УДК 004.032.26:004.352.246
DOI: 10.31866/2617-796X.5.1.2022.261305

Ткаченко Костянтин,
кандидат економічних наук,
доцент кафедри інформаційних технологій та дизайну,
Державний університет інфраструктури та технологій,
Київ, Україна
tkachenko.kostyantyn@gmail.com
https://orcid.org/0000-0003-0549-3396

Зуєнко Олексій,
бакалавр, кафедра інформаційних технологій та дизайну,
Державний університет інфраструктури та технологій,
Київ, Україна
a.zuenko.a@gmail.com
https://orcid.org/0000-0003-2890-9905

ВИКОРИСТАННЯ БАГАТОШАРОВОЇ LSTM-НЕЙРОМЕРЕЖІ В ПРОЦЕСІ РОЗПІЗНАВАННЯ ДРУКОВАНИХ ТЕКСТІВ

Метою статті є дослідження, аналіз і розгляд загальних проблем та перспектив щодо розробки систем розпізнавання друкованих текстів на основі використання нейронних мереж.

Методами дослідження є методи семантичного аналізу основних понять цієї предметної сфери (системи розпізнавання друкованих текстів). Розглянуто підходи до розробки та функціонування систем розпізнавання на основі нейромереж.

Новизною проведеного дослідження є розробка власного підходу до розпізнавання текстів на основі нейронних мереж, результати якого використано під час розробки власної системи розпізнавання друкованих текстів.

Висновки. У роботі розглянуто відомі погляди щодо розпізнавання образів на прикладі друкованих текстів, проаналізовано сучасні підходи з використанням нейромереж та їх навчанням. Ураховуючи результати проведеного аналізу, ухвалено рішення щодо розробки системи розпізнавання мов друкованих текстів з використанням нейромереж, що навчаються.

Ключові слова: нейронні мережі; навчання нейронних мереж; розпізнавання образів; система розпізнавання друкованих текстів.

Вступ. Усе більше й більше застосовують нейронні мережі (НМ) та нейромережеві технології під час розв'язання широкого спектра завдань у різних сферах, зокрема науці, суспільстві, економіці, виробництві тощо (Ставицький та Мозолевська, 2017; Бурлєєв, Василенко та Іваненко, 2021; Лещинський та Іщенко, 2017).

Проблемам навчання, розвитку та застосування НМ приділяють усе більше уваги. Зокрема, приділяють усе більше уваги опису перших принципів навчання нейронів (1949 р.), моделювання роботи перцептрона (1957 р.), основних етапів розвитку нейронних мереж, їх видів та класифікації (Субботін, Олійник А. та Олійник О., 2009).

На сьогодні все частіше використовують НМ у процесі розпізнавання образів (графічних, звукових, текстових тощо). Є багато різноманітних досліджень щодо розвитку та впровадження нейромереж і нейромережевих технологій у різноманітних сферах (Zheng, Meng and Jin, 2011; Bengio, 2009; Girshick, 2015). Зокрема, питанням використання НМ у теорії розпізнавання образів присвячені роботи багатьох сучасних фахівців у сфері штучного інтелекту (Алексеев и Квятковская, 2021; Олейник А., Зыбина и Олейник Е., 2020; Маркова и Жигалов, 2017). Тому актуальність проблеми застосування нейромереж у процесі розпізнавання друкованих текстів не викликає сумнівів.

Метою та основними завданнями статті є дослідження використання НМ під час розпізнавання друкованих текстів й опис програми розпізнавання мови друкованих текстів за допомогою НМ, що навчається.

Результати дослідження. У статті «Що таке рекуррентні нейронні мережі» («What are Recurrent Neural Networks») описано сутність НМ RNN (Recurrent neural network), в якій певна послідовність дій формується не тільки між шарами, а й у часі. Тобто в нейронів з'являється «пам'ять», інформацію з якої вони передають на наступний момент часу. Наприклад, використовуючи RNN під час аналізу відео, можна спрогнозувати наступний кадр (його номер та вміст). Структуру шару в RNN показано на рис. 1.

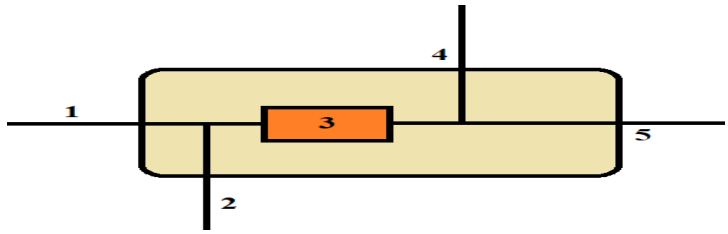

Рис. 1. Структура шару в RNN

На рис. 1 цифрові скорочення означають відповідно: 1 – вихідні дані з попереднього моменту часу h_{t-1} (якщо t = 0, то в ролі вихідних даних використовується нульовий вектор); 2 – вхідні дані L_0 або вихідні дані з попереднього шару L_{n-1}, де n – номер поточного шару; 3 – виконання операції множення вхідних даних на відповідні ваги та використання функції активації (як правило, для RNN – це гіпер-

болічний тангенс); 4 – передавання даних до наступного шару; 5 – передавання даних до наступного моменту часу.

Але RNN має проблему так званого «затухання градієнта» під час навчання. Цю проблему може розв'язати LSTM (Long short-temp memory) – нейрона мережа на основі архітектури RNN, але з різною структурою шарів (Gers, Schraudolph and Schmidhuber, 2002). Структура шару в LSTM (рис. 2) набагато складніша ніж в RNN.

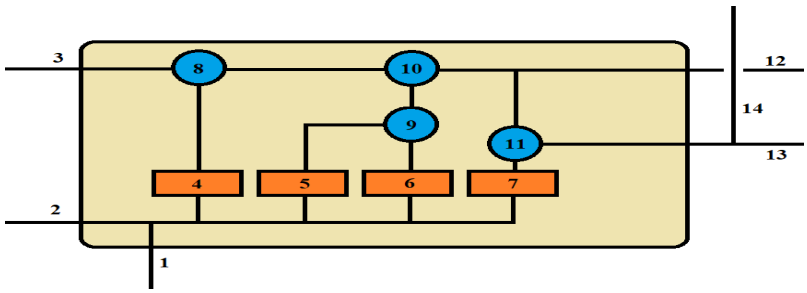

Рис. 2. Структура шару в LSTM

На рис. 2 цифрові скорочення означають відповідно: 1 – вихідні дані з попереднього моменту часу h(t-1), для початкового моменту часу (t = 0) підставляється нульовий вектор; 2 – вхідні дані L(0) або вихід з попереднього шару L(n-1); 3 – C(t-1), так званий Cell State («довгострокова пам'ять», яка передається до наступних проміжків часу, частково «забуває» попередню інформацію, але й додає нову), якщо t = 0, то ініціалізується C(0) з нульовими значеннями; 4 – Forget gate відповідає за часткове забуття попередньої інформації, має власні ваги для h- та L-даних, усе поєднує та активується сигмоїдальною функцією; 5 – Input gate відповідає за «додавання нової інформації», як і Forget gate, має власні ваги й активується сигмоїдальною функцією; 6 – Input modulation gate відповідає за додавання інформації, активується функцією гіперболічного тангенса; 7 – Output gate (за властивостями як 5 та 6) відповідає за вихідні дані, які передаються на наступний шар у наступний момент часу; 8 – застосування Forget gate до довгострокової пам'яті, перемножуючи матриці; 9 – злиття Input gate з модифікатором, перемножуючи матриці; 10 – злиття результату 8 з результатом 9 через додавання матриць; 11 – активація результату 10 гіперболічним тангенсом і злиття з Output gate, перемножуючи матриці; 12 – передача «оновленого» Cell State C(t) до наступного моменту часу; 13 – передача вихідного результату до наступного моменту часу; 14 – передача вихідного результату до наступного шару.

Крім базової LSTM (рис. 2), є й інші модифікації LSTM. Наприклад, коли Cell State бере участь у формуванні Forget gate. Хоча за допомогою LSTM вдалося позбутися «затухання градієнта», але проблема «вибуху» (коли градієнт стає нескінченним) залишається не до кінця розв'язаною. НМ LSTM дала суттєві покращення результату в процесі розв'язання багатьох завдань. Наприклад, точність розпізнавання картинок зросла на 50 %. Гугл за її допомогою покращив якість перекладання тексту так, що складно визначитися: зробила це НМ чи людина. LSTM

використовується в прогнозуванні, індустрії розваг і т. ін. (Goodfellow et al., 2014; Schmidhuber, 2015). Тобто LSTM стала однією з універсальних НМ. Є 2 види архітектури LSTM, різниця яких полягає в тому, що перша (рис. 3) повертає результат у кожний момент часу, а друга – лише в кінці.

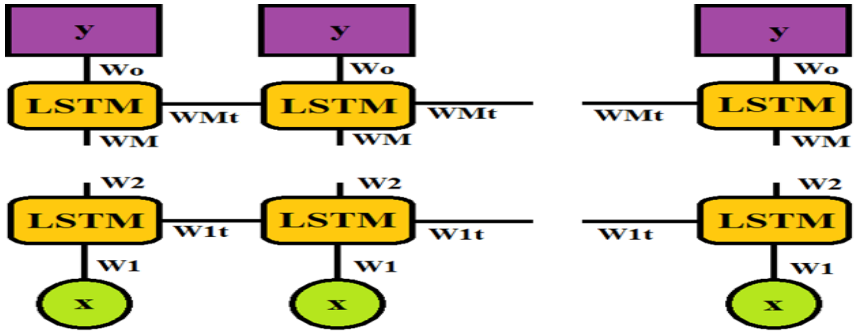

Рис. 3. Архітектура нейронної мережі LSTM

Навчання LSTM. Розглянемо процес навчання НМ LSTM, яка буде використовуватися для розпізнавання друкованих текстів, детально по основних її вузлах.

Для розгляду *Forward pass* введемо такі позначення: t – момент часу (пам'яті); n – номер поточного шару; N – кількість шарів часу (пам'яті); M – кількість шарів; u – результат, який має бути на виході; h(t) – вихідні дані для передачі в часі; L(n) – вихідні дані для передачі між шарами; C(t) – Cell State, який передається між шарами часу; η – швидкість навчання; f – Forget gate; i – Input gate; g – Input modify gate; o – Output gate; z – шар результату; y – шар результату, активований softmax-функцією.

Wx({f, i, g, o}, n) – ваги, які будуть застосовані до даних з попереднього шару; перший параметр ({f, i, g, o}) визначає, до якого входу належить, другий (n) – шар, до якого належить.

Wh({f, i, g, o}, n) – ваги, які будуть застосовані до даних з попереднього моменту часу; перший параметр ({f, i, g, o}) визначає, до якого входу належить, другий (n) – шар, до якого належить.

b({f, i, g, o}, n) – нейрони зміщення для кожного LSTM-блоку; V – ваги для знаходження вихідного шару; bV – нейрони зміщення для вихідного шару.

Функції активації, які будуть застосовані під час навчання LSTM:

1) tanh() – функція гіперболічного тангенса на відрізку [-1; 1]:

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

2) sigm() – сигмоїдальна функція на відрізку [0; 1]:

$$\mathrm{sigm}(x) = \frac{1}{1 + e^{-x}}$$

Будемо вважати, що всі вхідні, вихідні дані та ваги, нейрони зміщення, Cell State – це все представлено відповідними матрицями. Дія множення між ними (символ • у формулах) буде виконуватися за правилами матриць (але поелементно, тоді розмірності матриць мають бути однакові).

Символ + між матрицями означає дію поелементного додавання (розмірності матриць мають бути однакові).

Для знаходження LSTM-блоку слід виконати такі кроки:

1. Обчислити $\mathrm{f} = \mathrm{sigm}(\mathrm{L}(\mathrm{n}-1) * \mathrm{Wx}(\mathrm{f},\mathrm{n}) + \mathrm{h}(\mathrm{t}-1) * \mathrm{Wh}(\mathrm{f},\mathrm{n}) + \mathrm{b}(\mathrm{f},\mathrm{n}))$.
2. Обчислити $\mathrm{i} = \mathrm{sigm}(\mathrm{L}(\mathrm{n}-1) * \mathrm{Wx}(\mathrm{i},\mathrm{n}) + \mathrm{h}(\mathrm{t}-1) * \mathrm{Wh}(\mathrm{i},\mathrm{n}) + \mathrm{b}(\mathrm{i},\mathrm{n}))$.
3. Обчислити $\mathrm{g} = \tanh(\mathrm{L}(\mathrm{n}-1) * \mathrm{Wx}(\mathrm{g},\mathrm{n}) + \mathrm{h}(\mathrm{t}-1) * \mathrm{Wh}(\mathrm{g},\mathrm{n}) + \mathrm{b}(\mathrm{g},\mathrm{n}))$.
4. Обчислити $\mathrm{o} = \mathrm{sigm}(\mathrm{L}(\mathrm{n}-1) * \mathrm{Wx}(\mathrm{o},\mathrm{n}) + \mathrm{h}(\mathrm{t}-1) * \mathrm{Wh}(\mathrm{o},\mathrm{n}) + \mathrm{b}(\mathrm{o},\mathrm{n}))$.
5. Обчислити $C(\mathrm{t}) = C(\mathrm{t}-1) \bullet \mathrm{f} + \mathrm{i} \bullet \mathrm{g}$.
6. Обчислити $\mathrm{L}(\mathrm{n}) = \mathrm{h}(\mathrm{t}) = \mathrm{o} \bullet \tanh(C(\mathrm{t}))$.

Після того, як були обраховані всі потрібні LSTM-блоки, слід визначити шар результату за формулою: $\mathrm{z} = \mathrm{L}(\mathrm{n}-1) * \mathrm{V} + \mathrm{bV}$ *та застосувати функцію softmax для z: y = softmax(z).* $softmax(z) = \frac{e^{Zk}}{\sum_{l=0} e^{Zl}}$*; де k – індекс 1-го нейрона у векторі z.*

Розглянемо Backward pass. Для знаходження помилки застосовується формула ентропії: $E = -\sum_k u_k \ln(y_k)$*, де k – перерахунок нейронів вихідного шару.*

Слід вивести форму для $\frac{\partial E}{\partial z}$*. Для цього візьмемо 1-й нейрон у векторі z, та знайдемо для нього помилку, яку можна буде пристосувати для матричного вигляду:* $\frac{\partial E}{\partial z_1} = \frac{\partial E}{\partial y_1} * \frac{\partial y_1}{\partial z_1} + \frac{\partial E}{\partial y_2} * \frac{\partial y_2}{\partial z_1} + \cdots + \frac{\partial E}{\partial y_k} * \frac{\partial y_k}{\partial z_1}$.

Спочатку виведемо помилку для $\frac{\partial E}{\partial y_1}$*:* $\frac{\partial E}{\partial y_1} = \frac{\partial}{\partial y_1}(-u_1 \ln(y_1)) = -\frac{u_1}{y_1}$.

Аналогічна формула для $\{\frac{\partial E}{\partial y_1}, \frac{\partial E}{\partial y_2}, \ldots, \frac{\partial E}{\partial y_k}\}$.

Формула для матриці буде мати такий вигляд: $\frac{\partial E}{\partial y} = -\frac{u}{y}$.

Наступним кроком слід знайти $\frac{\partial y_1}{\partial z_1}$ *та* $\frac{\partial y_2}{\partial z_1}, \frac{\partial y_k}{\partial z_1}$*:*

$$\frac{\partial y_1}{\partial z_1} = \frac{\partial}{\partial z_1}\left(\frac{e^{Z_1}}{\sum_{l=1} e^{Z_l}}\right) = \frac{e^{Z_1}\sum_{l=1} e^{Z_l} - e^{Z_1} * e^{Z_1}}{(\sum_{l=1} e^{Z_l})^2} = \frac{e^{Z_1}}{\sum_{l=1} e^{Z_l}} * \left(1 - \frac{e^{Z_1}}{\sum_{l=1} e^{Z_l}}\right).$$

Оскільки $y_1 = \frac{e^{Z_1}}{\sum_{l=1} e^{Z_l}}$*, то* $\frac{\partial y_1}{\partial z_1}$ *буде мати вигляд* $\frac{\partial y_1}{\partial z_1} = y_1 * (1 - y_1)$

$$\frac{\partial y_2}{\partial z_1} = \frac{\partial}{\partial z_1}\left(\frac{e^{Z_2}}{\sum_{l=1} e^{Z_l}}\right) = -\frac{e^{Z_2} * e^{Z_1}}{(\sum_{l=1} e^{Z_l})^2} = -y_1 * y_2.$$

Для $\frac{\partial y_k}{\partial z_1}$ *результат буде схожий з* $\frac{\partial y_2}{\partial z_1}$*:* $\frac{\partial y_k}{\partial z_1} = \frac{\partial}{\partial z_1}\left(\frac{e^{Zk}}{\sum_{l=1} e^{Zl}}\right) = -\frac{e^{Zk} * e^{Z1}}{(\sum_{l=1} e^{Zl})^2} = -y_1 * y_k$.

Тепер знаходимо помилку для z_1:

$$\frac{\partial E}{\partial z_1} = -\frac{u_1}{y_1} * y_1 * (1 - y_1) + \frac{u_2}{y_2} * y_1 * y_2 + \cdots + \frac{u_k}{y_k} * y_1 * y_k =$$
$$-u_1 + u_1 y_1 + u_2 * y_1 + \cdots + u_k * y_1 = y_1(u_1 + u_2 + \cdots u_k) - u_1.$$

Оскільки $\sum_k u_k = 1$ через те, що на виході має бути лише один клас (для знаходження декількох класів не використовують функцію softmax) та інша формула знаходження помилок, то помилка для z_1 матиме вигляд: $\frac{\partial E}{\partial z_1} = y_1 - u_1$. Формула $\frac{\partial E}{\partial z}$ для матриць матиме вигляд: $\frac{\partial E}{\partial z} = y - u$.

Далі знаходимо помилку для вихідних даних з LSTM-блоку:

$\frac{\partial E}{\partial L} = \frac{\partial E}{\partial z}\frac{\partial z}{\partial L}$, *де* $\frac{\partial z}{\partial L} = \frac{\partial}{\partial L}(\mathrm{L} * \mathrm{V} + \mathrm{bV}) = \mathrm{V}$.

Робимо транспонування матриці V для того, щоб розмірності матриць відповідали правилам множення матриць: $\frac{\partial E}{\partial L} = \frac{\partial E}{\partial z} * V^T$.

Знаходимо помилку для Cell State за формулою: $\frac{\partial E}{\partial C} = \frac{\partial E}{\partial L} * \frac{\partial L}{\partial C}$, *де*

$\frac{\partial L}{\partial C} = \frac{\partial}{\partial C}(o \bullet \tanh(C)) = o \bullet (1 - tan^2(C))$, $\frac{\partial E}{\partial C} = \frac{\partial E}{\partial L} \bullet o \bullet \left(1 - tan^2(C)\right)$.

Помилка для Output gate обчислюється за формулою:

$\frac{\partial E}{\partial o} = \frac{\partial E}{\partial L} * \frac{\partial L}{\partial o}$, *де* $\frac{\partial L}{\partial o} = \frac{\partial}{\partial o}(o \bullet \tanh(C)) = \tanh(C)$, $\frac{\partial E}{\partial o} = \frac{\partial E}{\partial L} \bullet \tanh(C)$.

Помилка для Input gate:

$\frac{\partial E}{\partial i} = \frac{\partial E}{\partial L} * \frac{\partial L}{\partial C} * \frac{\partial C}{\partial i} = \frac{\partial E}{\partial C} * \frac{\partial C}{\partial i}$, *де* $\frac{\partial C}{\partial i} = \frac{\partial}{\partial i}(C(\mathrm{t}-1) \bullet \mathrm{f} + \mathrm{i} \bullet \mathrm{g}) = g$, $\frac{\partial E}{\partial i} = \frac{\partial E}{\partial C} \bullet \mathrm{g}$.

Помилка для Input modify gate:

$\frac{\partial E}{\partial g} = \frac{\partial E}{\partial L} * \frac{\partial L}{\partial C} * \frac{\partial C}{\partial g} = \frac{\partial E}{\partial C} * \frac{\partial C}{\partial g}$, *де* $\frac{\partial C}{\partial g} = \frac{\partial}{\partial g}(C(\mathrm{t}-1) \bullet \mathrm{f} + \mathrm{i} \bullet \mathrm{g}) = i$, $\frac{\partial E}{\partial g} = \frac{\partial E}{\partial C} \bullet \mathrm{i}$.

Помилка для Forget gate: $\frac{\partial E}{\partial f} = \frac{\partial E}{\partial L} * \frac{\partial L}{\partial C} * \frac{\partial C}{\partial f} = \frac{\partial E}{\partial C} * \frac{\partial C}{\partial f}$,

де $\frac{\partial C}{\partial f} = \frac{\partial}{\partial f}(C(\mathrm{t}-1) \bullet \mathrm{f} + \mathrm{i} \bullet \mathrm{g}) = C(t-1)$, $\frac{\partial E}{\partial f} = \frac{\partial E}{\partial C} \bullet C(\mathrm{t}-1)$.

Далі знаходимо дельти (Δ) для всіх ваг і нейронів зсуву. Слід також враховувати те, що дельти будуть різні для кожного моменту часу.

Знаходження дельти для нейронів зсуву (до вихідного шару):

$\Delta bV(t) = \frac{\partial E}{\partial bV} = \frac{\partial E}{\partial z}\frac{\partial z}{\partial bV}$, *де* $\frac{\partial z}{\partial bV} = \frac{\partial}{\partial bV}(\mathrm{L} * \mathrm{V} + \mathrm{bV}) = 1$, $\Delta bV(t) = \frac{\partial E}{\partial z}$.

Знаходження ваг до вихідного шару виконуємо за формулою:

$\Delta V(t) = \frac{\partial E}{\partial V} = \frac{\partial E}{\partial z}\frac{\partial z}{\partial V}$, *де* $\frac{\partial z}{\partial V} = \frac{\partial}{\partial V}(L * V + bV) = L$, $\Delta V(t) = L^T \frac{\partial E}{\partial z}$.

Output gate. Знаходження дельти нейронів зсуву:

$$\Delta b(o,n)(t) = \frac{\partial E}{\partial b(o,n)} = \frac{\partial E}{\partial o}\frac{\partial o}{\partial b(o,n)}$$

$$\frac{\partial o}{\partial b(o,n)} = \frac{\partial}{\partial b(o,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(o,n)} + \mathrm{h(t-1)} * \mathrm{Wh(o,n)} + \mathrm{b(o,n)}\big)\Big) = o \bullet (1 - o).$$

$$\Delta b(o,n)(t) = \frac{\partial E}{\partial b(o,n)} = \frac{\partial E}{\partial o} \bullet o \bullet (1-o).$$

Знаходження ваг для L(n-1): $\Delta Wx(o,n)(t) = \frac{\partial E}{\partial Wx(o,n)} = \frac{\partial E}{\partial o}\frac{\partial o}{\partial Wx(o,n)}$

$$\frac{\partial o}{\partial Wx(o,n)} = \frac{\partial}{\partial Wx(o,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(o,n)} + \mathrm{h(t-1)} * \mathrm{Wh(o,n)} + \mathrm{b(o,n)}\big)\Big) = L^T(n-1) * o \bullet (1-o).$$

$$\Delta Wx(o,n)(t) = \frac{\partial E}{\partial Wx(o,n)} = L^T(n-1) * \frac{\partial E}{\partial o} \bullet o \bullet (1-o).$$

Знаходження ваг для h(n-1): $\Delta Wh(o,n)(t) = \frac{\partial E}{\partial Wh(o,n)} = \frac{\partial E}{\partial o}\frac{\partial o}{\partial Wh(o,n)}$

$$\frac{\partial o}{\partial Wh(o,n)} = \frac{\partial}{\partial Wh(o,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(o,n)} + \mathrm{h(t-1)} * \mathrm{Wh(o,n)} + \mathrm{b(o,n)}\big)\Big) = h^T(n-1) * o \bullet (1-o).$$

$$\Delta Wh(o,n)(t) = \frac{\partial E}{\partial Wh(o,n)} = h^T(n-1) * \frac{\partial E}{\partial o} \bullet o \bullet (1-o).$$

Input gate. Знаходження дельти нейронів зсуву:

$$\Delta b(i,n)(t) = \frac{\partial E}{\partial b(i,n)} = \frac{\partial E}{\partial i}\frac{\partial i}{\partial b(i,n)}$$

$$\frac{\partial i}{\partial b(i,n)} = \frac{\partial}{\partial b(i,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(i,n)} + \mathrm{h(t-1)} * \mathrm{Wh(i,n)} + \mathrm{b(i,n)}\big)\Big) = i \bullet (1-i).$$

$$\Delta b(i,n)(t) = \frac{\partial E}{\partial b(i,n)} = \frac{\partial E}{\partial i} \bullet i \bullet (1-i).$$

Знаходження ваг для L(n-1): $\Delta Wx(i,n)(t) = \frac{\partial E}{\partial Wx(i,n)} = \frac{\partial E}{\partial i}\frac{\partial i}{\partial Wx(i,n)}$

$$\frac{\partial i}{\partial Wx(i,n)} = \frac{\partial}{\partial Wx(i,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(i,n)} + \mathrm{h(t-1)} * \mathrm{Wh(i,n)} + \mathrm{b(i,n)}\big)\Big) = L^T(n-1) * i \bullet (1-i).$$

$$\Delta Wx(i,n)(t) = \frac{\partial E}{\partial Wx(i,n)} = L^T(n-1) * \frac{\partial E}{\partial i} \bullet i \bullet (1-i).$$

Знаходження ваг для h(n-1): $\Delta Wh(i,n)(t) = \frac{\partial E}{\partial Wh(i,n)} = \frac{\partial E}{\partial i}\frac{\partial i}{\partial Wh(i,n)}$

$$\frac{\partial i}{\partial Wh(i,n)} = \frac{\partial}{\partial Wh(i,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(i,n)} + \mathrm{h(t-1)} * \mathrm{Wh(i,n)} + \mathrm{b(i,n)}\big)\Big) = h^T(n-1) * i \bullet (1-i).$$

$$\Delta Wh(i,n)(t) = \frac{\partial E}{\partial Wh(i,n)} = h^T(n-1) * \frac{\partial E}{\partial i} \bullet i \bullet (1-i).$$

Input modify gate. Знаходження дельти нейронів зсуву:

$$\Delta b(g,n)(t) = \frac{\partial E}{\partial b(g,n)} = \frac{\partial E}{\partial g}\frac{\partial g}{\partial b(g,n)}$$

$$\frac{\partial g}{\partial b(g,n)} = \frac{\partial}{\partial b(g,n)}\Big(tanh\big(\mathrm{L(n-1)} * \mathrm{Wx(g,n)} + \mathrm{h(t-1)} * \mathrm{Wh(g,n)} + \mathrm{b(g,n)}\big)\Big) = (1-g^2).$$

$$\Delta b(g,n)(t) = \frac{\partial E}{\partial b(g,n)} = \frac{\partial E}{\partial g} \bullet (1-g^2).$$

Знаходження ваг для L(n-1): $\Delta Wx(g,n)(t) = \frac{\partial E}{\partial Wx(g,n)} = \frac{\partial E}{\partial g}\frac{\partial g}{\partial Wx(g,n)}$

$$\frac{\partial g}{\partial Wx(g,n)} = \frac{\partial}{\partial Wx(g,n)}\Big(tanh\big(\mathrm{L(n-1)} * \mathrm{Wx(g,n)} + \mathrm{h(t-1)} * \mathrm{Wh(g,n)} + \mathrm{b(g,n)}\big)\Big) = L^T(n-1) * (1-g^2).$$

$$\Delta Wx(g,n)(t) = \frac{\partial E}{\partial Wx(g,n)} = L^T(n-1) * \frac{\partial E}{\partial g} \bullet (1-g^2).$$

Знаходження ваг для h(n-1): $\Delta Wh(g,n)(t) = \frac{\partial E}{\partial Wh(g,n)} = \frac{\partial E}{\partial g}\frac{\partial g}{\partial Wh(g,n)}$

$$\frac{\partial g}{\partial Wh(g,n)} = \frac{\partial}{\partial Wh(g,n)}\Big(tanh\big(\mathrm{L(n-1)} * \mathrm{Wx(g,n)} + \mathrm{h(t-1)} * \mathrm{Wh(g,n)} + \mathrm{b(g,n)}\big)\Big) = h^T(n-1) * (1-g^2).$$

$$\Delta Wh(g,n)(t) = \frac{\partial E}{\partial Wh(g,n)} = h^T(n-1) * \frac{\partial E}{\partial g} \bullet (1-g^2).$$

Forget gate. Знаходження дельти нейронів зсуву:

$$\Delta b(f,n)(t) = \frac{\partial E}{\partial b(f,n)} = \frac{\partial E}{\partial f}\frac{\partial f}{\partial b(f,n)}$$

$$\frac{\partial f}{\partial b(f,n)} = \frac{\partial}{\partial b(f,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(f,n)} + \mathrm{h(t-1)} * \mathrm{Wh(f,n)} + \mathrm{b(f,n)}\big)\Big) = f \bullet (1-f).$$

$$\Delta b(f,n)(t) = \frac{\partial E}{\partial b(f,n)} = \frac{\partial E}{\partial f} \bullet f \bullet (1-f).$$

Знаходження ваг для L(n-1): $\Delta Wx(f,n)(t) = \frac{\partial E}{\partial Wx(f,n)} = \frac{\partial E}{\partial f}\frac{\partial f}{\partial Wx(f,n)}$

$$\frac{\partial f}{\partial Wx(f,n)} = \frac{\partial}{\partial Wx(f,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(f,n)} + \mathrm{h(t-1)} * \mathrm{Wh(f,n)} + \mathrm{b(f,n)}\big)\Big) = L^T(n-1) * f \bullet (1-f).$$

$$\Delta Wx(f,n)(t) = \frac{\partial E}{\partial Wx(f,n)} = L^T(n-1) * \frac{\partial E}{\partial f} \bullet f \bullet (1-f).$$

Знаходження ваг для h(n-1): $\Delta Wh(f,n)(t) = \frac{\partial E}{\partial Wh(f,n)} = \frac{\partial E}{\partial f}\frac{\partial f}{\partial Wh(f,n)}$

$$\frac{\partial f}{\partial Wh(f,n)} = \frac{\partial}{\partial Wh(f,n)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(f,n)} + \mathrm{h(t-1)} * \mathrm{Wh(f,n)} + \mathrm{b(f,n)}\big)\Big) = h^T(n-1) * f \bullet (1-f).$$

$$\Delta Wh(f,n)(t) = \frac{\partial E}{\partial Wh(f,n)} = h^T(n-1) * \frac{\partial E}{\partial f} \bullet f \bullet (1-f).$$

Наступним кроком є знаходження помилки для L(n-1), щоб передати її значення до наступного LSTM-блоку:

$$\frac{\partial E}{\partial L(n-1)} = \frac{\partial E}{\partial o}\frac{\partial o}{\partial L(n-1)} + \frac{\partial E}{\partial i}\frac{\partial i}{\partial L(n-1)} + \frac{\partial E}{\partial g}\frac{\partial g}{\partial L(n-1)} + \frac{\partial E}{\partial f}\frac{\partial f}{\partial L(n-1)}$$

$$\frac{\partial o}{\partial L(n-1)} = \frac{\partial}{\partial L(n-1)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(o,n)} + \mathrm{h(t-1)} * \mathrm{Wh(o,n)} + \mathrm{b(o,n)}\big)\Big) = o \bullet (1-o) * Wx^T(o,n)$$

$$\frac{\partial E}{\partial o}\frac{\partial o}{\partial L(n-1)} = \frac{\partial E}{\partial o} \bullet o \bullet (1-o) * Wx^T(o,n)$$

$$\frac{\partial i}{\partial L(n-1)} = \frac{\partial}{\partial L(n-1)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(i,n)} + \mathrm{h(t-1)} * \mathrm{Wh(i,n)} + \mathrm{b(i,n)}\big)\Big) = i \bullet (1-i) * Wx^T(i,n)$$

$$\frac{\partial E}{\partial i}\frac{\partial i}{\partial L(n-1)} = \frac{\partial E}{\partial i} \bullet i \bullet (1-i) * Wx^T(i,n)$$

$$\frac{\partial g}{\partial L(n-1)} = \frac{\partial}{\partial L(n-1)}\Big(tanh\big(\mathrm{L(n-1)} * \mathrm{Wx(g,n)} + \mathrm{h(t-1)} * \mathrm{Wh(g,n)} + \mathrm{b(g,n)}\big)\Big) = (1-g^2) * Wx^T(g,n)$$

$$\frac{\partial E}{\partial g}\frac{\partial g}{\partial L(n-1)} = \frac{\partial E}{\partial g} \bullet (1-g^2) * Wx^T(g,n)$$

$$\frac{\partial f}{\partial L(n-1)} = \frac{\partial}{\partial L(n-1)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(f,n)} + \mathrm{h(t-1)} * \mathrm{Wh(f,n)} + \mathrm{b(f,n)}\big)\Big) = f \bullet (1-f) * Wx^T(f,n)$$

$$\frac{\partial E}{\partial f}\frac{\partial f}{\partial L(n-1)} = \frac{\partial E}{\partial f} \bullet f \bullet (1-f) * Wx^T(f,n).$$

Потім відбувається знаходження помилки для h(n-1), щоб передати її значення до LSTM-блоку попереднього моменту часу:

$$\frac{\partial E}{\partial h(t-1)} = \frac{\partial E}{\partial o}\frac{\partial o}{\partial h(t-1)} + \frac{\partial E}{\partial i}\frac{\partial i}{\partial h(t-1)} + \frac{\partial E}{\partial g}\frac{\partial g}{\partial h(t-1)} + \frac{\partial E}{\partial f}\frac{\partial f}{\partial h(t-1)}$$

$$\frac{\partial o}{\partial h(t-1)} = \frac{\partial}{\partial h(t-1)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(o,n)} + \mathrm{h(t-1)} * \mathrm{Wh(o,n)} + \mathrm{b(o,n)}\big)\Big) = o \bullet (1-o) * Wh^T(o,n)$$

$$\frac{\partial E}{\partial o}\frac{\partial o}{\partial h(t-1)} = \frac{\partial E}{\partial o} \bullet o \bullet (1-o) * Wh^T(o,n)$$

$$\frac{\partial i}{\partial h(t-1)} = \frac{\partial}{\partial h(t-1)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(i,n)} + \mathrm{h(t-1)} * \mathrm{Wh(i,n)} + \mathrm{b(i,n)}\big)\Big) = i \bullet (1-i) * Wh^T(i,n)$$

$$\frac{\partial E}{\partial i}\frac{\partial i}{\partial h(t-1)} = \frac{\partial E}{\partial i} \bullet i \bullet (1-i) * Wh^T(i,n)$$

$$\frac{\partial g}{\partial h(t-1)} = \frac{\partial}{\partial h(t-1)}\Big(tanh\big(\mathrm{L(n-1)} * \mathrm{Wx(g,n)} + \mathrm{h(t-1)} * \mathrm{Wh(g,n)} + \mathrm{b(g,n)}\big)\Big) = (1-g^2) * Wh^T(g,n)$$

$$\frac{\partial E}{\partial g}\frac{\partial g}{\partial h(t-1)} = \frac{\partial E}{\partial g} \bullet (1-g^2) * Wh^T(g,n)$$

$$\frac{\partial f}{\partial h(t-1)} = \frac{\partial}{\partial h(t-1)}\Big(sigm\big(\mathrm{L(n-1)} * \mathrm{Wx(f,n)} + \mathrm{h(t-1)} * \mathrm{Wh(f,n)} + \mathrm{b(f,n)}\big)\Big) == f \bullet (1-f) * Wh^T(f,n)$$

$$\frac{\partial E}{\partial f}\frac{\partial f}{\partial h(t-1)} = \frac{\partial E}{\partial f} \bullet f \bullet (1-f) * Wh^T(f,n).$$

Далі знаходимо ваги для інших LSTM-блоків за такими правилами:

1. Якщо LSTM-блок має два виходи: до наступного шару та наступного моменту часу, то помилка для його виходу буде розраховуватися за формулою: $\frac{\partial E}{\partial L} = \frac{\partial E}{\partial L} + \frac{\partial E}{\partial h}$.

2. Якщо LSTM-блок має один вихід – до наступного моменту часу, то помилка для його виходу буде розраховуватися за формулою: $\frac{\partial E}{\partial L} = \frac{\partial E}{\partial h}$.

3. Якщо LSTM-блок має один вихід – до наступного шару, то береться просто знайдена помилка $\frac{\partial E}{\partial L}$.

Для редагування ваг потрібно додати всі дельти в часі:

$$\Delta b(\{o,i,g,f\},n) = \sum_i^t \Delta b(\{o,i,g,f\},n)(i).$$

$$\Delta Wx(\{o,i,g,f\},n) = \sum_i^t \Delta Wx(\{o,i,g,f\},n)(i).$$

$$\Delta Wh(\{o,i,g,f\},n) = \sum_i^t \Delta Wh(\{o,i,g,f\},n)(i).$$

$\Delta V = \sum_i^t \Delta V(i)$ $\Delta bV = \sum_i^t \Delta bV(i)$.

Корегування ваг має такий вигляд:

$b(\{o,i,g,f\},n) = b(\{o,i,g,f\},n) + \Delta b(\{o,i,g,f\},n)\eta$

$Wx(\{o,i,g,f\},n) = Wx(\{o,i,g,f\},n) + \Delta Wx(\{o,i,g,f\},n)$

$Wh(\{o,i,g,f\},n) = Wh(\{o,i,g,f\},n) + \Delta Wh(\{o,i,g,f\},n)$

$V = V + \Delta V,\ bV = bV + \Delta bV.$

Використання LSTM в NLP-задачах. Neural Linguistic Programing (NLP) – робота з людською мовою, коли програма, спираючись на математичні алгоритми лінгвістики та машинне навчання, видає потрібний результат (Бурлєєв, Василенко та Іваненко, 2021). Є багато задач, де це використовують, зокрема:

1. Пояснення кроків прогнозування (коли НМ, спираючись на дані, які є в базі даних, може «пояснити» логіку отримання результату, наприклад прогнозування погоди (виведення різних параметрів, на яких воно базувалося)).

2. Перетворення звукових файлів у текстовий формат (наприклад, у багатьох мобільних застосунках є введення тексту за допомогою голосу).

3. Розпізнавання тональності тексту (оцінка коментарів під відео чи іншими медійними продуктами).

4. Опис фотографії (використовується, наприклад, в Instagram для альтернативного тексту, якщо не завантажилася фотографія).

5. Визначення класу, до якого зараховуємо текст (дещо подібне до тональності, але більш широке (наприклад, визначення мови, якою написаний текст)).

6. Чат-бот (НМ, з якою можна поспілкуватися (наприклад, бот для підтримки користувачів, який може зменшити навантаження на працівників компанії, відповідаючи на прості запитання клієнтів)).

Перекладач тексту.

Для покращення результату NLP використовують різні методи:

1. Токенізатор – за його допомогою розбивають текст на окремі речення, які у свою чергу розбиваються на слова. Покращеннями роботи через токенайзер, зокрема, є:

– фільтрація символів у тексті;
– переведення слів до нижнього регістру;
– морфологічний аналіз слова (розбиття на префікс, корінь і т. п.);
– визначення частини мови та набір граматичних атрибутів (рід, час, ступінь порівняння і т. ін.).

Після токенізації створюється база знань (БЗ), де мають міститися слова та їхні характеристики. Різноманітні слова можуть бути додані цілеспрямовано (наприклад, для формування текстових відповідей). Хоча слова краще обирати з навчальної вибірки та створювати відповіді на їх основі. Коли починається навчання, то текст подається на вхід токенізатора, проходить по всіх описаних кроках, знаходиться відповідне абсолютне значення для кожного слова та формується масив векторів, які краще нормалізувати, щоб зменшити ймовірність «вибуху» градієнта (наприклад, поділивши абсолютне значення на його максимум).

2. Стоп-слова – це слова в тексті, які часто використовуються, проте не так допомагають у розрахунках, як вводять непотрібні шуми. Для цього під час створення бази знань, спираючись на навчальну вибірку, можна підрахувати кількість слів, які траплялися в тексті. І, наприклад, якщо якесь слово має 0,5 % від кількості слів, то не виводити його, або не брати 10 топслів, які є в базі знань. Іноді ж просто створюють масиви таких слів вручну, щоб потім їх обходити.

3. N-грама, де N – це кількість слів у блоці, на які буде розбиватися текст. N-грама застосовується для пошуку «зв'язних» текстових блоків. Наприклад, текст «Я люблю нейромережі та програмування» можна розбити на такі блоки:

– Розбиття на 1-граму (уніграма):
– [«Я», «люблю», «нейромережі», «та», «програмування»].
– Розбиття на 2-граму (біграма):
– [«Я люблю», «люблю нейромережі», «нейромережі та», «та програмування»].
– Розбиття на 3-граму (триграма):

– [«Я люблю нейромережі», «люблю нейромережі та», «нейромережі та програмування»].

Приклад NLP на основі LSTM. Для демонстрації роботи LSTM вирішення NLP взято приклад «Визначення мови тексту». Мовами, які будуть визначатися, обрано українську, англійську, болгарську, італійську, польську, російську.

У прикладі застосовано ієрархічну структуру, тобто створюється певна кількість груп для класів з однаковими ознаками. Для того щоб спочатку визначити групи, а вже після знайти правильний варіант серед класів, що до них входять.

У процесі визначення мов можна виділити тип мови, у нашому разі це кирилиця та латиниця, тобто маємо 2 групи типів мов. До кирилиці можна зарахувати українську, російську та болгарську. До латиниці зараховуємо польську, італійську й англійську. Ієрархічна структура підвищує точність, але слід навчати три окремі НМ. Далі відбувається визначення групи, а потім визначення класу в групі, яка «перемогла».

Під час запуску програми розпізнавання мови друкованого тексту (на основі навчання НМ) відкривається папка, де є 2 вкладені папки, кожна з яких символізує групу. У кожній папці є відповідні класи, що розбиті на текстові файли, в кожному з яких розміщені тексти для навчання, що зчитуються і розбиваються на речення. На основі аналізу текстів формується БЗ і навчальна база даних.

Під час запуску (не першого разу) програма зчитує ваги, якщо НМ вже була навчена до цього. У процесі навчання використано для кожної НМ 64 «слоти» пам'яті, кожен з яких міститиме на вході одне слово. Після вхідного шару йдуть 3 шари LSTM (1-й шар – 72 нейрони, 2-й шар – 48 нейронів, 3-й шар – 24 нейрони) та вихідний шар (під час визначення групи містить 2 виходи, а під час визначення класів – 3 виходи). У процесі передачі навчальних даних (навчальної послідовності) до НМ, якщо всі 64 «слоти» пам'яті не заповнені, вони наповнюються цими ж даними поки не будуть повністю зайняті всі слоти.

Інтерфейс програми (рис. 4) має такий функціонал:

– у разі натискання на кнопку Визначення типу запускається навчання для визначення групи, до якої зараховується текст;

– у разі натискання на кнопку Визначення кирилиці або на кнопку Визначення латиниці запускається навчання НМ, щоб визначати потім, до якого класу належить текст;

– TextBox розміщений над кнопками навчання, у ньому вказується кількість навчальних речень, що будуть подані до НМ;

– кнопка Зберегти ваги зберігає всі ваги в кореневій папці програми;

– зліва від кнопки Збереження розташовані індикатори прогресу навчання НМ, усі навчання НМ можна запускати паралельно;

– лівий верхній TextBox зчитує текст для розпізнавання;

– кнопка Визначити запускає визначення тексту, який був розташований у TextBox

– під кнопкою Визначити розташовані результати: тип мови – група, до якої зараховується текст; назва мови – клас, до якого зараховується текст, що розпізнається.

Навчання для визначення групи мови проходить відносно легко та швидко. Це можна побачити на графіку впевненості визначення під час навчання. Графік

поступово зростає (рис. 5). Має також бути кращий результат навчання, щоб ймовірність помилкового класу була мінімальна.

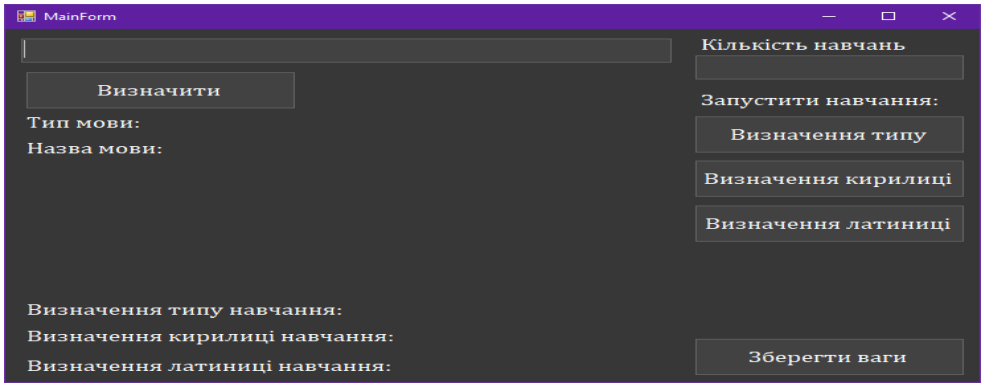

Рис. 4. Інтерфейс програми розпізнавання мови тексту

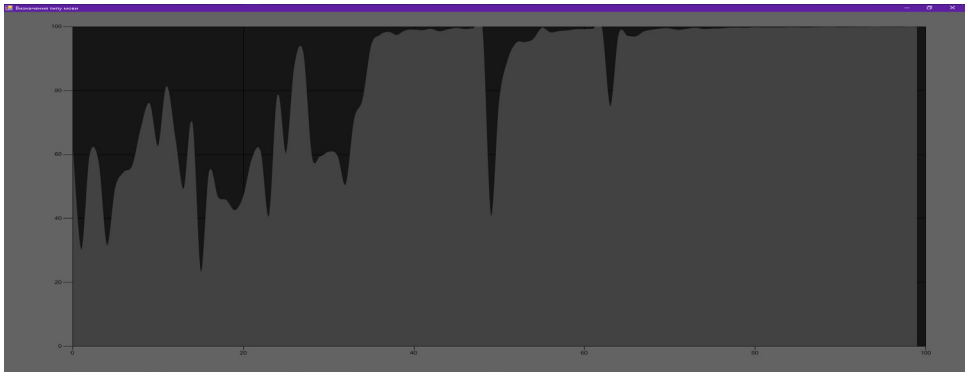

Рис. 5. Графік результатів навчань «Визначення групи»

На рис. 6 показано графік результатів навчань для класів кирилиці.

Можемо побачити, що «точність» понад 80 %, через те що у вибраних мовах багато слів, які схожі одне на одного. Тому слід використовувати або більше зв'язків, або іншу структуру нейронів й обробки слів.

На рис. 7 показано графік результатів навчань для класів латиниці.

Можемо побачити, що «точність» наближається до 90 %, це значно вище, ніж для визначення кирилиці, через те що менше слів перетинається.

Приклади роботи програми розпізнавання мови тексту, яка побудована на основі НМ LSTM (рис. 8), демонструють той факт, що тип мови визначається достатньо легко та з високою точністю.

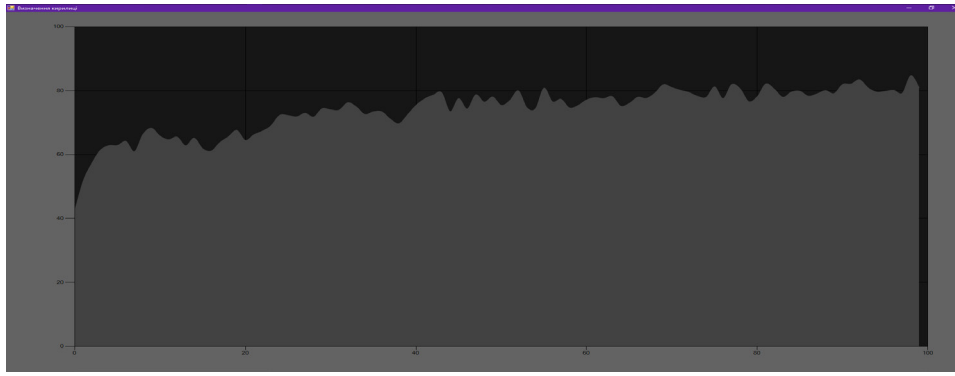

Рис. 6. Графік результатів навчань «Визначення кирилиці»

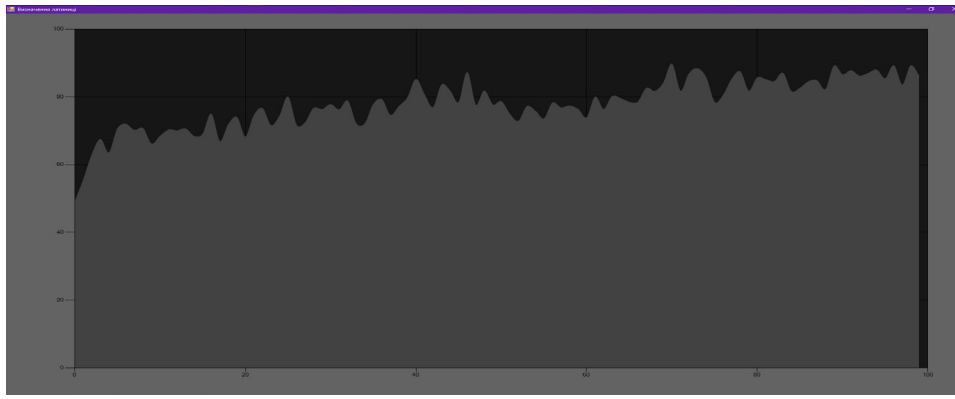

Рис. 7. Графік результатів навчань «Визначення латиниці»

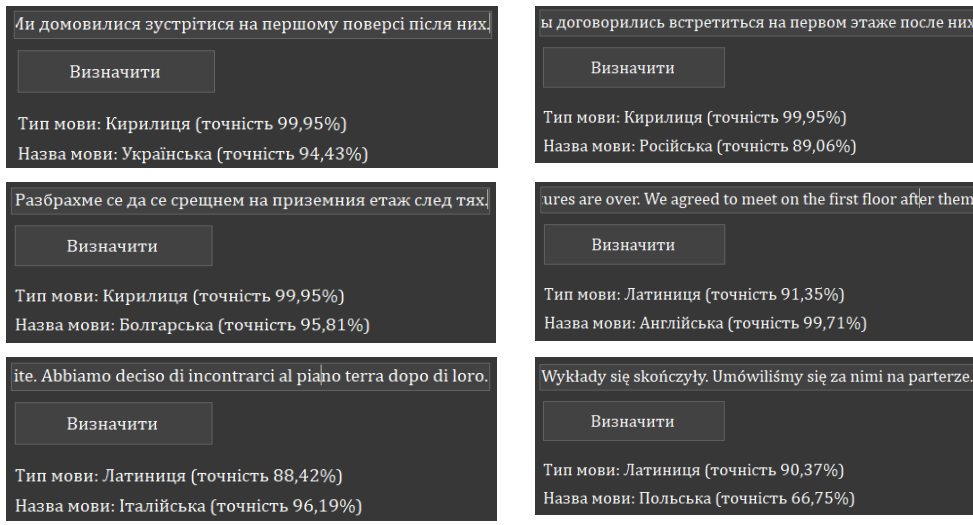

Рис. 8. Приклади роботи програми розпізнавання друкованого тексту

Висновки. У роботі розглянуто наявні погляди щодо розпізнавання образів на прикладі друкованих текстів; проаналізовано сучасні підходи з використанням нейромереж та їх навчанням.

Ураховуючи результати проведеного аналізу використання НМ, розроблено програмний продукт розпізнавання друкованих текстів, визначення мов і типів мов (кирилиця чи латиниця) за допомогою використання нейромереж, що навчаються.

СПИСОК ПОСИЛАНЬ

Алексеев, П.П. и Квятковская, И.Ю., 2021. Применение нейронных сетей для распознавания принципиальных условно-графических электрических обозначений. *Вестник Астраханского государственного технического университета. Серия: Управление, вычислительная техника и информатика*, [online] 2, с.47-56. Доступно: <http://www.mathnet.ru/links/57f0ef0c3d69c2f4645481ceae10f0f4/ vagtu669.pdf> [Дата звернення 25 квітня 2022].

Бурлєєв, О., Василенко, О. та Іваненко, Р., 2021. Ефективність використання штучних нейронних мереж в економіці. *Економіка та суспільство*, [online] 31. Доступно: <https://doi.org/10.32782/2524-0072/2021-31-27> [Дата звернення 22 квітня 2022].

Лещинський, О.Л. та Іщенко, А.О., 2017. Використання нейромереж у процесі інтелектуального (кластерного) аналізу даних. *Економіка та суспільство*, [online] 11, с.578-581. Доступно: <https://economyandsociety.in.ua/journals/11_ukr/93.pdf> [Дата звернення 23 квітня 2022].

Маркова, С.В. и Жигалов, К.Ю., 2017. Применение нейронной сети для создания системы распознавания изображений. *Фундаментальные исследования*, [online] 8 (1), с.60-64. Доступно: <> [Дата звернення 25 квітня 2022].

Олейник, А., Зыбина, Е. и Олейник, Е., 2020. Обзор Решений Нейронных Сетей Применяемых для Распознавания Образов. В: *Інформаційні системи та технології ICT-2020.* 9 Міжнародна науково-технічна конференція. Харків, Україна, [online] 17-20 листопада 2020 р., с.128-131. Харків: Друкарня Мадрид. Доступно: <> [Дата звернення 25 квітня 2022].

Ставицький, О.В. та Мозолевська, М.О., 2017. Використання нейронних мереж для прогнозування у фінансовій сфері. *Актуальні проблеми економіки та управління,* [online] 11. Доступно: < http://ape.fmm. kpi.ua/article/view/102584 > [Дата звернення 23 квітня 2022].

Субботін, С.О., Олійник, А.О. та Олійник, О.О., 2009. *Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей*. Запоріжжя: ЗНТУ.

Bengio, Y., 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2, pp.1-127.

Bengio, Y., LeCun, Y. and Hinton G. 2015. Deep Learning. *Nature*, 521. pp.436-444.

Gers, F., Schraudolph, N. and Schmidhuber, J. 2002. Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, 3, pp.115-143.

Girshick, R., 2015. Fast R-CNN. *Computer Vision and Pattern Recognition*, [online] Available at: <https://arxiv.org/abs/1504.08083> [Accessed 22 April 2022].

Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative Adversarial Networks. *Cornell University*, [online] 10 June. Available at: <https://arxiv.org/pdf/1406.2661.pdf> [Accessed 25 April 2022].

Schmidhuber, J., 2015. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61, pp.85-117.

What are Recurrent Neural Networks. [online] Available at: <https://www.ibm.com/ cloud/learn/recurrent-neural-networks> [Accessed 22 April 2022].

Zheng, Y., Meng, Y. and Jin, Y., 2011. Object Recognition using Neural Networks with Bottom-up and Top-down Pathways. *Neurocomputing*, 74, pp.3158-3169.

REFERENCES

Alekseev, P.P. and Kviatkovskaia, I.Iu., 2021. Primenenie neironnykh setei dlia raspoznavaniia printcipialnykh uslovno-graficheskikh elektricheskikh oboznachenii [The use of neural networks for the recognition of fundamental conventional graphic electrical symbols]. *Vestnik Astrakhanskogo gosudarstvennogo tekhnicheskogo universiteta. Seriia: Upravlenie, vychislitelnaia tekhnika i informatika*, [online] 2, pp.47-56. Available at: <http://www.mathnet.ru/links/57f0ef0c3d69c2f4645481ceae10f0f4/ vagtu669.pdf> [Accessed 25 April 2022].

Bengio, Y., 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2, pp.1-127.

Bengio, Y., LeCun, Y. and Hinton G. 2015. Deep Learning. *Nature*, 521. pp.436-444.

Burlieiev, O., Vasylenko, O. and Ivanenko, R., 2021. Efektyvnist vykorystannia shtuchnykh neironnykh merezh v ekonomitsi [Efficiency of use of artificial neural networks in economy]. *Ekonomika ta suspilstvo*, [online] 31. Available at: <https://doi.org/10.32782/2524-0072/2021-31-27> [Accessed 22 April 2022].

Gers, F., Schraudolph, N. and Schmidhuber, J. 2002. Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, 3, pp.115-143.

Girshick, R., 2015. Fast R-CNN. *Computer Vision and Pattern Recognition*, [online] Available at: <https://arxiv.org/abs/1504.08083> [Accessed 22 April 2022].

Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative Adversarial Networks. *Cornell University*, [online] 10 June. Available at: <https://arxiv.org/pdf/1406.2661.pdf> [Accessed 25 April 2022].

Leshchynskyi, O.L. and Ishchenko, A.O., 2017. Vykorystannia neiromerezh u protsesi intelektualnoho (klasternoho) analizu danykh [The use of neural networks in the process of intelligent (cluster) data analysis]. *Ekonomika ta suspilstvo*, [online] 11, pp.578-581. Available at: <https://economyandsociety.in.ua/journals/11_ukr/93.pdf> [Accessed 23 April 2022].

Markova, S.V. and Zhigalov, K.Iu., 2017. Primenenie neironnoi seti dlia sozdaniia sistemy raspoznavaniia izobrazhenii [Application of a neural network to create an image recognition system]. *Fundamentalnye issledovaniia*, [online] 8 (1), pp.60-64. Available at: <http://www.fundamental-research. ru/ru/article/view?id=41621> [Accessed 25 April 2022].

Oleinik, A., Zybina, E. and Oleinik, E., 2020. Obzor Reshenii Neironnykh Setei Primeniaemykh dlia Raspoznavaniia Obrazov [Overview of Neural Network Solutions Applied to Pattern Recognition]. In: *Informatsiini systemy ta tekhnolohii IST-2020* [Information systems and technologies IST-2020]. 9th International Scientific and Technical Conference. Kharkiv, Ukraine, [online] November 17-20, 2020, pp.128-131. Kharkiv: Madrid Printing House. Available at: <https://openarchive.nure.ua/bitstream/document/16163/1/IST-2020-128-131.pdf> [Accessed 25 April 2022].

Schmidhuber, J., 2015. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61, pp.85-117.

Stavytskyi, O.V. and Mozolevska, M.O., 2017. Vykorystannia neironnykh merezh dlia prohnozuvannia u finansovii sferi [The use of neural networks for forecasting in the financial sector]. *Aktualni problemy ekonomiky ta upravlinnia*, [online] 11. Available at: <http://ape.fmm. kpi.ua/article/view/102584> [Accessed 23 April 2022].

Subbotin, S.O., Oliinyk, A.O. and Oliinyk, O.O., 2009. *Neiteratyvni, evoliutsiini ta multyahentni metody syntezu nechitkolohichnykh i neiromerezhnykh modelei* [Non-iterative, evolutionary and multiagent methods of synthesis of fuzzy and neural network models]. Zaporozhye: ZNTU.
What are Recurrent Neural Networks. [online] Available at: <https://www.ibm.com/ cloud/ learn/recurrent-neural-networks> [Accessed 22 April 2022].
Zheng, Y., Meng, Y. and Jin, Y., 2011. Object Recognition using Neural Networks with Bottom-up and Top-down Pathways. *Neurocomputing*, 74, pp.3158-3169.

UDC 004.032.26:004.352.246

Tkachenko Kostiantyn,
PhD in Economics,
Associate Professor at the Department of Information Technology and Design,
State University of Infrastructure and Technology,
Kyiv, Ukraine
tkachenko.kostyantyn@gmail.com
https://orcid.org/0000-0003-0549-3396

Zuienko Oleksii,
Bachelor at the Department of Information Technology and Design,
State University of Infrastructure and Technology,
Kyiv, Ukraine
a.zuenko.a@gmail.com
https://orcid.org/0000-0003-2890-9905

USE OF MULTILAYER LSTM NEURAL NETWORK IN THE PROCESS OF PRINTED TEXTS RECOGNITION

The purpose of the article is to study, analyze and consider general problems and prospects for the development of printed text recognition systems based on the use of neural networks.

The research methodology consists in methods of semantic analysis of this subject area's basic concepts (recognition systems of printed texts). Approaches to the development and operation of recognition systems based on neural networks are considered.

The scientific novelty of the research is the development of its own approach to text recognition based on neural networks, the results of which were used in the development of its own system of print recognition.

Conclusions. The paper considers the well-known views on pattern recognition on the example of printed texts and analyzes modern approaches to the use of neural networks and their training. Taking into account the results of the analysis, the authors decided to develop a system for recognizing the languages of printed texts using learning neural networks.

Keywords: neural networks; neural network training; pattern recognition; printed text recognition system.

29.04.2022