmatrix}. \quad (1.14)$$

$$dZ^{[1]} = (W^{[2]})^T \cdot dZ^{[2]} \circ \varphi^{[1]\prime}(Z^{[1]}) =$$
$$= (W^{[2]})^T \cdot dZ^{[2]} \circ \sigma(Z^{[1]}) \circ (1 - \sigma(Z^{[1]})) =$$
$$= \begin{bmatrix} -2 & -2 & 2 \\ 1 & -1 & -3 \end{bmatrix}^T \cdot \begin{bmatrix} 0.3 \\ -0.08 \end{bmatrix} \circ \sigma\left(\begin{bmatrix} 0 \\ 0 \\ 13 \end{bmatrix}\right) \circ \left(1 - \sigma\left(\begin{bmatrix} 0 \\ 0 \\ 13 \end{bmatrix}\right)\right) =$$
$$= \begin{bmatrix} -2 & 1 \\ -2 & -1 \\ 2 & -3 \end{bmatrix} \cdot \begin{bmatrix} 0.3 \\ -0.08 \end{bmatrix} \circ \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \end{bmatrix} \circ \left(1 - \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \end{bmatrix}\right) = \quad (1.15)$$
$$= \begin{bmatrix} -0.68 \\ -0.52 \\ 0.84 \end{bmatrix} \circ \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0.5 \\ 0.5 \\ 0 \end{bmatrix} = \begin{bmatrix} -0.17 \\ -0.13 \\ 0 \end{bmatrix}.$$

$$dW^{[1]} = \frac{1}{m} dZ^{[1]} \cdot (A^{[0]})^T = \frac{1}{m} dZ^{[1]} \cdot X^T = \frac{1}{1} \begin{bmatrix} -0.17 \\ -0.13 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} -3 \\ 1 \end{bmatrix}^T =$$
$$= \begin{bmatrix} -0.17 \\ -0.13 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} -3 & 1 \end{bmatrix} = \begin{bmatrix} 0.51 & -0.17 \\ 0.39 & -0.13 \\ 0 & 0 \end{bmatrix}. \quad (1.16)$$

$$db^{[1]} = \frac{1}{m} \sum_{i=1}^{m} dZ^{[1]} = \frac{1}{1} \begin{bmatrix} -0.17 \\ -0.13 \\ 0 \end{bmatrix}. \quad (1.17)$$

Здійснимо корегування вагових коефіцієнтів:

$$W^{[2]} := W^{[2]} - \alpha \cdot dW^{[2]} =$$
$$= \begin{bmatrix} -2 & -2 & 2 \\ 1 & -1 & -3 \end{bmatrix} - 0.2 \cdot \begin{bmatrix} 0.15 & 0.15 & 0.3 \\ -0.04 & -0.04 & -0.08 \end{bmatrix} =$$
$$= \begin{bmatrix} -2 & -2 & 2 \\ 1 & -1 & -3 \end{bmatrix} - \begin{bmatrix} 0.03 & 0.03 & 0.06 \\ -0.01 & -0.01 & -0.02 \end{bmatrix} = \quad (1.18)$$
$$= \begin{bmatrix} -2.03 & -2.03 & 1.94 \\ 1.01 & -0.99 & -2.98 \end{bmatrix};$$

$$b^{[2]} := b^{[2]} - \alpha \cdot db^{[2]} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} - 0.2 \cdot \begin{bmatrix} 0.3 \\ -0.08 \end{bmatrix} =$$
$$= \begin{bmatrix} 2 \\ -1 \end{bmatrix} - \begin{bmatrix} 0.06 \\ -0.02 \end{bmatrix} = \begin{bmatrix} 1.94 \\ -0.98 \end{bmatrix}; \quad (1.19)$$

$$W^{[1]} := W^{[1]} - \alpha \cdot dW^{[1]} = \begin{bmatrix} 1 & 2 \\ -1 & -2 \\ -3 & 1 \end{bmatrix} - 0.2 \cdot \begin{bmatrix} 0.51 & -0.17 \\ 0.39 & -0.13 \\ 0 & 0 \end{bmatrix} =$$
$$= \begin{bmatrix} 1 & 2 \\ -1 & -2 \\ -3 & 1 \end{bmatrix} - \begin{bmatrix} 0.1 & -0.03 \\ 0.08 & -0.03 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0.9 & 2.03 \\ -1.08 & -1.97 \\ -3 & 1 \end{bmatrix}; \quad (1.20)$$

$$b^{[1]} := b^{[1]} - \alpha \cdot db^{[1]} = \begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix} - 0.2 \cdot \begin{bmatrix} -0.17 \\ -0.13 \\ 0 \end{bmatrix} =$$
$$= \begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix} - \begin{bmatrix} -0.03 \\ -0.03 \\ 0 \end{bmatrix} = \begin{bmatrix} 1.03 \\ -0.97 \\ 3 \end{bmatrix}. \quad (1.21)$$

Знову здійснимо пряме поширення, використовуючи оновлені вагові коефіцієнти. Визначимо значення суматорної функції першого шару нейронної мережі:

$$Z^{[1]} = W^{[1]} X + b^{[1]} = \begin{bmatrix} 0.9 & 2.03 \\ -1.08 & -1.97 \\ -3 & 1 \end{bmatrix} \cdot \begin{bmatrix} -3 \\ 1 \end{bmatrix} + \begin{bmatrix} 1.03 \\ -0.97 \\ 3 \end{bmatrix} =$$
$$= \begin{bmatrix} -0.67 \\ 1.27 \\ 10 \end{bmatrix} + \begin{bmatrix} 1.03 \\ -0.97 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.36 \\ 0.3 \\ 13 \end{bmatrix}. \quad (1.22)$$

Визначимо активацію першого шару нейронної мережі:

$$A^{[1]} = \sigma(Z^{[1]}) = \sigma\left(\begin{bmatrix} 0.36 \\ 0.3 \\ 13 \end{bmatrix}\right) = \begin{bmatrix} 0.59 \\ 0.57 \\ 1 \end{bmatrix}. \quad (1.23)$$

Визначимо значення суматорної функції вихідного шару нейронної мережі:

$$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]} = \begin{bmatrix} -2.03 & -2.03 & 1.94 \\ 1.01 & -0.99 & -2.98 \end{bmatrix} \cdot \begin{bmatrix} 0.59 \\ 0.57 \\ 1 \end{bmatrix} + \begin{bmatrix} 1.94 \\ -0.98 \end{bmatrix} = \\ = \begin{bmatrix} -0.41 \\ -2.95 \end{bmatrix} + \begin{bmatrix} 1.94 \\ -0.98 \end{bmatrix} = \begin{bmatrix} 1.53 \\ -3.93 \end{bmatrix}. \quad (1.24)$$

Визначимо активацію вихідного шару нейронної мережі, що і є відповіддю усієї мережі на набір прикладів:

$$\hat{y} = A^{[2]} = \sigma(Z^{[2]}) = \sigma\left(\begin{bmatrix} 1.53 \\ -3.93 \end{bmatrix}\right) = \begin{bmatrix} 0.82 \\ 0.02 \end{bmatrix}. \quad (1.25)$$

Після обчислення прямого поширення знову визначимо значення функції вартості.

$$J = \frac{1}{m}\sum_{i,j} L(\hat{y}, y) = \frac{1}{1} \cdot \frac{1}{2}\sum_{i,j}\left(\begin{bmatrix} -2 \\ 4 \end{bmatrix} - \begin{bmatrix} 0.82 \\ 0.02 \end{bmatrix}\right)^2 = \frac{1}{2}\left(\begin{bmatrix} -2.82 \\ 3.98 \end{bmatrix}\right)^2 = \\ = \frac{1}{2}\sum_{i,j}\left(\begin{bmatrix} 7.95 \\ 15.84 \end{bmatrix}\right) = \frac{1}{2} \cdot 23.79 = 11.9. \quad (1.26)$$

Як бачимо, після оновлення вагових коефіцієнтів значення функції вартості зменшилось — відбулось навчання.

**Завдання 1.4**

**Постановка завдання**

Використовуючи фреймворк Keras [3], побудувати та навчити повнозв'язну нейронну мережу прямого поширення. Мережа має містити два прихованих шари. Навчання здійснити на наборі даних «Boston Housing». Для оцінювання якості нейронної мережі, виділити 10 % прикладів для тестового набору даних. При оцінюванні використати у якості метрики середню абсолютну помилку (MAE). Інші параметри встановити використовуючи вихідні дані за варіантом. Побудувати графіки кривих навчання нейронної мережі.

**Вихідні дані**

Варіанти вихідних даних до завдання доступні за посиланням [10].

**Приклад виконання завдання**

Розглянемо нейронну мережу, що вирішує завдання двокласової класифікації на наборі даних «breast_cancer». Цей набір даних містить характеристики ядр клітин, зібраних із зображень біопсій грудей, і використовується для прогнозування, чи є пухлина злоякісною або доброякісною. Як метрику використаємо точність.

Вихідні дані:

– кількість нейронів у першому прихованому шарі: 300;
– кількість нейронів у другому прихованому шарі: 200;
– кількість нейронів у третьому прихованому шарі: 100;
– активаційна функція першого прихованого шару: ReLU;
– активаційна функція другого прихованого шару: ReLU;
– активаційна функція третього прихованого шару: ReLU;
– оптимізатор: Adagrad;

– кількість епох навчання: 75;

– розмір пакету прикладів: 100.

Програмний код:

```
# Імпортування необхідних бібліотек
import matplotlib.pyplot as plt
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from keras.models import Sequential
from keras.layers import Dense

# Завантаження та підготовка даних
data = load_breast_cancer()
X = data.data
y = data.target

# Розділення даних на тренувальний та тестовий набори
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# Нормалізація даних
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Створення моделі
model = Sequential()

# Додавання шарів до моделі
# Перший прихований шар
model.add(Dense(300, input_shape=(X.shape[1],), activation="relu"))
# Другий прихований шар
model.add(Dense(200, activation="relu"))
# Третій прихований шар
model.add(Dense(100, activation="relu"))
# Вихідний шар
model.add(Dense(1, activation="sigmoid"))

# Компіляція моделі
model.compile(
    optimizer="adagrad",
    loss="binary_crossentropy",
    metrics=["accuracy"],
)
```

```python
# Навчання моделі
history = model.fit(
    X_train,
    y_train,
    epochs=75,
    batch_size=100,
    validation_split=0.1,
    verbose=1,
)

# Оцінка моделі на тестових даних
test_loss, test_acc = model.evaluate(X_test, y_test)

print(
    f"Значення функції вартості на тестових даних: {test_loss:.3f}, "
    f"Значення точності на тестових даних: {test_acc:.3%}"
)

# Побудова графіків точності
plt.plot(history.history["accuracy"], label="Навчальні дані")
plt.plot(
    history.history["val_accuracy"],
    label="Валідаційні дані",
    linestyle="--",
)
plt.title("Графік точності")
plt.xlabel("Епоха")
plt.ylabel("Точність")
plt.legend()
plt.show()

# Побудова графіків функції вартості
plt.plot(history.history["loss"], label="Навчальні дані")
plt.plot(
    history.history["val_loss"],
    label="Валідаційні дані",
    linestyle="--",
)
plt.title("Графік функції вартості")
plt.xlabel("Епоха")
plt.ylabel("Функція вартості")
plt.legend()
plt.show()
```

На рисунку 1.7 наведено криві навчання нейронної мережі.

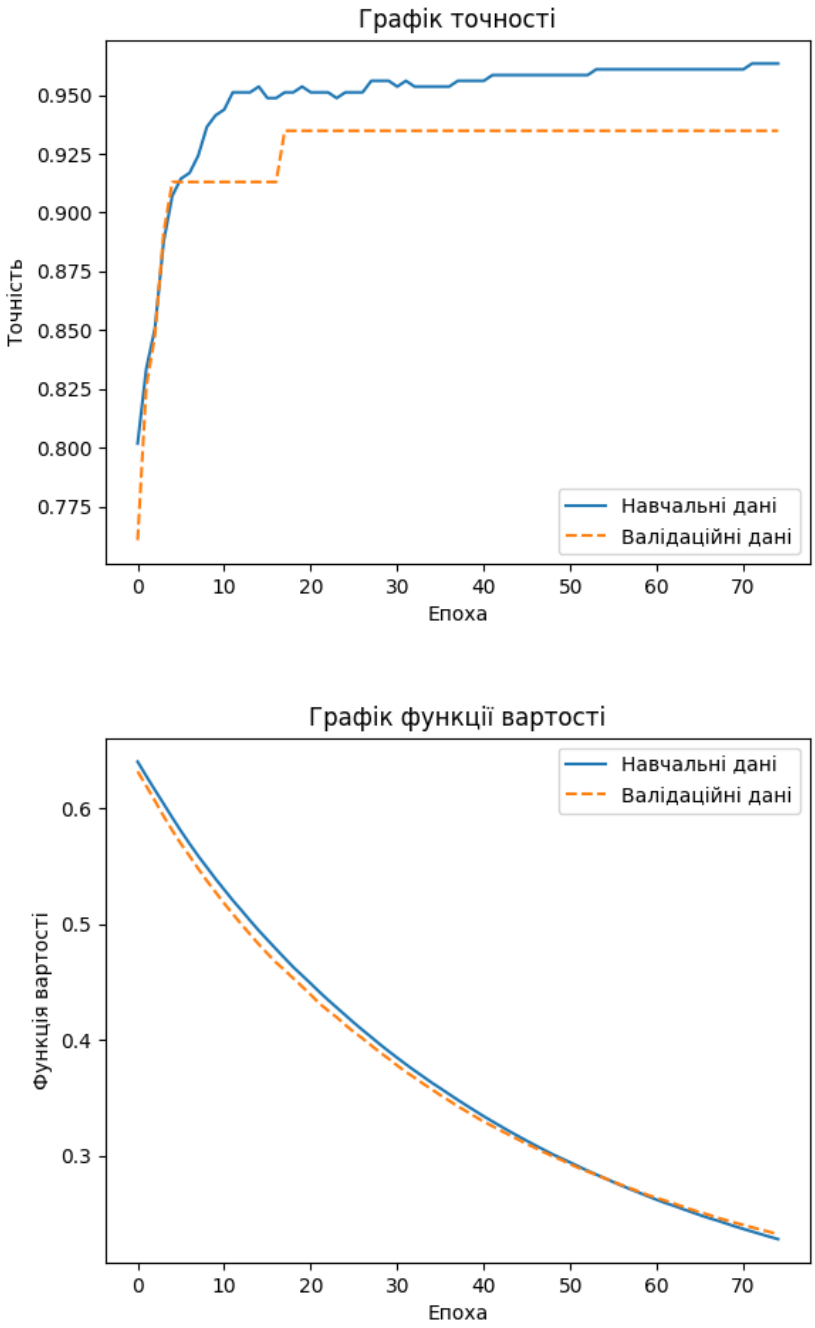


Рисунок 1.7 — Криві навчання нейронної мережі

Результати нейронної мережі на тестових даних:

– значення функції вартості: 0.254;

– значення точності: 95.614 %.

**Зміст звіту**

1 Титульний аркуш.

2 Мета роботи.

3 Вирішення завдання 1.1: вихідні дані за варіантом, розрахунки.

4 Вирішення завдання 1.2: вихідні дані за варіантом, розрахунки.

5 Вирішення завдання 1.3: вихідні дані за варіантом, розрахунки.

6 Вирішення завдання 1.4: вихідні дані за варіантом, програмний код і графіки кривих навчання.

7 Короткі висновки з роботи.

**Запитання для самоконтролю**

1 Що таке повнозв’язна нейронна мережа прямого поширення і які основні принципи її роботи?

2 З яких елементів архітектури складаються повнозв’язні нейронні мережі?

3 Які активаційні функції використовуються у повнозв’язних нейронних мережах?

4 Як повнозв’язні нейронні мережі застосовуються для вирішення задач класифікації та регресії, і в чому основна різниця між цими двома типами задач?

5 Які є етапи процесу навчання повнозв’язної нейронної мережі?

6 Як працює метод зворотного поширення помилки?

7 Які функції втрат використовуються при навчанні повнозв’язних нейронних мереж?

8 Які метрики використовуються для оцінювання якості повнозв’язних нейронних мереж?

9 Які кроки потрібно виконати для побудови та навчання повнозв’язної нейронної мережі на прикладі набору даних «Boston Housing» за допомогою фреймворку Keras?

10 Як можна оцінити ефективність навчання повнозв’язної нейронної мережі?

## Лабораторна робота 2
## ДОСЛІДЖЕННЯ ТА ЗАСТОСУВАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

### Мета роботи

Дослідити архітектуру та принципи роботи згорткових нейронних мереж. Набути навички застосування згорткових нейронних мереж для вирішення завдань комп’ютерного зору.

### Порядок виконання роботи

1 За матеріалами лекцій вивчити теоретичні відомості.
2 Виконати практичні завдання.
3 Оформити звіт з лабораторної роботи.
4 Захистити роботу у викладача.

### Практичні завдання

### Завдання 2.1

### Постановка завдання

Виконати обчислення операції згортки, використовуючи вихідні дані за варіантом. Заокруглення виконувати до двох знаків після коми.

Архітектуру нейронної мережі наведено на рисунку 2.1.

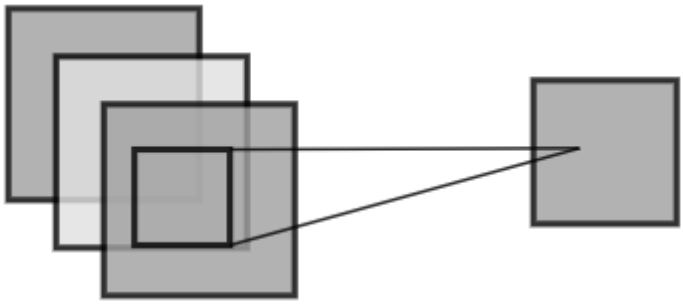

Рисунок 2.1 — Архітектура НМ для завдання 2.1

**Вихідні дані**

Варіанти вихідних даних до завдання доступні за посиланням [11].

**Приклад виконання завдання**

Приклад є спрощеною версією завдання; завдання за варіантами передбачає більший обсяг розрахунків.

Архітектуру нейронної мережі наведено на рисунку 2.2.

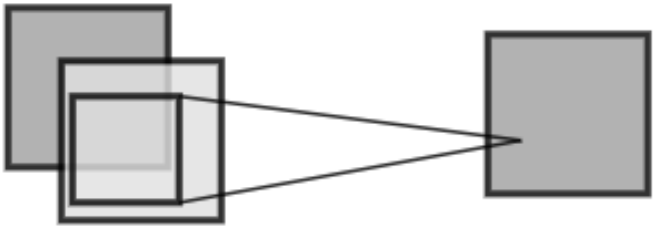

Рисунок 2.2 — Архітектура НМ для прикладу до завдання 2.1

Вихідні дані:

– матриця зображення:

$$X = \begin{bmatrix} \begin{bmatrix} 0.5 & 0 & 0.5 \\ 0.6 & 1 & 0.8 \\ 0.3 & 0.1 & 0 \end{bmatrix} \\ \begin{bmatrix} 0.9 & 0 & 0.1 \\ 0.3 & 0.5 & 0.3 \\ 0.5 & 0.4 & 0.8 \end{bmatrix} \end{bmatrix};$$

– матриця фільтра (вагові коефіцієнти):

$$W = \begin{bmatrix} \begin{bmatrix} -0.2 & -0.1 \\ -2.3 & -0.2 \end{bmatrix} \\ \begin{bmatrix} 1.7 & 0.9 \\ -2.3 & 0.5 \end{bmatrix} \end{bmatrix};$$

– значення зміщення:

$$b = -1.3.$$

Обчислення операції згортки.

$$Z = X \times W + b = \begin{bmatrix} \begin{bmatrix} 0.5 & 0 & 0.5 \\ 0.6 & 1 & 0.8 \\ 0.3 & 0.1 & 0 \end{bmatrix} \\ \begin{bmatrix} 0.9 & 0 & 0.1 \\ 0.3 & 0.5 & 0.3 \\ 0.5 & 0.4 & 0.8 \end{bmatrix} \end{bmatrix} \times \begin{bmatrix} \begin{bmatrix} -0.2 & -0.1 \\ -2.3 & -0.2 \end{bmatrix} \\ \begin{bmatrix} 1.7 & 0.9 \\ -2.3 & 0.5 \end{bmatrix} \end{bmatrix} + (-1.3) =$$

$$= \begin{bmatrix} 0.5 \cdot (-0.2) + 0 \cdot (-0.1) + & 0 \cdot (-0.2) + 0.5 \cdot (-0.1) + \\ +0.6 \cdot (-2.3) + 1 \cdot -0.2 & +1 \cdot (-2.3) + 0.8 \cdot (-0.2) \\ 0.6 \cdot (-0.2) + 1 \cdot (-0.1) + & 1 \cdot (-0.2) + 0.8 \cdot (-0.1) + \\ +0.3 \cdot (-2.3) + 0.1 \cdot -0.2 & +0.1 \cdot (-2.3) + 0 \cdot (-0.2) \end{bmatrix} +$$

$$+ \begin{bmatrix} 0.9 \cdot 1.7 + 0 \cdot 0.9 + & 0 \cdot 1.7 + 0.1 \cdot 0.9 + \\ +0.3 \cdot (-2.3) + 0.5 \cdot 0.5 & +0.5 \cdot (-2.3) + 0.3 \cdot 0.5 \\ 0.3 \cdot 1.7 + 0.5 \cdot 0.9 + & 0.5 \cdot 1.7 + 0.3 \cdot 0.9 + \\ +0.5 \cdot (-2.3) + 0.4 \cdot 0.5 & +0.4 \cdot (-2.3) + 0.8 \cdot 0.5 \end{bmatrix} + (-1.3) = \quad (2.1)$$

$$= \begin{bmatrix} -1.68 & -2.51 \\ -0.93 & -0.51 \end{bmatrix} + \begin{bmatrix} 1.09 & -0.91 \\ 0.01 & 0.6 \end{bmatrix} - 1.3 = \begin{bmatrix} -0.59 & -3.42 \\ -0.92 & 0.09 \end{bmatrix} - 1.3 =$$

$$= \begin{bmatrix} -1.89 & -4.72 \\ -2.22 & -1.21 \end{bmatrix}.$$

**Завдання 2.2**

**Постановка завдання**

Виконати обчислення вектора розмірності вихідної мапи ознак згорткової нейронної мережі, використовуючи вихідні дані за варіантом. Заокруглення виконувати до двох знаків після коми.

Архітектуру нейронної мережі наведено на рисунку 2.3.

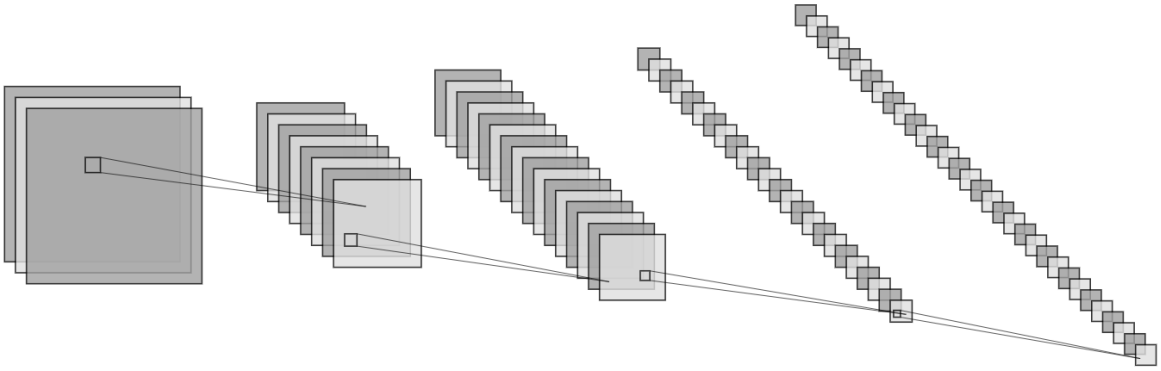

Рисунок 2.3 — Архітектура НМ для завдання 2.2

**Вихідні дані**

Варіанти вихідних даних до завдання доступні за посиланням [12].

**Приклад виконання завдання**

Приклад є спрощеною версією завдання; завдання за варіантами передбачає більший обсяг розрахунків.

Архітектуру нейронної мережі наведено на рисунку 2.4.

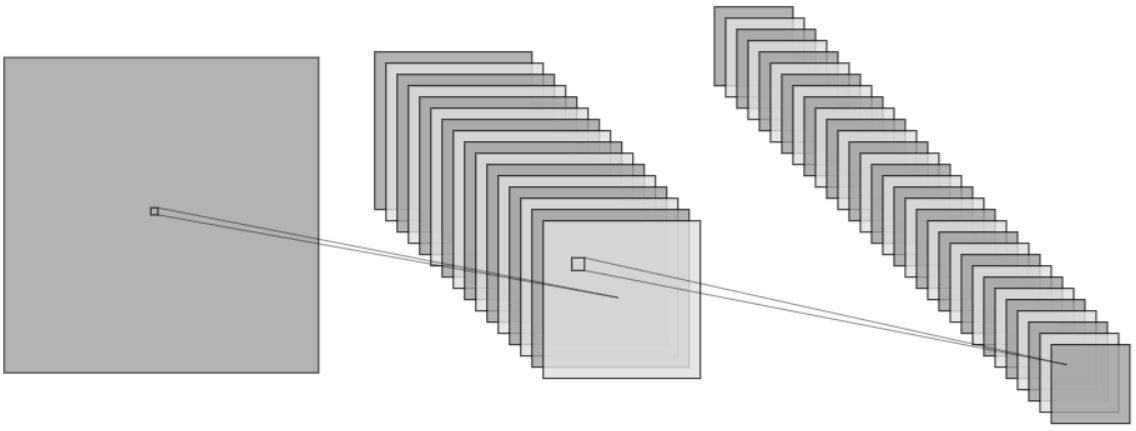

Рисунок 2.4 — Архітектура НМ для прикладу до завдання 2.2

Вихідні дані:

– розмірність вхідного зображення:

$$n_{img} = (224, 224, 1);$$

– розмірність фільтра першого згорткового шару:

$$n_f^{[1]} = (5, 5, 1);$$

– кількість фільтрів у першому згортковому шарі:

$$n_c^{[1]} = 16;$$

– значення падінгу в першому шарі:

$$p^{[1]} = 2;$$

– значення страйду в першому шарі:

$$s^{[1]} = 2;$$

– розмірність фільтра другого згорткового шару:

$$n_f^{[2]} = (9, 9, 16);$$

– кількість фільтрів у другому згортковому шарі:

$$n_c^{[2]} = 32;$$

– значення падінгу в другому шарі:

$$p^{[2]} = 4;$$

– значення страйду в другому шар:

$$s^{[2]} = 2.$$

Обчислення розмірностей.

Обчислимо перший компонент вектора розмірності мапи ознак першого згорткового шару:

$$n_{map,1}^{[1]} = \left\lfloor \frac{n_{img,1} + 2p^{[1]} - n_{f,1}^{[1]}}{s^{[1]}} \right\rfloor + 1 = \left\lfloor \frac{224 + 2 \cdot 2 - 5}{2} \right\rfloor + 1 = \left\lfloor 111.5 \right\rfloor + 1 = 111 + 1 = 112. \quad (2.2)$$

Вектор розмірності мапи ознак першого згорткового шару:

$$n_{map}^{[1]} = (112, 112, 16). \quad (2.3)$$

Обчислимо перший компонент вектора розмірності мапи ознак другого згорткового шару:

$$n_{map,1}^{[2]} = \left\lfloor \frac{n_{map,1}^{[1]} + 2p^{[2]} - n_{f,1}^{[2]}}{s^{[2]}} \right\rfloor + 1 = \left\lfloor \frac{112 + 2 \cdot 4 - 9}{2} \right\rfloor + 1 = \left\lfloor 55.5 \right\rfloor + 1 = 55 + 1 = 56. \quad (2.4)$$

Вектор розмірності мапи ознак другого згорткового шару:

$$n_{map}^{[2]} = (56, 56, 32). \quad (2.5)$$

**Завдання 2.3**

**Постановка завдання**

Використовуючи фреймворк Keras [3], побудувати та навчити згорткову нейронну мережу. Мережа має містити два згорткових шари, один шар пулінгу з параметром розмірності 2 та один повнозв'язний шар. Навчання здійснити на наборі даних «CIFAR-10». Для оцінювання якості нейронної мережі, виділити 10 % прикладів для тестового набору даних. При оцінюванні використати у якості метрики точність (accuracy). Інші параметри встановити використовуючи вихідні дані за варіантом. Побудувати графіки кривих навчання нейронної мережі.

**Вихідні дані**

Варіанти вихідних даних до завдання доступні за посиланням [13].

**Приклад виконання завдання**

Розглянемо нейронну мережу, що вирішує завдання класифікації зображень з набору даних MNIST. MNIST містить зображення 10 класів.

Вихідні дані:

– активаційна функція для згорткових шарів: ReLU;
– кількість фільтрів у першому згортковому шарі: 16;
– розмірність фільтрів у першому згортковому шарі: 3×3;
– кількість фільтрів у другому згортковому шарі: 32;
– розмірність фільтрів у другому згортковому шарі: 3×3;
– тип пулінгу: MaxPooling;
– кількість нейронів у повнозв'язному шарі: 128;
– оптимізатор: Adam;
– кількість епох навчання: 5;
– розмір пакету прикладів: 1024.

Програмний код:

```
# Імпорт необхідних бібліотек
import matplotlib.pyplot as plt
from keras.datasets import mnist
from keras.layers import Conv2D, Dense, Flatten, MaxPooling2D
from keras.models import Sequential
from keras.utils import to_categorical

# Завантаження набору даних MNIST
(X_train, y_train), (X_test, y_test) = mnist.load_data()

# Попередня обробка даних

# Змінюємо форму вхідних даних для того, щоб вони відповідали формату,
# який очікує Keras: кількість зразків x висота зображення x ширина
# зображення x кількість каналів
X_train = X_train.reshape(X_train.shape[0], 28, 28, 1)
X_test = X_test.reshape(X_test.shape[0], 28, 28, 1)

# Нормалізація значень пікселів до діапазону 0-1
X_train = X_train.astype("float32") / 255
X_test = X_test.astype("float32") / 255

# Кодування міток у форматі one-hot
y_train = to_categorical(y_train, 10)
y_test = to_categorical(y_test, 10)

# Побудова моделі згорткової нейронної мережі

model = Sequential()

# Додавання згорткового шару з 16 фільтрами розміром 3x3
# Використовуємо активаційну функцію ReLU для введення нелінійності
model.add(
    Conv2D(16, (3, 3), activation="relu", input_shape=(28, 28, 1))
)

# Другий згортковий шар
model.add(Conv2D(32, (3, 3), activation="relu"))

# Шар пулінгу для зменшення розмірності вхідного зображення
model.add(MaxPooling2D(pool_size=(2, 2)))

# Розгортка даних для входу у повнозв'язний шар
model.add(Flatten())

# Повнозв'язний шар з 128 нейронами і активаційною функцією ReLU
model.add(Dense(128, activation="relu"))
```

```python
# Вихідний шар з 10 нейронами (кількість класів) і активаційною
# функцією softmax
# для класифікації
model.add(Dense(10, activation="softmax"))

# Компіляція моделі з оптимізатором Adam і функцією втрат
# categorical_crossentropy
model.compile(
    optimizer="adam",
    loss="categorical_crossentropy",
    metrics=["accuracy"],
)

# Навчання моделі
# Використовуємо 10 % даних як валідаційний набір
history = model.fit(
    X_train,
    y_train,
    epochs=5,
    batch_size=1024,
    validation_split=0.1,
    verbose=1,
)

# Оцінка моделі на тестових даних
test_loss, test_acc = model.evaluate(X_test, y_test)

print(
    f"Значення функції вартості на тестових даних: {test_loss:.3f}, "
    f"Значення точності на тестових даних: {test_acc:.3%}"
)

# Виведення графіків точності
plt.plot(history.history["accuracy"], label="Навчальні дані")
plt.plot(
    history.history["val_accuracy"],
    label="Валідаційні дані",
    linestyle="--",
)
plt.title("Графік точності")
plt.xlabel("Епоха")
plt.ylabel("Точність")
plt.legend()
plt.show()

# Виведення графіків функції вартості
plt.plot(history.history["loss"], label="Навчальні дані")
plt.plot(
    history.history["val_loss"],
    label="Валідаційні дані",
```

```
    linestyle="--",
)
plt.title("Графік функції вартості")
plt.xlabel("Епоха")
plt.ylabel("Функція вартості")
plt.legend()
plt.show()
```

На рисунку 2.5 наведено криві навчання нейронної мережі.

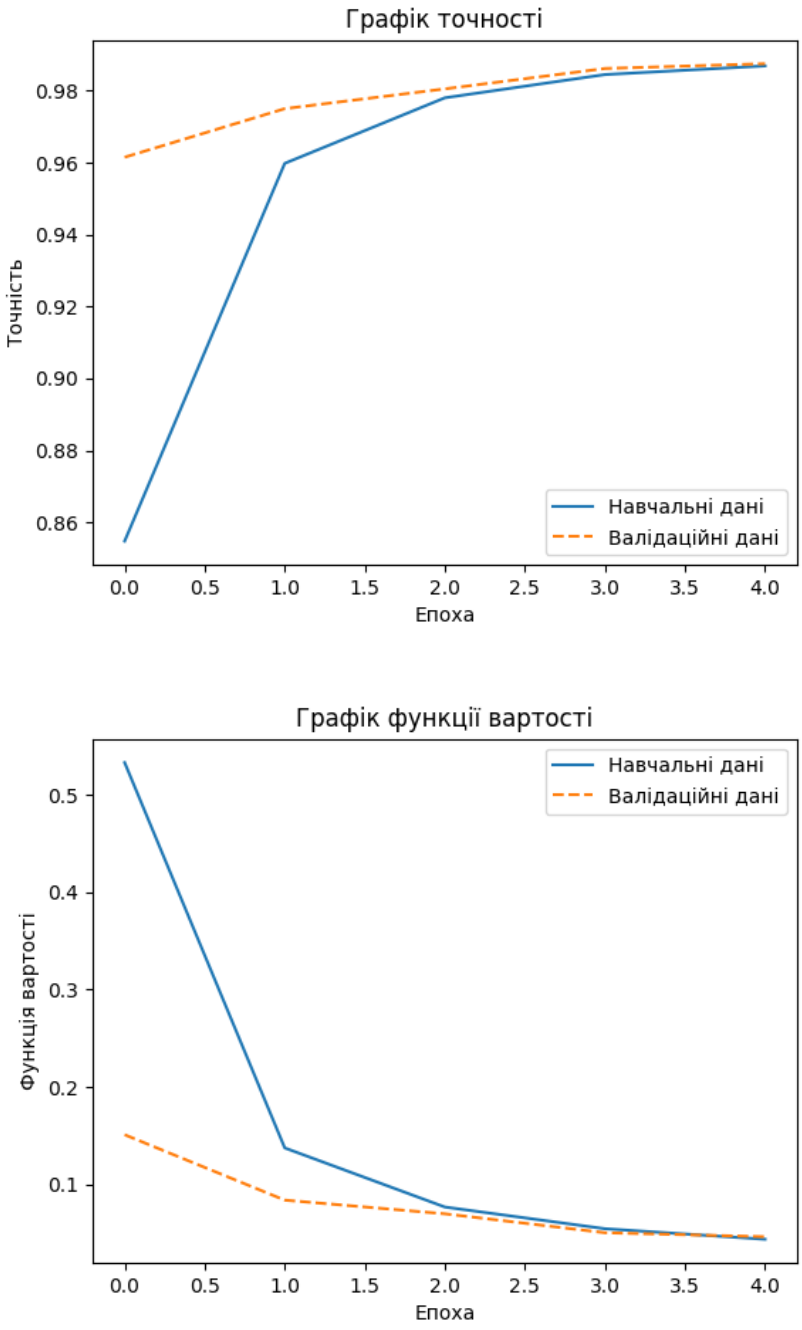


Рисунок 2.5 — Криві навчання нейронної мережі

Результати нейронної мережі на тестових даних:

– значення функції вартості: 0.052;

– значення точності: 98.41 %.

**Зміст звіту**

1 Титульний аркуш.

2 Мета роботи.

3 Вирішення завдання 2.1: вихідні дані за варіантом, розрахунки.

4 Вирішення завдання 2.2: вихідні дані за варіантом, розрахунки.

5 Вирішення завдання 2.3: вихідні дані за варіантом, програмний код і графіки кривих навчання.

6 Короткі висновки з роботи.

**Запитання для самоконтролю**

1 Які етапи виконання операції згортки у контексті згорткових нейронних мереж?

2 Чим відрізняється операція згортки від операції обчислення суматорної функції у шарі повнозв’язної нейронній мережі?

3 Як впливають параметри фільтра (розмір ядра, страйд, падінг) на розмірність вихідної мапи ознак?

4 Які є переваги використання згорткових нейронних мереж порівняно з повнозв’язними нейронними мережами для обробки зображень?

5 Як пулінг впливає на вихідну мапу ознак у згортковій нейронній мережі?

6 Як обрати кількість фільтрів у згортковому шарі і як це впливає на модель?

7 Які фактори потрібно враховувати при побудові архітектури згорткової нейронної мережі?

8 На що впливає вибір функції активації у згорткових і повнозв’язних шарах згорткової нейронної мережі?

9 Які критерії важливі при виборі оптимізатора для тренування згорткової нейронної мережі?

10 Як аналізувати криві навчання під час тренування згорткової нейронної мережі і як це може допомогти в покращенні моделі?

## Лабораторна робота 3
## ДОСЛІДЖЕННЯ ТА ЗАСТОСУВАННЯ РЕКУРЕНТНИХ НЕЙРОННИХ МЕРЕЖ

### Мета роботи

Дослідити архітектуру та принципи роботи рекурентних нейронних мереж. Набути навички застосування рекурентних нейронних мереж для вирішення завдань обробки послідовних даних.

### Порядок виконання роботи

1 За матеріалами лекцій вивчити теоретичні відомості.
2 Виконати практичні завдання.
3 Оформити звіт з лабораторної роботи.
4 Захистити роботу у викладача.

### Практичні завдання

### Завдання 3.1

### Постановка завдання

Виконати обчислення прямого поширення в класичній рекурентній нейронній мережі рівня символів, використовуючи вихідні дані за варіантом. Прихований стан ініціалізувати нулями. Заокруглення виконувати до двох знаків після коми.

Архітектуру нейронної мережі наведено на рисунку 3.1.

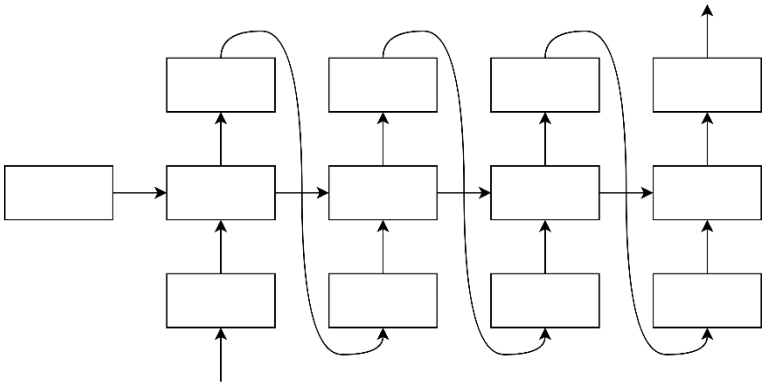

Рисунок 3.1 — Архітектура НМ для завдання 3.1

Словник для one-hot кодування:

$$V = \begin{bmatrix} \text{'a} & \text{'d'} & \text{'e'} & \text{'h'} & \text{'i'} & \text{'n'} & \text{'o'} & \text{'r'} & \text{'s'} & \text{'t'} \end{bmatrix}^T.$$

**Вихідні дані**

Варіанти вихідних даних до завдання доступні за посиланням [14].

**Приклад виконання завдання**

Приклад є спрощеною версією завдання; завдання за варіантами передбачає більший обсяг розрахунків.

Архітектуру нейронної мережі наведено на рисунку 3.2.

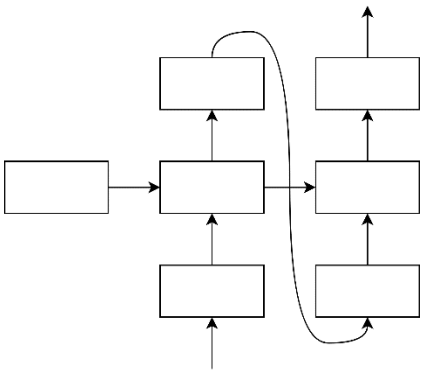

Рисунок 3.2 — Архітектура НМ для прикладу до завдання 3.1

Словник для one-hot кодування:

$$V = \begin{bmatrix} \text{'A'} & \text{'E'} & \text{'N'} & \text{'T'} \end{bmatrix}^T;$$

Вихідні дані:

– вхідне значення — перший елемент послідовності:

$$x^{\langle 1 \rangle} = \text{'N'};$$

– матриця вагових коефіцієнтів прихованого шару для перетворення вхідного значення:

$$W_{hx} = \begin{bmatrix} 0.6 & 0.4 & 0.8 & 0.2 \\ 0.5 & -1.4 & 1.7 & 0.2 \end{bmatrix};$$

– матриця вагових коефіцієнтів прихованого шару для перетворення попереднього прихованого стану:

$$W_{hh} = \begin{bmatrix} -1 & 0.3 \\ 0.3 & 1 \end{bmatrix};$$

– вектор зміщень прихованого шару:

$$b_h = \begin{bmatrix} 0.6 \\ -0.1 \end{bmatrix};$$

– матриця вагових коефіцієнтів вихідного шару:

$$W_{yh} = \begin{bmatrix} -1.9 & -0.1 \\ 0.2 & 0.9 \\ -1.6 & -0.1 \\ 0.4 & -1.3 \end{bmatrix};$$

– вектор зміщень вихідного шару:

$$b_y = \begin{bmatrix} -0.5 \\ 0.6 \\ -0.5 \\ 0.4 \end{bmatrix}.$$

Перетворимо вхідне значення на вектор за допомогою one-hot кодування. Отримаємо:

$$x^{\langle 1 \rangle} = \text{'N'} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}.$$

Значення прихованого стану в момент часу 0 ініціалізуємо нулями. З розміру матриць прихованого шару бачимо, що розмірність такого вектора має дорівнювати 2:

$$h^{\langle 0 \rangle} = \begin{bmatrix} 0 \\ 0 \end{bmatrix};$$

Обчислення прямого поширення.

Визначимо значення прихованого стану для моменту часу 1:

$$h^{\langle 1\rangle} = \tanh\left(W_{hx}x^{\langle 1\rangle} + W_{hh}h^{\langle 0\rangle} + b_h\right) =$$

$$= \tanh\left(\begin{bmatrix} 0.6 & 0.4 & 0.8 & 0.2 \\ 0.5 & -1.4 & 1.7 & 0.2 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0.3 \\ 0.3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0.6 \\ -0.1 \end{bmatrix}\right) = \quad (3.1)$$

$$= \tanh\left(\begin{bmatrix} 0.8 \\ 1.7 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0.6 \\ -0.1 \end{bmatrix}\right) = \tanh\left(\begin{bmatrix} 1.4 \\ 1.6 \end{bmatrix}\right) = \begin{bmatrix} 0.89 \\ 0.92 \end{bmatrix}.$$

Визначимо відповідь нейронної мережі для моменту часу 1:

$$y^{\langle 1\rangle} = \text{softmax}\left(W_{yh}h^{\langle 1\rangle} + b_y\right) =$$

$$= \text{softmax}\left(\begin{bmatrix} -1.9 & -0.1 \\ 0.2 & 0.9 \\ -1.6 & -0.1 \\ 0.4 & -1.3 \end{bmatrix} \cdot \begin{bmatrix} 0.89 \\ 0.92 \end{bmatrix} + \begin{bmatrix} -0.5 \\ 0.6 \\ -0.5 \\ 0.4 \end{bmatrix}\right) = \quad (3.2)$$

$$= \text{softmax}\left(\begin{bmatrix} -1.78 \\ 1.01 \\ -1.52 \\ -0.84 \end{bmatrix} + \begin{bmatrix} -0.5 \\ 0.6 \\ -0.5 \\ 0.4 \end{bmatrix}\right) = \text{softmax}\left(\begin{bmatrix} -2.28 \\ 1.61 \\ -2.02 \\ -0.44 \end{bmatrix}\right) = \begin{bmatrix} 0.02 \\ 0.85 \\ 0.02 \\ 0.11 \end{bmatrix}.$$

Обираючи значення з найбільшою ймовірністю, отримаємо відповідь нейронної мережі для моменту часу 1:

$$y^{\langle 1\rangle} = \begin{bmatrix} 0.02 \\ 0.85 \\ 0.02 \\ 0.11 \end{bmatrix} \rightarrow y^{\langle 1\rangle} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \text{'E'}. \quad (3.3)$$

Визначимо значення прихованого стану для моменту часу 2:

$$h^{\langle 2\rangle} = \tanh\left(W_{hx}x^{\langle 2\rangle} + W_{hh}h^{\langle 1\rangle} + b_h\right) =$$

$$= \tanh\left(\begin{bmatrix} 0.6 & 0.4 & 0.8 & 0.2 \\ 0.5 & -1.4 & 1.7 & 0.2 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \right.$$

$$\left. + \begin{bmatrix} -1 & 0.3 \\ 0.3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.89 \\ 0.92 \end{bmatrix} + \begin{bmatrix} 0.6 \\ -0.1 \end{bmatrix}\right) = \tag{3.4}$$

$$= \tanh\left(\begin{bmatrix} 0.4 \\ -1.4 \end{bmatrix} + \begin{bmatrix} -0.61 \\ 1.19 \end{bmatrix} + \begin{bmatrix} 0.6 \\ -0.1 \end{bmatrix}\right) = \tanh\left(\begin{bmatrix} 0.39 \\ -0.31 \end{bmatrix}\right) = \begin{bmatrix} 0.37 \\ -0.3 \end{bmatrix}.$$

Визначимо відповідь нейронної мережі для моменту часу 2:

$$y^{\langle 2\rangle} = \text{softmax}\left(W_{yh}h^{\langle 1\rangle} + b_y\right) =$$

$$= \text{softmax}\left(\begin{bmatrix} -1.9 & -0.1 \\ 0.2 & 0.9 \\ -1.6 & -0.1 \\ 0.4 & -1.3 \end{bmatrix} \cdot \begin{bmatrix} 0.37 \\ -0.3 \end{bmatrix} + \begin{bmatrix} -0.5 \\ 0.6 \\ -0.5 \\ 0.4 \end{bmatrix}\right) = \tag{3.5}$$

$$= \text{softmax}\left(\begin{bmatrix} -0.67 \\ -0.2 \\ -0.56 \\ 0.54 \end{bmatrix} + \begin{bmatrix} -0.5 \\ 0.6 \\ -0.5 \\ 0.4 \end{bmatrix}\right) = \text{softmax}\left(\begin{bmatrix} -1.17 \\ 0.4 \\ -1.06 \\ 0.94 \end{bmatrix}\right) = \begin{bmatrix} 0.07 \\ 0.32 \\ 0.07 \\ 0.54 \end{bmatrix}.$$

Обираючи значення з найбільшою ймовірністю, отримаємо відповідь нейронної мережі для моменту часу 2:

$$y^{\langle 2\rangle} = \begin{bmatrix} 0.07 \\ 0.32 \\ 0.07 \\ 0.54 \end{bmatrix} \rightarrow y^{\langle 2\rangle} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \text{'T'}. \tag{3.6}$$

Отже, рекурентна нейронна мережа згенерувала послідовність символів «NET».

### Завдання 3.2

#### Постановка завдання

Використовуючи фреймворк Keras [3], побудувати та навчити рекурентну нейронну мережу для аналізу сентименту текстових відгуків. Навчання здійснити на наборі даних IMDB, що містить відгуки на фільми з маркуванням «позитивний» або «негативний». Для оцінювання якості нейронної мережі використати 10 % прикладів для тестового набору даних. Як метрику використати точність (accuracy). Інші параметри встановити використовуючи вихідні дані за варіантом. Побудувати графіки кривих навчання нейронної мережі.

#### Вихідні дані

Варіанти вихідних даних до завдання доступні за посиланням [15].

#### Приклад виконання завдання

Розглянемо нейронну мережу, що вирішує завдання класифікації текстів за темами з набору даних Reuters.

Вихідні дані:

– розмір словника: 10000;
– максимальна довжина послідовності: 500;
– розмірність вектора вхідного шару Embedding: 32;
– кількість нейронів у рекурентному шарі: 32;
– оптимізатор: RMSprop;
– кількість епох навчання: 10;
– розмір пакету прикладів: 128.

Програмний код:

```
# Імпорт необхідних бібліотек
import matplotlib.pyplot as plt
from keras.datasets import reuters
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Embedding, SimpleRNN, Dense
from keras.utils import to_categorical

vocab_size = 10000
maxlen = 500

# Завантаження даних з набору даних Reuters
# Встановлюємо num_words=10000, щоб обмежити набір до 10,000 найбільш
# часто використовуваних слів
(X_train, y_train), (X_test, y_test) = reuters.load_data(
    num_words=10000, test_split=0.1
)

# Підготовка даних
# Обмежуємо довжину вхідних новин до 500 слів
X_train = pad_sequences(X_train, maxlen=maxlen)
X_test = pad_sequences(X_test, maxlen=maxlen)

# Конвертація векторів класів у бінарну матрицю класів
# Число класів
num_classes = max(y_train) + 1
y_train = to_categorical(y_train, num_classes)
y_test = to_categorical(y_test, num_classes)

# Створення моделі
model = Sequential()
# Вхідний шар Embedding
model.add(Embedding(vocab_size, 32))
# Рекурентний шар
model.add(SimpleRNN(32))
# Вихідний шар
model.add(Dense(num_classes, activation="softmax"))

# Компіляція моделі
model.compile(
    optimizer="rmsprop",
    loss="categorical_crossentropy",
    metrics=["accuracy"],
)

# Навчання моделі
history = model.fit(
    X_train,
```

```
    y_train,
    epochs=10,
    batch_size=128,
    validation_split=0.1,
    verbose=1,
)

# Оцінка моделі на тестових даних
test_loss, test_acc = model.evaluate(X_test, y_test)

print(
    f"Значення функції вартості на тестових даних: {test_loss:.3f}, "
    f"Значення точності на тестових даних: {test_acc:.3%}"
)

# Побудова графіків точності
plt.plot(history.history["accuracy"], label="Навчальні дані")
plt.plot(
    history.history["val_accuracy"],
    label="Валідаційні дані",
    linestyle="--",
)
plt.title("Графік точності")
plt.xlabel("Епоха")
plt.ylabel("Точність")
plt.legend()
plt.show()

# Побудова графіків функції вартості
plt.plot(history.history["loss"], label="Навчальні дані")
plt.plot(
    history.history["val_loss"],
    label="Валідаційні дані",
    linestyle="--",
)
plt.title("Графік функції вартості")
plt.xlabel("Епоха")
plt.ylabel("Функція вартості")
plt.legend()
plt.show()
```

На рисунку 3.3 наведено криві навчання нейронної мережі.

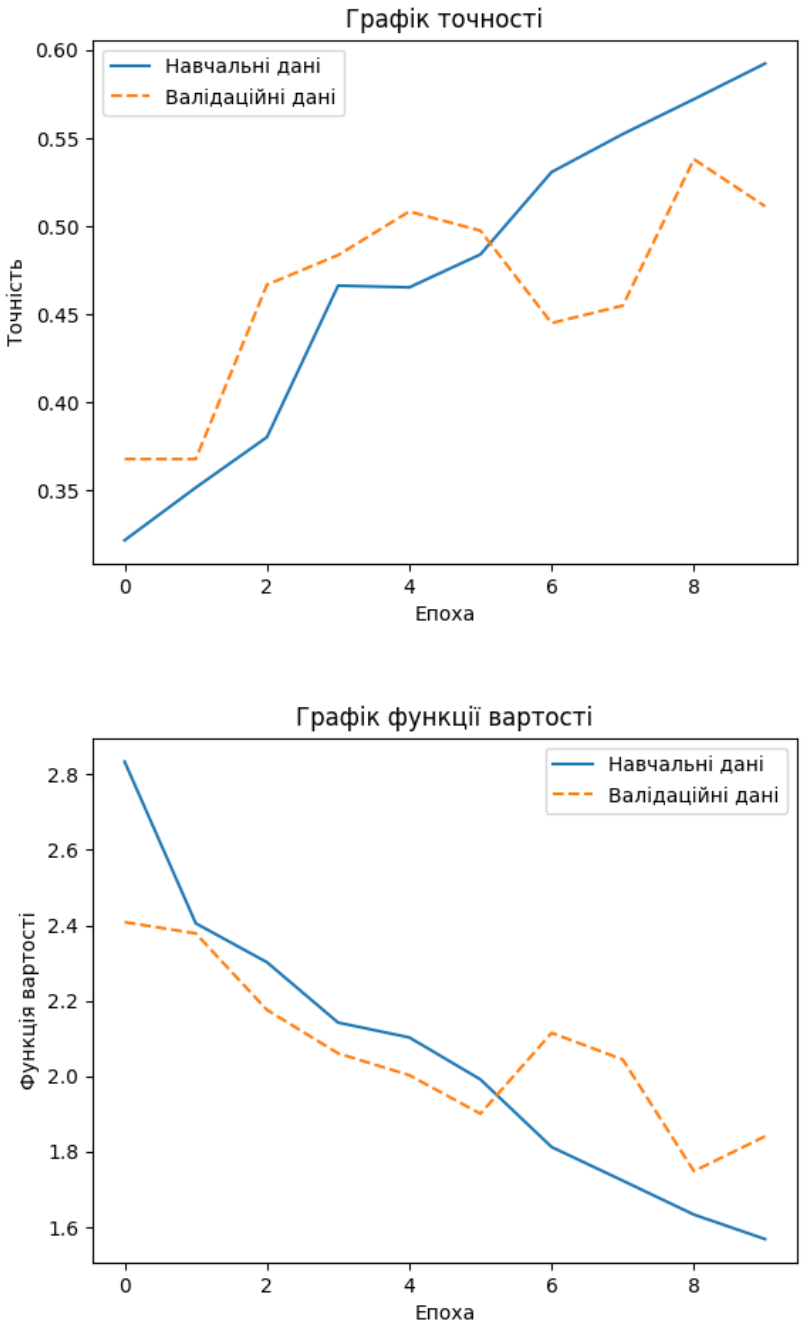


Рисунок 3.3 — Криві навчання нейронної мережі

Результати нейронної мережі на тестових даних:

– значення функції вартості: 2.173;

– значення точності: 48.353 %.

**Зміст звіту**

1 Титульний аркуш.

2 Мета роботи.

3 Вирішення завдання 3.1: вихідні дані за варіантом, розрахунки.

4 Вирішення завдання 3.2: вихідні дані за варіантом, програмний код та графіки кривих навчання.

5 Короткі висновки з роботи.

**Запитання для самоконтролю**

1 Що таке рекурентна нейронна мережа та в чому полягає її основна відмінність від згорткових та повнозв’язних нейронних мереж?

2 Які основні компоненти рекурентної нейронної мережі?

3 Яку роль відіграє прихований стан у рекурентних нейронних мережах?

4 Чим відрізняється обчислення прямого поширення в рекурентних нейронних мережах порівняно з іншими типами нейронних мереж?

5 Як виконується кодування вхідних даних для рекурентних нейронних мереж?

6 Які проблеми можуть виникнути під час тренування рекурентних нейронних мереж і як їх можна вирішити?

7 Як рекурентні нейронні мережі можуть застосовуватися для аналізу сентименту текстових відгуків?

8 Які особливості рекурентних нейронних мереж роблять їх придатними для обробки послідовних даних?

9 Які параметри та архітектурні рішення важливі при проєктуванні рекурентних нейронних мереж для конкретних завдань?

10 Як оцінювати ефективність рекурентних нейронних мереж і як інтерпретувати результати їхньої роботи?

# СПИСОК ЛІТЕРАТУРИ


1 Goodfellow I., Bengio Y., Courville A. Deep learning. MIT Press, 2016. 800 p. URL: http://www.deeplearningbook.org (дата звернення: 15.05.2024).

2 Google Colaboratory. *Google Colab*. URL: https://colab.research.google.com (дата звернення: 15.05.2024).

3 Keras. *Keras: Deep Learning for humans*. URL: https://keras.io (дата звернення: 15.05.2024).

4 Lenail A. NN SVG. Візуалізатор архітектури штучних нейронних мереж. *Alex Lenail*. URL: https://alexlenail.me/NN-SVG (дата звернення: 15.05.2024).

5 Nielsen M. Neural Networks and Deep Learning. Determination Press, 2015. URL: http://neuralnetworksanddeeplearning.com (дата звернення: 15.05.2024).

6 Басюк Т. М. та ін. Машинне навчання: навчальний посібник. Львів: Новий світ-2000, 2019. 329 с.

7 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 1, завдання 1 з дисципліни «Штучні нейронні мережі». *GitHub*. URL: https://github.com/oleksa-iv/ann-course/blob/master/lab_1_FCNN/lab_1.1_data.xlsx (дата звернення: 15.05.2024).

8 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 1, завдання 2 з дисципліни «Штучні нейронні мережі». *GitHub*. URL: https://github.com/oleksa-iv/ann-course/blob/master/lab_1_FCNN/lab_1.2_data.xlsx (дата звернення: 15.05.2024).

9 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 1, завдання 3 з дисципліни «Штучні нейронні мережі». *GitHub*. URL: https://github.com/oleksa-iv/ann-

course/blob/master/lab_1_FCNN/lab_1.3_data.xlsx (дата звернення: 15.05.2024).

10 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 1, завдання 4 з дисципліни «Штучні нейронні мережі». *GitHub*. URL: https://github.com/oleksa-iv/ann-course/blob/master/lab_1_FCNN/lab_1.4_data.xlsx (дата звернення: 15.05.2024).

11 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 2, завдання 1 з дисципліни «Штучні нейронні мережі». *GitHub*. URL: https://github.com/oleksa-iv/ann-course/blob/master/lab_2_CNN/lab_2.1_data.xlsx (дата звернення: 15.05.2024).

12 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 2, завдання 2 з дисципліни «Штучні нейронні мережі». *GitHub*. URL: https://github.com/oleksa-iv/ann-course/blob/master/lab_2_CNN/lab_2.2_data.xlsx (дата звернення: 15.05.2024).

13 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 2, завдання 3 з дисципліни «Штучні нейронні мережі». *GitHub*. URL: https://github.com/oleksa-iv/ann-course/blob/master/lab_2_CNN/lab_2.3_data.xlsx (дата звернення: 15.05.2024).

14 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 3, завдання 1 з дисципліни «Штучні нейронні мережі». *GitHub*. URL: https://github.com/oleksa-iv/ann-course/blob/master/lab_3_RNN/lab_3.1_data.xlsx (дата звернення: 15.05.2024).

15 Іванюк О. І. Варіанти вихідних даних до лабораторної роботи 3, завдання 2 з дисципліни «Штучні нейронні мережі». *GitHub*. URL:

https://github.com/oleksa-iv/ann-course/blob/master/lab_3_RNN/lab_3.2_data.xlsx (дата звернення: 15.05.2024).

16 Субботін С. О. Нейронні мережі: теорія та практика: навч. посіб. Житомир: О. О. Євенок, 2020. 184 с.

17 Ткаліченко С. В. Штучні нейронні мережі: навч посіб. Кривий Ріг : Держ. ун-т економіки і технологій, 2023. 150 с.

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт

з дисципліни

«*ШТУЧНІ НЕЙРОННІ МЕРЕЖІ*»

Відповідальний за випуск Іванюк О. І.

---

Підписано до друку 17.06.2024 р.
Умовн. друк. арк. 3,25. Тираж . Замовлення № .
Видавець та виготовлювач Український державний університет залізничного транспорту,
61050, Харків-50, майдан Фейєрбаха,7.
Свідоцтво суб'єкта видавничої справи ДК № 6100 від 21.03.2018 р.