DZIAŁ NAUKOWY

MATEMATYKA STOSOWANA 4, 2003

Ministerstwo Nauki i Szkolnictwa Wyższego

ANDRZEJ BIELECKI (Kraków)

Matematyczne podstawy sztucznych sieci neuronowych

1. WSTĘP

Jedną z zasadniczych cech rozwoju informatyki jest szybki wzrost możliwości obliczeniowych sprzętu komputerowego. Powoduje to zatosowanie informatyki do rozwiązywania coraz bardziej złożonych zagadnień i, co za tym idzie, zapotrzebowanie na coraz większe i bardziej złożone oprogramowanie. Nie zawsze możemy użyć klasycznych metod algorytmicznych do stworzenia tego typu oprogramowania. Po pierwsze, często nie dysponujemy dobrym modelem zależności parametrów wyjściowych systemu od wejściowych — podejście algorytmiczne wymaga znajomości zależności funkcyjnej lub regułowej. Po drugie, nawet jeśli dysponujemy odpowiednim modelem, podejście algorytmiczne może okazać się niemożliwe ze względu na zbyt dużą złożoność, i to zarówno w sensie złożoności przedsięwzięcia na etapie tworzenia oprogramowania, jak i zbyt powolnego działania zaimplementowanego systemu, co jest elementem krytycznym zwłaszcza w systemach czasu rzeczywistego. Dlatego też intensywnie rozwijane są alternatywne w stosunku do klasycznego podejścia algorytmicznego metody sztucznej inteligencji. Do tej grupy metod należą, między innymi, sztuczne sieci neuronowe, czyli układy, których budowa i działanie wzorowane są na biologicznych układach neuronalnych.

Odkrycia neurofizjologów związane z funkcjonalnymi własnościami układu nerwowego umożliwiły na początku lat czterdziestych dwudziestego wieku sformułowanie cybernetycznego modelu neuronu ([44]), który, z pewnymi modyfikacjami, jest powszechnie wykorzystywany do dziś ([1]). Na przełomie lat pięćdziesiątych i sześćdziesiątych skonstruowano pierwsze sztuczne sieci neuronowe — PERCEPTRON i ADALINE, które były układami elektromechanicznymi. Podano też pierwsze algorytmy ustawiania wag synap-

([1]) Więcej informacji na temat neurofizjologicznych podstaw sztucznych sieci neuronowych można znaleźć w [55, str. 6], [35, podrozdział 1.3], [57, podrozdział 1.2] oraz, przede wszystkim, w [56].

tycznych w tego typu układach. Te pionierskie próby zwróciły uwagę na możliwości tego typu układów, ale równocześnie pojawiły się dość istotne ograniczenia. Dziś, z perspektywy wielu lat wiemy, że spowodowane to było z jednej strony brakiem dobrych matematycznych modeli sieci neuronowych, a z drugiej stosowaniem tylko jednego typu sieci — sieci warstowej z neuronami progowymi, jedynej wtedy znanej. Niemniej jednak intensywna krytyka nowego podejścia (przede wszystkim [41]) spowodowała mniej więcej piętnastoletni impas w badaniach nad sieciami neuronowymi. Trwające do dziś ożywienie nastąpiło w połowie lat osiemdziesiątych, kiedy zaproponowano wiele nowych typów sieci neuronowych i algorytmów ich uczenia oraz rozpoczęto matematyczną analizę szucznych układów neuronalnych, co jest głównym tematem niniejszego artykułu.

W rozdziale drugim przedstawiona jest zwięzła taksonomia sieci neuronowych oraz metod ich uczenia. Opisane są również zastosowania sieci neuronowych. Rozdział trzeci poświęcony jest zasadniczym aspektom matematycznej analizy sieci neuronowych. W rozdziale czwartym zaprezentowano kilka przykładów matematycznej analizy sieci neuronowych.

2. SIECI NEURONOWE I ICH ZASTOSOWANIA

W niniejszym rozdziale prezentujemy używane obecnie modele neuronów (podrozdział 2.1), przegląd znanych typów sieci neuronowych oraz metod ich uczenia (podrozdział 2.2), a także zastosowania (podrozdział 2.3).

2.1. Modele neuronu. Podstawową jednostką funkcjonalną układu nerwowego jest neuron. W związku z tym, aby móc zaimplementować sztuczną sieć neuronową, musimy dysponować na tyle dobrym modelem neuronu, by mógł on być punktem wyjścia dla implementacji.

Neuron jest jednostką przetwarzającą sygnały, posiadającą kilka, powiedzmy n, wejść i jedno wyjście. Na wejście neuronu podawane są równocześnie sygnały wejściowe $x_1, \ldots, x_n$ tworzące wektor $\vec{x} = [x_1, \ldots, x_n] \in \mathbb{R}^n$, przy czym m-ta składowa sygnału podawana jest na m-te wejście neuronu. Wyróżniamy trzy zasadnicze typy neuronów — deterministyczne, deterministyczne z histerezą i probabilistyczne.

W neuronach deterministycznych zależność sygnału wyjściowego od wejściowego ma charakter funkcyjny, przy czym funkcja ta zależy również od pewnej rodziny parametrów. Oznaczając przez Ξ zbiór parametrów, neuron możemy przedstawić jako funkcję

$$F : \mathbb{R}^n \times \Xi \ni (\vec{x}, \theta) \mapsto F(\vec{x}, \theta) \in \mathbb{R}.$$

W przypadku neuronów deterministycznych opartych na modelu opracowanym przez McCullocha i Pittsa ([44]) parametrami są, między innymi, wagi wejść neuronu, które tworzą wektor wag $\vec{w} = [w_1, \ldots, w_n]$. Po podaniu na

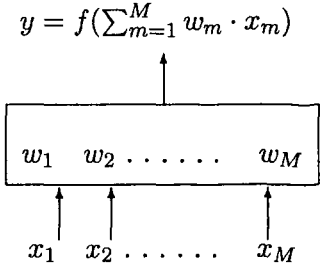
wejście sygnału $\vec{x}$ obliczany jest iloczyn skalarny wektora wag neuronu przez wektor sygnału wejściowego — tzw. całkowite wzbudzenie neuronu

$$s = \vec{w} \circ \vec{x}. \tag{1}$$

Tę liczbę przetwarzamy następnie za pomocą zadanej zależności funkcyjnej — funkcji aktywacji neuronu f, otrzymując sygnał wyjściowy $y = f(s)$ (por. rys. 1). Oryginalny model McCullocha–Pittsa był modelem z funkcją Heaviside'a jako funkcją aktywacji. Wspomniane już ożywienie badań nad sieciami neuronowymi w połowie lat osiemdziesiątych zaowocowało, między innymi, użyciem metod gradientowych do uczenia sieci neuronowych, czyli do znajdowania odpowiednich zestawów parametrów opisujących poszczególne neurony sieci. W związku z tym zaczęto wtedy używać funkcji różniczkowalnych jako funkcji aktywacji neuronu. Najczęściej używane dziś sieci są zbudowane z neuronów deterministycznych z logistyczną funkcją aktywacji

$$f_\beta(s) = \frac{1}{1 + \exp(-\beta s)}. \tag{2}$$

W neuronie tego typu $\Xi = \{\vec{w} \in \mathbb{R}^n, \beta \in \mathbb{R}\}$.



Rys. 1. Model neuronu typu McCullocha–Pittsa

Inną klasą neuronów deterministycznych są neurony radialne (wprowadzone w 1988 roku, [13], [40]), które realizują funkcję zmieniającą się radialnie wokół wybranego centrum $\vec{c} \in \mathbb{R}^n$ ([36], [45, rozdział 5]). Najczęściej używane funkcje aktywacji mają postać $f(r) = \exp(-\frac{r^2}{2\sigma^2})$ oraz $f(r) = g(r^2 - \sigma^2)$, gdzie $r = \|\vec{x} - \vec{c}\|$. Zbiór parametrów neuronów tego typu ma postać $\Xi = \{\vec{c} \in \mathbb{R}^n, \sigma \in \mathbb{R}\}$.

W 1991 roku uogólniono modele bazujące na koncepcji McCullocha i Pittsa, wprowadzając neurony z histerezą ([39, rozdział 6], [58]), w których stan wyjściowy neuronu obliczany w kolejnym kroku zależy nie tylko od całkowitego wzbudzenia, ale również od stanu danego neuronu w poprzednim kroku:

$$\begin{aligned} y(0) &= y(s), \\ y(k+1) &= f(s, y(k)). \end{aligned} \tag{3}$$

Najczęściej używana funkcja aktywacji neuronu z histerezą ma postać

$$y(k+1) = \begin{cases} 1 & \text{dla } s(k+1) > p_g, \\ y(k) & \text{dla } p_d \leq s(k+1) \leq p_g, \\ 0 & \text{dla } s(k+1) < p_d, \end{cases} \tag{4}$$

gdzie p_g, p_d są odpowiednio górnym i dolnym progiem neuronu. W przypadku neuronu z histerezą opisanego wzorem (4) zbiór parametrów jest postaci $\Xi = \{\vec{w} \in \mathbb{R}^n, p_g \in \mathbb{R}, p_d \in \mathbb{R}\}$. Jeszcze dalej idącym uogólnieniem tego typu jest neuron z pamięcią, w którym na stan obecny mają wpływ wszystkie stany poprzednie. Przykładem takiego neuronu jest model wprowadzony przez Aiharę w 1990 roku ([3], [34, podrozdział 3.1]), w którym wyjście neuronu dane jest wzorem

$$y(k+1) = f_\beta\Big(\sum_{j=1}^{n} w_{ij}x_j - \sum_{r=0}^{k} \eta^r y(k-r)\Big), \tag{5}$$

gdzie f jest funkcją aktywacji zależną od rodziny parametrów β, przy czym, na ogół, $\beta \in \mathbb{R}$, natomiast $\eta \in (0,1)$ jest współczynnikiem tłumienia. Jak wynika ze wzoru (5), na bieżący stan neuronu wpływa sygnał wejściowy $\vec{x}$ oraz wszystkie jego poprzednie stany, przy czym, ze względu na obecność tłumienia, im odleglejszy w czasie stan, tym mniejszy ma wpływ na stan obecny. W modelu Aihary zbiorem parametrów jest $\Xi = \{\vec{w} \in \mathbb{R}^n, \eta \in (0,1), \beta\}$.

Ważną grupę neuronów deterministycznych stanowią neurony analogowe, wprowadzone przez Amita w 1989 roku ([2], [34, podrozdział 3.1]). Ich stan opisany jest równaniami różniczkowymi. Oczywiście neurony tego typu mogą być realizowane wyłącznie elektronicznie.

W neuronach deterministycznych, przy ustalonych parametrach, wartości zmiennych niezależnych jednoznacznie wyznaczają sygnał wyjściowy neuronu. Natomiast w neuronach probabilistycznych wartość sygnału wyjściowego jest losowana zgodnie z zadanym rozkładem prawdopodobieństwa. Jako przykład można podać opis działania probabilistycznego neuronu dwustanowego, mogącego na wyjściu przyjmować jedynie wartości $+1$, -1 ([29, podrozdział 5.6]):

$$\begin{aligned} p(y=+1) &= \frac{1}{1+\exp(-2\beta s)}, \\ p(y=-1) &= \frac{1}{1+\exp(+2\beta s)}. \end{aligned} \tag{6}$$

W neuronach opisanych wzorami (6) zbiorem parametrów jest $\Xi = \{\vec{w} \in \mathbb{R}^n, \beta \in \mathbb{R}\}$.

Opierając się na wynikach uzyskanych na gruncie neurofizjologii, wiemy dziś, że wszystkie znane obecnie modele neuronu są zbyt prymitywne, aby

opisać w uproszczony (co jest przy tworzeniu modelu nieuniknione), ale w miarę adekwatny sposób zjawiska zachodzące przy przekazywaniu i przetwarzaniu sygnałów w komórce nerwowej. Już analiza funkcjonalnych własności biologicznego neuronu dokonana przez Waxmana ([53, rozdział 5], [59]) wskazuje na cztery różne obszary przetwarzania sygnałów w biologicznym neuronie: obszar dendryczny, ciało nerwu i początkowy segment aksonu, drzewko aksonalne i wyjście synaptyczne. Obecnie wiadomo, że rzeczywisty neuron jest o wiele bardziej skomplikowany, niż przewiduje koncepcja Waxmana. Wiemy na przykład, że akson może zmieniać kod czasowy z pnia drzewka aksonalnego w kod przestrzenny gałązek tego drzewka.

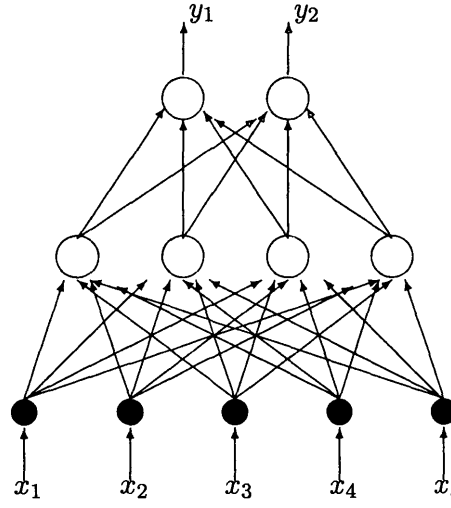
Przetwarzanie impulsów w neuronie odbywa się na błonie komórkowej. Hodgkin i Huxley opisali za pomocą równań różniczkowych typu dyfuzyjnego procesy elektryczne na błonie komórki nerwowej (równania te wraz z ich analizą można znaleźć w [53, rozdział 4 oraz dodatki F, G] i [56, podrozdział 4.5]). Zostały one potwierdzone eksperymentalnie i są dziś uznawane za podstawowe równania neurodynamiki. Model ten jest jednak tak skomplikowany obliczeniowo, że przy dzisiejszych możliwościach systemów komputerowych nie możemy go użyć do modelowania zachowań większych struktur błonowych (na przykład aksonu), a tym bardziej całej komórki.

2.2. Taksonomia sieci neuronowych i metod ich nauki. Sztuczna sieć neuronowa jest systemem neuronów połączonych w ten sposób, że sygnały wyściowe jednych neuronów są podawane na inne neurony jako sygnały wejściowe. Taksonomię sieci neuronowych można przeprowadzić ze względu na różne kryteria. Jednym z kluczowych kryteriów jest rodzaj neuronów, z jakich sieć jest zbudowana. Każdy z typów neuronów wymienionych w poprzednim podrozdziale może służyć jako jednostka funkcjonalna sztucznej sieci neuronowej. Rozróżniamy więc sieci zbudowane na neuronach deterministycznych typu McCullocha–Pittsa (takie używane są najczęściej), typu RBF (*Radial Based Functions*), czyli zbudowane z neuronów radialnych, oraz sieci z histerezą, przy czym badania nad tymi ostatnimi pozostają nieco w tyle w stosunku do innych rodzajów. Używane są również oczywiście sieci probabilistyczne.

Innym, bardzo ważnym kryterium jest podział sieci ze względu na strukturę, jaką tworzą połączone neurony. W tym kontekście wyróżniamy trzy zasadnicze typy sieci.

• *Sieci warstwowe* — w sieciach tego typu propagacja sygnału zachodzi tylko w jednym kierunku. Neurony są ułożone w warstwy, przy czym połączenia występują jedynie między neuronami warstw sąsiednich (por. rys. 2). Sieć warstwowa zawsze ma warstwę wejściową i wyjściową oraz może mieć pewną liczbę warstw ukrytych. Po podaniu na wejście sieci, czyli na wejścia neuronów warstwy wejściowej, sygnału wejściowego $\vec{x}$ każdy neuron sieci jest

pobudzany dokładnie jeden raz, po czym przekazuje swój sygnał wyjściowy na wejścia neuronów następnej warstwy. W wyniku tego procesu pojawiają się sygnały na wyjściach ostatniej, czyli wyjściowej warstwy neuronów. Wektor $\vec{y}$ utworzony z sygnałów wyjściowych neuronów ostatniej warstwy jest odpowiedzią sieci na pobudzenie $\vec{x}$.



Rys. 2. Przykład warstwowej sieci neuronowej

• *Sieci rekurencyjne* — w sieciach tego typu połączenia neuronów tworzą zamknięte pętle. Dzięki temu po podaniu na wejście sieci sygnału $\vec{x}$ neurony są pobudzane wielokrotnie, tworząc dynamiczny przebieg wzbudzeń. Wygenerowana w ten sposób dynamika stanów sieci może po pewnym czasie osiągnąć stan stabilny, tzn. taki, w którym w dwóch kolejnych przebiegach stan żadnego neuronu nie ulegnie zmianie, co implikuje, że w kolejnych przebiegach stan sieci również się nie zmieni. Inną możliwością jest wpadnięcie sieci w oscylacje lub wygenerowanie przebiegów chaotycznych ([34, rozdziały 9 i 10]).

• *Sieci komórkowe* są sieciami o połączeniach lokalnych utworzonych w ten sposób, że neurony umiaszczamy w węzłach pewnej siatki (najczęściej jest to siatka kwadratowa na płaszczyźnie, ale możliwe są też inne topologie) i każdy neuron jest połączony jedynie z neuronami leżącymi w sąsiedztwie o zadanym promieniu. Sąsiedztwo należy tu rozumieć w sensie zgodnym z topologią sieci. Ponieważ neurony leżące na brzegu sieci naruszają symetrię połączeń międzyneuronalnych, często stosuje się topologię torusa, łącząc ze sobą przeciwległe krawędzie sieci. Dynamikę zmian stanów sieci komórkowej opisują równania różniczkowe dla sieci zbudowanych z neuronów analogowych lub równania różnicowe w przeciwnym wypadku.

Chcąc użyć sieci neuronowej do modelowania pewnego zjawiska lub do rozwiązania pewnego problemu, musimy odpowiednio dobrać parametry poszczególnych neuronów. Jest wiele metod realizacji tego zadania. Taksonomia metod uczenia sieci neuronowych przedstawia się następująco.

1. Metody uczenia z nadzorem, zwane też metodami z nauczycielem (*supervised learning*) charakteryzują się tym, że mamy pewne informacje dotyczące poprawnego działania nauczonej sieci. Informacje te wykorzystujemy podczas uczenia sieci neuronowej. W zależności od tego, jaki mają one charakter i w jaki sposób je wykorzystujemy, możemy w metodach z nadzorem wyróżnić następujące strategie.

(a) Metody iteracyjne — są stosowane najczęściej. Polegają na tym, że parametry neuronów zmieniane są krok po kroku, za każdym razem o pewną niewielką wartość. W tym wypadku też mamy do czynienia z dwiema zasadniczymi grupami metod.

(i) Metody uczenia typu *delta* — w tych metodach dysponujemy zbiorem sygnałów wejściowych, dla których znamy odpowiedzi, jakie sieć powinna generować na wyjściu — jest to tzw. ciąg uczący. Podstawą metody uczenia jest porównanie znanej odpowiedzi prawidłowej z faktyczną oraz zmiana wag w ten sposób, aby podczas następnej prezentacji tego samego sygnału wejściowego odpowiedź sieci była bliższa poprawnej. Najczęściej stosowanymi metodami tego typu są metody gradientowe, w których znany i ustalony ciąg uczący jest podstawą zdefiniowania różniczkowalnej funkcji błędu posiadającej minimum globalne dla tych wag sieci, dla których odpowiedzi sieci pokrywają się z odpowiedziami prawidłowymi. Najczęściej używaną funkcją błędu jest funkcja kwadratowa (por. wzór (14)).

(ii) Metody typu probabilistycznego — polegają na losowym wielokrotnym ustawianiu parametrów zgodnie z rozkładami prawdopodobieństwa, które definiujemy na podstawie informacji posiadanych o rozwiązaniu. Przykładem tego typu strategii są algorytmy genetyczne, które są czasem wykorzystywane do nauki sieci neuronowych.

(b) Metody nieiteracyjne — mają obecnie znaczenie marginalne. Możemy ich użyć wtedy, gdy poblem jest prosty, a sieć mała. W takim wypadku czasem możliwe jest znalezienie wartości parametrów (np. wag) przez rozwiązanie układu nierówności. Również dla sieci Hopfielda modelującej pamięć skojarzeniową, przy bardzo restrykcyjnych założeniach dotyczących działania sieci i znając wzorce, które chcemy w sieci zakodować, możemy ustawić wagi zgodnie ze wzorem algebraicznym ([39, str. 45, wzory 3.5, 3.6], [52, str. 44, wzór 4.7].

2. Metody uczenia bez nadzoru charakteryzują się tym, że nie mamy żadnej informacji o ewentualnym rozwiązaniu problemu. Jedyne więc, co sieć może zrobić, to zaklasyfikować sygnały podobne do siebie do tej samej klasy sygnałów wejściowych. Z matematycznego punktu widzenia tak postawiony problem wyznacza *implicite* pewną relację równoważności $\mathcal{S}$ — dwa sygnały wejściowe są ze sobą w relacji $\mathcal{S}$, jeśli są do siebie podobne. Oczywiście dla danego problemu wzmiankowane podobieństwo musi być dobrze zdefiniowane. Wynikiem uczenia sieci będzie podział przestrzeni sygnałów wejściowych X na klasy równoważności. W wyniku otrzymamy przestrzeń ilorazową $X/_{\mathcal{S}}$. Te metody również dzielą się na dwie zasadnicze grupy.

(a) Metody typu konkurencyjnego — polegają na tym, że po zaprezentowaniu sygnału wejściowego znajdowany jest ten neuron, którego wektor wag jest najbardziej podobny do zaprezentownego sygnału, i modyfikowane są wagi tylko tego neuronu w ten sposób, aby oba wektory były jeszcze bardziej do siebie podobne (algorytmy typu *winner takes all*), lub modyfikujemy wagi kilku najbardziej podobnych do zaprezentowanego sygnału neuronów (algorytmy typu *winner takes most*).

(b) Metody oparte na regule Hebba, która mówi, że wielokrotnie pobudzany neuron staje się coraz bardziej czuły na pobudzenie.

Zarówno analiza działania nauczonych sieci neuronowych, jak też analiza metod ich uczenia wymagają użycia zaawansowanych metod matematycznych, przy czym wiele problemów wciąż pozostaje otwartych. Zagadnienia te poruszamy w rozdziale trzecim i czwartym.

2.3. Zastosowania sieci neuronowych. Sieci neuronowe stały się standardem używanym w wielu dziedzinach do rozwiązywania różnych zadań. Tak więc używane są w technice do sterowania ([29, podrozdział 6.3], [35, podrozdział 7.5], [45, podrozdział 4.4], [51, rozdział 5], [61, podrozdział 8.3]), w robotyce ([61, podrozdział 8.4]), do rozpoznawania mowy ([29, podrozdział 6.3], [55, podrozdział 10.1]), w rozpoznawaniu obrazów ([29, podrozdział 6.3], [45, podrozdział 4.1], [61, podrozdział 8.2]), przy kompresji danych ([29, podrozdział 6.3], [45, podrozdział 4.2]), w systemach eksperckich do modelowania bazy wiedzy ([61, podrozdział 8.3]). W medycynie sieci neuronowe są używane głównie do wspomagania diagnostyki. Zbudowano już prototypowe układy neuronalne do diagnostyki chorób serca, skóry czy pleców ([61, podrozdział 8.5]). Ich skuteczność sięgająca ponad 80% poprawnie zdiagnozowanych przypadków jest porównywalna ze skutecznością ekspertów. W ekonomii sieci neuronowe są używane, między innymi, do analizy rynków kapitałowych, modelowania kursu walut i obligacji, analizy kondycji finansowej firmy, prognozowania przepływów gotówkowych ([38,

podrozdział 1.6]). Tak szerokie zastosowania wymagają jednak głębokiego zrozumienia specyfiki sieci neuronowych, ich działania i możliwości. To z kolei rodzi zapotrzebowanie na matematyczne modelowanie sieci neuronowych i bazujących na nich algorytmów.

3. MATEMATYCZNE ASPEKTY SIECI NEURONOWYCH

Impas w badaniach nad sieciami neuronowymi, jaki miał miejsce w latach siedemdziesiątych dwudziestego wieku, wynikał, między innymi, z braku dobrych modeli matematycznych opisujących sieci neuronowe. Wspomniany impas został przełamany, gdy na początku lat osiemdziesiątych podano algorytm uczenia sieci warstwowej metodą propagacji wstecznej błędu. Tak więc stworzenie nowego modelu matematycznego, na którego bazie powstał algorytm uczenia, pozwoliło na wyjście z kryzysu. Od tej pory teoretyczne badanie sieci neuronowych stało się istotnym elementem badania ich specyfiki i możliwości.

Od początku powstania sztucznych sieci neuronowych badania rozwinęły się w dwóch zasadniczych kierunkach tematycznych. Pierwszy z nich związany jest z biologicznymi aspektami sztucznych sieci neuronowych. Próbuje się, mianowicie, kreować modele biologicznych procesów neuronalnych i na tej podstawie wprowadzać algorytmy uczenia sieci bądź tworzyć sieci oparte na nowych paradygmatach ([56]). Z nurtem tym jest ściśle związane modelowanie biologicznych procesów neuronalnych za pomocą sieci neuronowych. Drugi nurt, abstrahując od biologicznych korzeni tematyki, analizuje sieci neuronowe jako układy obliczeniowe i modelujące. Wprowadzane są matematyczne modele opisujące zarówno działanie, jak i uczenie sieci różnego typu. Tytułem przykładu można wymienić opis sieci za pomocą macierzy i grafów ([10], [49], [61]), analizę reprezentacji sygnałów wejściowych ([50]), badanie dynamiki sieci rekurencyjnych ([29, section 2.2], [34], [39]) oraz procesów uczenia sieci różnego typu ([7], [8], [10], [39], [46], [55, rozdział 9]). Przykłady dotyczące tej problematyki podajemy w rozdziale czwartym.

Generalnie w matematycznej analizie sieci neuronowych można wyróżnić trzy grupy tematyczne.

• *Problemy optymalizacyjne.* Chcąc rozwiązać dane zadanie za pomocą sieci neuronowych, chcemy dobrać sieć neuronową optymalną co do liczby neuronów i połączeń między nimi. Istotnym zagadnieniem jest również algorytm zmiany parametrów procesu uczenia, np. współczynnika szybkości uczenia h w równaniu (12). Obecnie problemy te nie są w ogóle rozpracowane matematycznie — dysponujemy jedynie algorytmami opartymi na heurystyce. Oprócz optymalizacji samej sieci pojawia się również problem użycia sztucznych układów neuronalnych do rozwiązania zadań optymalizacyjnych. Ponieważ polegają one na znalezieniu minimum lub maksimum

pewnego funkcjonału, należy zdefiniować funkcjonał, którego minima (lub maksima) lokalne są rozwiązaniami problemu. Sieć neuronowa jest używana do znajdowania tych minimów. Często używa się do tego celu sieci Hopfielda. Szczegółowe i szerokie omówienie tej tematyki można znaleźć w [39].

• *Problemy aproksymacyjne.* Tworzenie modeli polega często na podaniu zależności funkcyjnej między pewnymi zbiorami danych pomiarowych, które na ogół są wielowymiarowe. Sieci neuronowe, zwłaszcza warstwowe, modelujące odwzorowania $\mathbb{R}^k \to \mathbb{R}^n$, są powszechnie używane do rozwiązywania zadań tego typu. Dlatego też intensywnie prowadzone są badania dotyczące możliwości aproksymacyjnych sieci różnego typu. Szereg wartościowych rezultatów uzyskano pod koniec lat osiemdziesiątych i w latach dziewięćdziesiątych ([14], [24], [25], [26], [27], [28], [31], [37]). Zagadnienia te omówione są bardziej szczegółowo w podrozdziale 4.2.

• *Problemy związane z dynamiką.* Analizowana jest zarówno dynamika sieci rekurencyjnych, opisująca przebieg pobudzeń neuronów w odpowiedzi na sygnał wejściowy ([34], [39]), jak i dynamika procesów nauki sieci różnego typu. Badane są zarówno deterministyczne, jak i probabilistycvzne układy dynamiczne [23]. Często stosuje się metody zaczerpnięte z modeli używanych w fizyce ([29], [34]). Przykładowe zagadnienia tego typu omówione są w podrozdziałach 4.1 i 4.3.

4. PRZYKŁADOWE MODELE MATEMATYCZNE

W niniejszym rozdziale zaprezentowane są trzy przykładowe grupy zagadnień związanych z matematyczną analizą sieci neuronowych. Dwa pierwsze podrozdziały prezentują wyniki, które w dziedzinie matematycznych podstaw sieci neuronowych mają już dziś rangę klasycznych. Trzeci podrozdział w dużej części przedstawia „przednią linię frontu" badań naukowych.

4.1. Dynamika sieci Hopfielda. Sieci Hopfielda są sieciami, w których wyjście każdego neuronu połączone jest z wejściami wszystkich neuronów sieci. Jest to więc przykład sieci rekurencyjnej. Od momentu wprowadzenia tego typu sieci w połowie lat osiemdziesiątych dwudziestego wieku badania ich dynamiki przy ustalonych wagach należą do głównego nurtu badań teoretycznych dotyczących sieci rekurencyjnych.

Rozważmy najpierw sieć Hopfielda przy bardzo mocnych założeniach. Dla sieci Hopfielda możemy utworzyć macierz wag $\mathbf{W}$ całej sieci składającą się z wektorów wag poszczególnych neuronów. Jeśli sieć składa się z N neuronów, to $\mathbf{W}$ jest macierzą $N \times N$ o wyrazach rzeczywistych. Przyjmijmy, że rozważana sieć Hopfielda nie ma samosprzężeń i jest symetryczna, czyli $w_{ii} = 0$, $w_{ij} = w_{ji}$, $i, j \in \{1, \ldots, N\}$. Niech ponadto będzie ona siecią binarną unipolarną, czyli wszystkie składowe wektora sygnału wyjściowego

w każdym kroku, jak i początkowego sygnału pobudzającego, mogą jedynie przyjmować wartości 0 lub 1. Załóżmy również, że sieć działa w trybie synchronicznym, tzn. czas jest modelowany przez zbiór liczb naturalnych i wszystkie neurony są pobudzane w kolejnych krokach równocześnie. Niech kolejne pobudzenia i-tego neuronu przebiegają według schematu

$$y_i(k+1) = f(s_i(k)), \tag{7}$$

gdzie $s_i(k) = \sum_{j=1}^{N} w_{ij} \cdot y_j(k)$, natomiast funkcja aktywacji f jest dana wzorem

$$y_i(k+1) = \begin{cases} 1 & \text{dla } s_i(k) > 0, \\ y_i(k) & \text{dla } s_i(k) = 0, \\ 0 & \text{dla } s_i(k) < 0. \end{cases} \tag{8}$$

Przy powyższych założeniach zachodzi następujące twierdzenie ([52, rozdział 4]).

TWIERDZENIE 4.1.1. *Sieć Hopfielda spełniająca powyższe założenia dla dowolnego pobudzenia początkowego osiąga stan stabilny w skończonej liczbie kroków. Oznacza to, że dla każdego* $\vec{y}(0) \in \mathbb{R}^N$ *istnieje liczba naturalna* K *taka, że* $y_i(K+1) = y_i(K)$ *dla każdego* $i \in \{1, \ldots, N\}$.

Dowód. W najogólniejszym przypadku każdy rodzaj neuronu bazujący na modelu McCullocha–Pittsa może mieć tzw. próg, czyli stałą dodawaną do całkowitego wzbudzenia obliczanego według wzoru (1). Jednak w rozważaniach teoretycznych wygodnie jest przyjąć, że neuron ma dodatkowe wejście z wagą równą progowi, na które zawsze podajemy jedynkę jako odpowiednią składową sygnału wejściowego. Bez szkody dla ogólności możemy więc ograniczyć się do neuronów bez progu.

Zdefiniujmy funkcję $E : \mathbb{R}^N \to \mathbb{R}$ w następujący sposób:

$$E(\vec{y}) := -\frac{1}{2} \cdot \vec{y}^T \cdot \mathbf{W} \cdot \vec{y} = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij} y_i y_j. \tag{9}$$

Stany sieci zmieniają się synchronicznie. Można jednak rozłożyć każdy synchroniczny krok na N podkroków (czyli tyle, ile jest neuronów). Obliczmy zmianę wartości funkcji E w jednym podkroku. Przyjmujemy więc, że może się zmienić jedynie stan m-tego neuronu, natomiast stany pozostałych neuronów są ustalone. Wówczas

$$E_m = -\frac{1}{2} y_m \sum_{i, i \neq m} w_{im} y_i.$$

Można było przyjąć, bez szkody dla ogólności, że sumowanie przebiega po $i \neq m$, bo zgodnie z założeniami $w_{mm} = 0$. Ponieważ zmienia się tylko stan m-tego neuronu, więc $y_i(k+1) = y_i(k)$ dla $i \neq m$ i wtedy zmiana wartości

funkcji E w jednym podkroku wyraża się wzorem

$$\Delta E_m := E_m(k+1) - E_m(k) = -\frac{1}{2}(y_m(k+1) - y_m(k)) \cdot \sum_{i,i\neq m} w_{im}y_i$$

$$= -\frac{1}{2}\Delta y_m \cdot \sum_{i,i\neq m} w_{im}y_i.$$

Jeśli $\Delta y_m = 1$, czyli stan neuronu zmienił się z 0 na 1, to zgodnie ze wzorem (8), $s_i(k) := \sum_{j=1}^{N} w_{ij} \cdot y_j(k) > 0$. Jeśli natomiast $\Delta y_m = -1$ (zmiana stanu z 1 na 0), to całkowite wzbudzenie jest ujemne, czyli $s_i(k) := \sum_{j=1}^{N} w_{ij} \cdot y_j(k) < 0$. W obu wypadkach $\Delta E_m < 0$. Całkowita zmiana energii w jednym kroku $\Delta E := \sum_{m=1}^{N} \Delta E_m$ jest równa zeru wtedy i tylko wtedy, gdy $\Delta y_m = 0$ dla każdego $m \in \{1, \ldots, N\}$, co oznacza, że stan całej sieci nie uległ zmianie, czyli sieć osiągnęła stan stabilny. W przeciwnym razie zmiana wartości funkcji E jest ujemna. Wykazaliśmy więc, że funkcja E jest malejąca wzdłuż trajektorii układu dynamicznego generowanego przez sieć Hopfielda.

Zbiór wartości funkcji E jest ograniczony, bo

$$|E(\vec{y})| \leq \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} |w_{ij}|.$$

Zatem ciąg wartości $E(\vec{y}(k))$ dąży do $\alpha \in \mathbb{R}$, jako ciąg malejący i ograniczony.

Zbiór wartości funkcji E jest skończony, ponieważ $y_i \in \{0, 1\}$. Oznacza to, że istnieje stała $c > 0$ taka, że jeśli $\Delta E > 0$, to $\Delta E > c$. Zatem zmiany wartości funkcji E nie mogą być dowolnie małe, co oznacza, że funkcja E osiągnie minimum lokalne w skończonej liczbie kroków, a to bezpośrednio implikuje osiągnięcie przez sieć Hopfielda w skończonej liczbie kroków stanu stabilnego, co kończy dowód. ∎

Dowód twierdzenia 4.1.1 opierał się w głównej mierze na zdefiniowaniu funkcji Lapunowa E dla układu dynamicznego opisującego dynamikę sieci Hopfielda. W przypadku matematycznej analizy sieci Hopfielda jest to postępowanie standardowe. Analogicznie postępuje się dla sieci pracujących w trybie asynchronicznym zbudowanych z neuronów dwustanowych oraz dla modeli z neuronem posiadającym ciągłą funkcję aktywacji dla czasu zarówno dyskretnego, jak i ciągłego (por. [29, rozdział 3], [35, rozdział 4], [39, rozdziały 3, 4], [52, rozdział 4], [61, rozdział 5].

Sieci Hopfielda mogą również generować inne typy dynamiki. Przy odpowiednich założeniach możliwe są oscylacje oraz dynamika chaotyczna nawet w bardzo prostych sieciach (por. [34, rozdział 9]).

4.2. Właściwości aproksymacyjne sieci warstwowych. Wiele modeli opisujących różne zjawiska ma charakter aproksymacyjny. Dlatego po-

wszechne użycie sieci neuronowych do modelowania zwłaszcza nieliniowych zjawisk zrodziło zapotrzebowanie na analizę aproksymacyjnych własności sieci neuronowych.

W niniejszym podrozdziale zaprezentowano i omówiono kilka znanych twierdzeń dotyczących aproksymacyjnych własności sieci warstwowych zbudowanych z neuronów bazujących na modelu McCullocha–Pittsa. Twierdzenia zostały zaprezentowane bez dowodów, które są zbyt długie, aby mogły być przedstawione w niniejszej publikacji. Znaleźć je można w oryginalnych artykułach — [4], [14], [24], [25], [26], [27], [28], [37]. W polskojęzycznej literaturze omówienie tych problemów (również bez dowodów) można znaleźć w [36], [38, podrozdział 1.4], [61, paragraf 4.4.2].

Często punktem wyjścia prac dotyczących aproksymacyjnych własności sieci warstwowych jest twierdzenie Kołmogorowa o reprezentacji funkcji wielu zmiennych za pomocą funkcji jednej zmiennej [33]:

TWIEDZENIE 4.2.1. *Dla dowolnego naturalnego* $n \geq 2$ *istnieje taka rodzina* $\{\psi_{pq}\}$, $p \in \{1, \ldots, n\}$, $q \in \{1, \ldots, 2n+1\}$, *funkcji rzeczywistych ciągłych na odcinku* $[0, 1]$, *że na* n*-wymiarowej kostce jednostkowej* I^n *dowolna ciągła funkcja* $f : I^n \to \mathbb{R}$ *może być przedstawiona w postaci*

$$f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} \chi_q\Big(\sum_{p=1}^{n} \psi_{pq}(x_p)\Big),$$

gdzie $\{\chi_q\}$, $q \in \{1, \ldots, 2n+1\}$, *jest rodziną funkcji rzeczywistych ciągłych.*

Opierając się na twierdzeniu Kołmogorowa, Hecht-Nielsen zaproponował do aproksymacji ciągłej funkcji $f : I^n \to \mathbb{R}$ sieć z jedną warstwą ukrytą ([24]). Była to więc sieć o strukturze warstwowej jak na rys. 2, z tym że sieć zaproponowana przez Hecht-Nielsena ma w warstwie wyjściowej jeden neuron. Pierwsza warstwa składająca się z n jednostek wprowadza sygnał wejściowy $\vec{x} = [x_1, \ldots, x_n]$ na wejścia wszystkich neuronów warstwy ukrytej. Warstwa ukryta składa się z $2n+1$ neuronów, w których sygnały wyjściowe dane są wzorami

$$u_q = \sum_{p=1}^{n} \lambda_{pq} \psi_q(x_p),$$

gdzie $\{\lambda_{pq}\}$, $q \in \{1, \ldots, 2n+1\}$, $p \in \{1, \ldots, n\}$, jest rodziną stałych rzeczywistych. Pojedynczy neuron warstwy wyjściowej generuje sygnał wyjściowy według wzoru

$$y = \sum_{q=1}^{2n+1} \phi(u_q),$$

gdzie ϕ jest pewną ciągłą funkcją nieliniową.

Powyższa propozycja ma charakter egzystencjalny, gdyż nie podano sposobu doboru funkcji ψ_q i ϕ. Nieco bardziej konstruktywny rezultat uzyskał

Cybenko ([14]). Rozważmy mianowicie warstwową sieć neuronową z jedną warstwą ukrytą składającą się z N neuronów, każdy z funkcją aktywacji $f : \mathbb{R} \to \mathbb{R}$ będącą funkcją ciągłą, rosnącą i spełniającą warunki graniczne

$$\lim_{s\to-\infty} f(s) = 0, \qquad \lim_{s\to\infty} f(s) = 1 \tag{10}$$

i z progiem γ_k, $k \in \{1, \ldots, N\}$, przy czym warstwa wyjściowa jest pojedynczym neuronem liniowym. Niech ponadto sygnał wejściowy będzie d-wymiarowy, czyli $\vec{x} \in \mathbb{R}^d$. Taka sieć generuje sygnał wyjściowy $y \in \mathbb{R}$ zgodnie ze wzorem

$$y_N(\vec{x}) = \sum_{k=1}^{N} v_k \cdot f(\vec{w}_k \circ \vec{x} + \gamma_k), \tag{11}$$

gdzie $\vec{w}_k$ jest wektorem wag k-tego neuronu warstwy ukrytej, natomiast $\vec{v} = [v_1, \ldots, v_N]$ jest wektorem wag neuronu wyjściowego (symbol $\circ$ oznacza iloczyn skalarny wektorów).

TWIEDZENIE 4.2.2. *Dla ustalonej funkcji f, dowolnej funkcji ciągłej $g : I^d \to \mathbb{R}$ i dowolnego $\varepsilon > 0$ istnieje liczba naturalna N oraz parametry $\vec{v}$, $\vec{w}_k$, γ_k takie, że dla dowolnego $\vec{x} \in I^d$*

$$|g(\vec{x}) - y_N(\vec{x})| < \varepsilon.$$

Innymi słowy, za pomocą sieci neuronowej z jedną warstwą ukrytą, posiadającą neurony np. z sigmoidalną funkcją aktywacji daną wzorem (2) i z pojedynczym neuronem liniowym w warstwie wyjściowej, można na skończenie wymiarowej kostce aproksymować każdą funkcję ciągłą z dowolną dokładnością. Twierdzenie Cybenki podaje co prawda postacie funkcji aktywacji, jednak nie mówi, w jaki sposób dobierać parametry. Nie estymuje też dokładności oszacowania. Estymację dokładności oszacowania podał Barron ([4], [36, twierdzenie 11.2]) dla sieci działającej według schematu (11) z ciągłą, rosnącą funkcją aktywacji f w warstwie ukrytej, spełniającą warunki (10). Wyszczególniamy dwa warunki na funkcje $g : \mathbb{R}^d \to \mathbb{R}$:

(C1) Funkcja g posiada reprezentację całkową postaci

$$g(x) = \int_{\mathbb{R}^d} e^{i\cdot\omega^T\cdot x} \cdot \widetilde{g}(\omega)\, d\omega,$$

gdzie $\widetilde{g}$ jest funkcją zmiennej zespolonej oraz $i^2 = -1$.

(C2) Funkcja $\sqrt{\omega^T\omega} \cdot \widetilde{g}(\omega)$ jest całkowalna w $\mathbb{R}^d$, co implikuje, że

$$C_f := \int_{\mathbb{R}^d} \sqrt{\omega^T \cdot \omega} \cdot |\widetilde{g}(\omega)|\, d\omega < \infty.$$

Oznaczmy przez $\Im_d$ rodzinę funkcji $g : \mathbb{R}^d \to \mathbb{R}$ spełniających warunki (C1) i (C2).

TWIEDZENIE 4.2.3. *Ustalmy funkcję* $f : \mathbb{R} \to \mathbb{R}$ *spełniającą powyższe warunki. Wybierzmy kulę* $B_r := \{x \in \mathbb{R}^d : \|x\| < r,\ r > 0\}$. *Załóżmy, że* $g \in \mathfrak{F}_d$. *Wtedy dla każdego naturalnego* $N \geq 1$ *istnieje taki zestaw parametrów* $\vec{v} = [v_1, \ldots, v_N]$, $\vec{w_k}$, γ_k, *gdzie* $k = 1, \ldots, N$, *dla którego sieć* (11) *generuje funkcję* $y_N : \mathbb{R}^d \to \mathbb{R}$ *spełniającą nierówność*

$$\int_{B_r} (g(\vec{x}) - y_N(\vec{x}))^2 < \frac{2 \cdot r \cdot C_f}{N}.$$

Dowód twierdzenia Barrona oparty jest na mocnym prawie wielkich liczb. Twierdzenie można uogólnić na obszary aproksymacji inne niż kula, a w całce można stosować miary inne niż miara Lebesgue'a.

Nieco inne podejście do problemów aproksymacji polega na tym, aby sieć neuronowa modelowała szereg Fouriera ([36]). Odrębnym zagadnieniem są możliwości aproksymacyjne sieci radialnych ([31], [47]).

4.3. Analiza dynamiki procesu uczenia sieci warstwowych. W ostatnich latach, w związku z upowszechnieniem komputerów i szybkim rozwojem ich możliwości obliczeniowych, daje się zaobserwować duże zainteresowanie matematyką obliczeniową, a w szczególności szeroko rozumianymi metodami numerycznymi, zwłaszcza w kontekście ich wykorzystania algorytmicznego. Do tej klasy zagadnień należy dynamika numeryczna. Badane są na przykład własności dyskretnych układów dynamicznych otrzymywanych z układów ciągłych. Wiąże się to z faktem, że dysponujemy szerokim wachlarzem skutecznych metod analizy układów ciągłych (tzw. potoków), natomiast implementować możemy oczywiście jedynie układy dyskretne (czyli kaskady). Podstawowym pytaniem jest, przy jakich założeniach jakościowe własności potoków są zachowywane przy danej metodzie dyskretyzacji. W tym kontekście badana jest stabilność ([32]), struktura punktów siodłowych ([1], [5]), rozmaitości niezmiennicze ([12], [16]), czy też niezmienniki algebraiczno-topologiczne ([42], [54]). Badania dotyczą zarówno aspektów lokalnych, jak i globalnych ([6], [17], [18], [19], [20], [22]). Dyskutowane są aspekty numeryczne ([21]).

Okazuje się, że zagadnienia te są ściśle związane z niektórymi metodami uczenia sieci neuronowych (por. podrozdział 2.2). Otóż uczenie sieci warstwowych metodami gradientowymi można potraktować jako proces dynamiczny realizujący algorytm numeryczny. Dlatego też wspomniane wyniki można zastosować do analizy procesu uczenia.

W niniejszym podrozdziale najpierw krótko prezentujemy uczenie warstwowych sieci neuronowych metodą gradientową największego spadku z propagacją wsteczną błędu. Następnie analizujemy uczenie sieci liniowych. Uzyskane wyniki są powszechnie znane, są tu jednak analizowane od strony teorii układów dynamicznych, co w tym kontekście jest raczej rzadko spo-

tykane. Natomiast wyniki dotyczące sieci słabo nieliniowych i nieliniowych zostały uzyskane niedawno i są w większości wynikami częściowymi, wymagającymi uogólnień.

4.3.1. *Uczenie warstwowych sieci neuronowych metodą gradientową największego spadku z propagacją wsteczną błędu.* W celu przeprowadzenia analizy procesu uczenia warstwowej sieci neuronowej metodą największego spadku gradientu z propagacją wsteczną błędu przyjmijmy, że dany jest ustalony ciąg uczący $\{\vec{x}(n), \vec{z}(n)\}_{n=1,2,\ldots,N}$ i warstwowa sieć neuronowa, przy czym funkcja aktywacji f każdego neuronu jest różniczkowalna. Niech ponadto prawo opisujące proces uczenia będzie zdefiniowane poprzez metodę gradientową największego spadku (por. np. [35, rozdz. 2.4.1], [45, rozdz. 2.5.2]), która daje następujący wzór na zmianę wag sieci:

$$w_\iota(k+1) = w_\iota(k) - h \cdot \frac{\partial E(\mathbf{w}(k))}{\partial w_\iota}, \tag{12}$$

gdzie parametr k numeruje krok iteracji w procesie uczenia oraz $\mathbf{w} = \{w_\iota\}_{\iota \in \Gamma}$ jest wektorem wszystkich wag w sieci. Jak wiadomo, powyższy wzór określa ciąg kolejnych punktów leżących na łamanej Eulera przybliżającej rozwiązanie równania różniczkowego

$$\dot{\mathbf{w}} = -\operatorname{grad} E(\mathbf{w}). \tag{13}$$

W związku z tym proces uczenia, opisany wzorem (12), możemy badać, stosując elementy teorii równań różniczkowych i układów dynamicznych.

Zajmiemy się obecnie najczęściej używaną funkcją kosztu, jaką jest następująca funkcja kwadratowa:

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \sum_{t=1}^{T_R} [y_{R,t}(n) - z_t(n)]^2, \tag{14}$$

gdzie $y_{R,t}(n) = f_{R,t}(\vec{w}_{R,t} \circ \vec{x}_{R,t}(n))$, $f_{R,t}$ jest różniczkowalną funkcją aktywacji t-tego neuronu w R-tej warstwie, sygnał wyjściowy przedostatniej warstwy jest podawany na ostatnią warstwę jako jej sygnał wejściowy $\vec{x}_{R,t}(n) = \vec{y}_{R-1}$, $\vec{w}_{R,t} = [w_{R,t,1}, \ldots, w_{R,t,m}]$ oraz przyjęto następującą konwencję co do wskaźników:

- $n = 1, \ldots, N$ — numer wektora w ciągu uczącym — wskaźnik górny w nawiasie,
- $r = 1, \ldots, R$ — numer warstwy sieci,
- $t = 1, \ldots, T_r$ — numer neuronu w r-tej warstwie,
- $m = 1, \ldots, M_r$ — numer wejścia w neuronie r-tej warstwy.

Oczywiście, jak już wspomniano, $T_{r-1} = M_r$.

Przyjęto więc następującą konwencję numerowania sygnałów wyjściowych i wag:

- $y_{r,t}(k)$ — sygnał wyjściowy t-tego neuronu w r-tej warstwie w k-tym kroku uczenia,
- $w_{r,t,m}(k)$ — m-ta waga w t-tym neuronie w r-tej warstwie w k-tym kroku uczenia,
- $\vec{w}_{r,t}$ — wektor wag t-tego neuronu r-tej warstwy.

Tak więc na przykład $y_{3,5}(10)$ oznacza sygnał wyjściowy piątego neuronu w trzeciej warstwie w dziesiątym kroku uczenia.

Jak widzimy, funkcja E zależy explicite jedynie od wag neuronów warstwy wyjściowej oraz od wartości sygnałów podawanych na warstwę wyjściową. Te sygnały z kolei, stanowiące sygnały wyjściowe warstwy przedostatniej, zależą w sposób bezpośredni od wag neuronów przedostatniej warstwy oraz od sygnałów podawanych na tę warstwę itd., przy czym sygnały wejściowe podawane na pierwszą warstwę sieci są określone przez ustalenie ciągu uczącego. W związku z tym funkcja E w istotny sposób zależy od wag wszystkich neuronów sieci.

Aby efektywnie stosować wzór (12), musimy umieć obliczyć $\frac{\partial E}{\partial w_\iota}$, gdzie w_ι przebiega wszystkie wagi neuronów w sieci. Obliczmy zatem zmiany wag neuronów warstwy wyjściowej. Sygnał wyjściowy sieci zależy bezpośrednio od tych wag, mamy więc

$$y_{R,t}(k) = f_{R,t}\Big(\sum_{m=1}^{M_R} x_{R,m}(k) \cdot w_{R,t,m}(k)\Big),$$

gdzie $x_{R,m}(k) = y_{R-1,m}(k)$. Na podstawie wzoru (12) reguła zmiany wag dla neuronów warstwy wyjściowej przyjmuje postać

$$w_{R,t,m}(k+1) = w_{R,t,m}(k) - h \cdot \frac{\partial E(\mathbf{w}(k))}{\partial w_{R,t,m}}.$$

Uwzględniając wprowadzoną definicję funkcji kosztu, możemy obliczyć (przyjęto konwencję, że wskaźniki nieprimowane są bieżące):

$$\begin{aligned}
\frac{\partial E}{\partial w_{R,t',m'}} &= \frac{\partial}{\partial w_{R,t',m'}} \frac{1}{2} \sum_{n=1}^{N} \sum_{t=1}^{T_R} \Big[f_{R,t}\Big(\sum_{m=1}^{M_R} y_{R-1,m}(n) \cdot w_{R,t,m}^{(n)}\Big) - z_t(n)\Big]^2 \\
&= \frac{1}{2} \sum_{n=1}^{N} \frac{\partial}{\partial w_{R,t',m'}} \sum_{t=1}^{T_R} \Big[f_{R,t}\Big(\sum_{m=1}^{M_R} y_{R-1,m}(n) \cdot w_{R,t,m}(n)\Big) - z_t(n)\Big]^2 \\
&= \sum_{n=1}^{N} [y_{R,t'}(n) - z_{t'}(n)] \cdot f'_{R,t'}(s_{R,t'}(n)) \cdot y_{R-1,m'}(n),
\end{aligned}$$

gdzie $s_{R,t'}(n)$ jest całkowitym wzbudzeniem t'-tego neuronu warstwy R-tej w n-tym kroku, a prim przy funkcji f oznacza jej pochodną.

Aby obliczyć zmiany wag neuronów warstw ukrytych, musimy zastosować różniczkowanie funkcji złożonej, gdyż sygnał wyjściowy sieci zależy od

wag neuronów warstw ukrytych (tzn. warstw numerowanych wskaźnikiem mniejszym od R) w sposób pośredni. Dla warstwy przedostatniej mamy więc

$$w_{R-1,t,m}(k+1) = w_{R-1,t,m}(k) - h \cdot \frac{\partial E}{\partial w_{R-1,t,m}},$$

$$\frac{\partial E}{\partial w_{R-1,t,m}} = \sum_{n=1}^{N} \sum_{t=1}^{T_{R-1}} \frac{\partial E}{\partial y_{R-1,t}(n)} \cdot \frac{\partial y_{R-1,t}(n)}{\partial w_{R-1,t,m}}.$$

Dalsze obliczenia przebiegają analogicznie jak dla neuronów warstwy wyjściowej.

Dla kolejnych warstw postępujemy podobnie, różniczkując funkcję wielokrotnie złożoną. Opisany sposób modyfikacji wag nosi nazwę propagacji wstecznej błędu ([29, podrozdziały 6.1, 6.2], [43, podrozdział 6.2], [46], [45, podrozdział 2.3], [61, podrozdział 4.4]).

4.3.2. *Liniowe sieci neuronowe.* Warstwowa sieć liniowa składa się z neuronów z liniową funkcją aktywacji f. Ponieważ $f : \mathbb{R} \to \mathbb{R}$, więc w najogólniejszym przypadku

$$f(s) = \alpha \cdot s = \alpha \cdot \vec{x} \circ \vec{w} = \vec{x} \circ (\alpha \cdot \vec{w}) = \vec{x} \circ \vec{\omega},$$

gdzie $\vec{\omega} = \alpha \cdot \vec{w}$, $\alpha \in \mathbb{R}$. Tak więc stałą α można zakodować w wektorze wag neuronu i w związku z tym wystarczy się ograniczyć do rozważenia przypadku $f = \mathrm{id}_{\mathbb{R}}$. Z drugiej strony, ponieważ w sieci warstwowej poszczególne warstwy realizują złożenie funkcji, a złożenie funkcji liniowych jest funkcją liniową, wystarczy się ograniczyć do sieci jednowarstwowych. Jednakże w takich sieciach neurony nie są ze sobą połączone, więc równanie na zmianę wag w procesie uczenia separuje się na tyle niezależnych równań, ile jest neuronów w sieci. Wystarczy więc, bez szkody dla ogólności, ograniczyć się do rozważenia pojedynczego neuronu. W takim przypadku sygnał wyjściowy y będzie miał tylko wskaźnik n w ciągu uczącym, w związku z czym funkcja kosztu będzie miała postać

$$E(w_1, \ldots, w_M) = \frac{1}{2} \sum_{n=1}^{N} [y(n) - z(n)]^2,$$

gdzie

$$y(n) = \sum_{m=1}^{M} x_m(n) w_m.$$

Obliczmy wartość prawej strony równania definiującego proces uczenia:

$$\frac{\partial E(w_1, \ldots, w_M)}{\partial w_{m'}} = \frac{\partial}{\partial w_{m'}} \sum_{n=1}^{N} \Big[\sum_{m=1}^{M} x_m(n) w_m - z(n) \Big]^2$$

Oznaczając

$$H(n) = \sum_{m=1}^{M} x_m(n) w_m - z(n),$$

otrzymujemy

$$\begin{aligned}\frac{\partial E(w_1, \ldots, w_M)}{\partial w_{m'}} &= \sum_{n=1}^{N} H(n) \cdot \frac{\partial (y(n) - z(n))}{\partial w_{m'}} = \sum_{n=1}^{N} H(n) \cdot \frac{\partial y(n)}{\partial w_{m'}} \\ &= \sum_{n=1}^{N} H(n) \cdot \frac{\partial (\sum_{m=1}^{M} x_m(n) w_m)}{\partial w_{m'}} = \sum_{n=1}^{N} H(n) \cdot x_{m'}(n) \\ &= \sum_{n=1}^{N} x_{m'}(n) \cdot \Big[\Big(\sum_{m=1}^{M} x_m(n) w_m \Big) - z(n) \Big].\end{aligned}$$

Dla neuronu liniowego równanie opisujące proces uczenia jest więc równaniem różniczkowym liniowym niejednorodnym, które można przedstawić w postaci

$$\frac{d\vec{w}}{dt} = -(\mathbf{A} \cdot \vec{w} - \mathbf{B}),$$

gdzie $\mathbf{A}$ jest macierzą Grama:

$$\mathbf{A} = G(\overline{x}_1, \ldots, \overline{x}_M), \tag{15}$$

przy czym przez $\overline{x}_m$, $m = 1, \ldots, M$, oznaczono wektor N-wymiarowy, którego składowymi są kolejne sygnały wejściowe podawane na m-te wejście neuronu.

Tak więc, jeśli neuron ma M wejść, a ciąg uczący składa się z N wektorów $\vec{x}^{(n)}$, to macierz Grama $\mathbf{A}$ jest macierzą kwadratową wymiaru $M \times M$, przy czym jej wyrazy są dane wzorem

$$a_{ij} = \overline{x}_i \circ \overline{x}_j.$$

Wektor $\mathbf{B}$ jest natomiast wektorem M-wymiarowym o współrzędnych

$$b_m = \overline{x}_m \circ \vec{z}, \qquad \vec{z} = (z(1), \ldots, z(N)). \tag{16}$$

Dla całej sieci liniowej proces uczenia można zapisać w postaci macierzowej

$$\frac{d\mathbf{W}}{dt} = -(\mathbf{A} \cdot \mathbf{W} - \mathbf{B}), \tag{17}$$

gdzie $\mathbf{W}$ jest macierzą, której t-tą kolumnę tworzy wektor $\vec{w}_t$, natomiast $\mathbf{B}$ jest macierzą złożoną z elementów $b_{m,t} = \overline{x}_m \circ \vec{z}_t$.

PRZYKŁAD. Rozważmy prosty przykład uczenia jednego neuronu z dwoma wejściami i ciągiem uczącym składającym się z trzech elementów. Zgodnie z otrzymanym wzorem mamy

$$\frac{dw_1}{dt} = -\sum_{n=1}^{3} x_1(n) \cdot [(x_1(n) w_1 + x_2(n) w_2) - z(n)]$$

$$
\begin{aligned}
&= -(x_1(1)x_1(1)w_1 + x_1(1)x_2(1)w_2 - x_1(1)z(1) + x_1(2)x_1(2)w_1 \\
&\quad +x_1(2)x_2(2)w_2 - x_1(2)z(2) + +x_1(3)x_1(3)w_1 \\
&\quad +x_1(3)x_2(3)w_2 - x_1(3)z(3)) \\
&= -[(x_1(1)x_1(1) + x_1(2)x_1(2) + x_1(3)x_1(3))w_1 \\
&\quad +(x_1(1)x_2(1) + x_1(2)x_2(2) + x_1(3)x_2(3))w_2 \\
&\quad -x_1(1)z(1) - x_1(2)z(2) - x_1(3)z(3)] \\
&= -[(\overline{x}_1 \circ \overline{x}_1) \cdot w_1 + (\overline{x}_1 \circ \overline{x}_2) \cdot w_2 - (\overline{x}_1 \circ \vec{z})].
\end{aligned}
$$

Analogicznie oblicza się $\frac{dw_2}{dt}$. Reasumując, otrzymaliśmy układ równań

$$
\begin{aligned}
\frac{dw_1}{dt} &= -[(\overline{x}_1 \circ \overline{x}_1) \cdot w_1 + (\overline{x}_1 \circ \overline{x}_2) \cdot w_2 - (\overline{x}_1 \circ \vec{z})], \\
\frac{dw_2}{dt} &= -[(\overline{x}_2 \circ \overline{x}_1) \cdot w_1 + (\overline{x}_2 \circ \overline{x}_2) \cdot w_2 - (\overline{x}_2 \circ \vec{z})],
\end{aligned}
$$

co można zapisać jako

$$
\frac{d\vec{w}}{dt} = -(\mathbf{A} \cdot \vec{w} - \mathbf{B}),
$$

gdzie

$$
\mathbf{A} = \begin{pmatrix} \overline{x}_1 \circ \overline{x}_1 & \overline{x}_1 \circ \overline{x}_2 \\ \overline{x}_2 \circ \overline{x}_1 & \overline{x}_2 \circ \overline{x}_2 \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} \overline{x}_1 \circ \vec{z} \\ \overline{x}_2 \circ \vec{z} \end{pmatrix}.
$$

Przejdźmy do analizy stabilności procesu uczenia liniowej sieci neuronowej. W tym celu wystarczy zbadać asymptotyczną stabilność równania jednorodnego (por. [15, str. 91, wniosek 2]):

$$
\frac{d\vec{w}}{dt} = -\mathbf{A} \cdot \vec{w}.
$$

Można łatwo pokazać (na przykład korzystając z kryterium Hurwitza [15, str. 116]), że układ dynamiczny, modelujący proces uczenia sieci liniowej, generowany przez równanie (17) jest asymptotycznie stabilny oraz, co za tym idzie, globalnie asymptotycznie stabilny pod warunkiem, że wektory $\overline{x}_1, \ldots, \overline{x}_M$ są liniowo niezależne. Równanie (17) ma wtedy dokładnie jeden punkt stały.

Rozważmy topologiczne sprzężenie dyskretnego układu dynamicznego $(\phi_h, \mathbb{R}^n)$ generowanego przez dyskretyzację równania różniczkowego (17) opisującego proces nauki sieci liniowej i układu $(\psi_h, \mathbb{R}^n)$, generowanego przez rozwiązanie tego równania metodą Eulera dla dostatecznie małego kroku czasowego h. Z twierdzenia Grobmana–Hartmana ([48, str. 60, tw. 4.1]) wynika, że w pobliżu hiperbolicznego punktu stałego układy ϕ_h i ψ_h są lokalnie topologicznie sprzężone przez pewien homeomorfizm H. Można pokazać, że globalnym homeomorfizmem sprzęgającym jest odwzorowanie $F : \mathbb{R}^n \to \mathbb{R}^n$

zdefiniowane następująco:

$$F(x) = \psi_h^n(H(\phi_h^n(x))),$$

gdzie n jest minimalną liczbą naturalną taką, że punkt $\phi(x, nh)$ należy do otoczenia punktu stałego, w którym układy ϕ_h i ψ_h są sprzężone (globalne sprzężenie układów ϕ_h i ψ_h wynika również z twierdzenia Fečkana, cytowanego w następnym podrozdziale — tw. 4.3.1). Istnienie globalnego homeomorfizmu sprzęgającego implikuje, że dynamika dyskretyzacji różniczkowego modelu uczenia sieci liniowej i metody Eulera, która jest implementowana na komputerze, jest taka sama.

Globalną asymptotyczną stabilność udowodniliśmy dla przypadku, w którym wektory $\bar{x}_1, \ldots, \bar{x}_M$, utworzone przez ciąg uczący, są liniowo niezależne. Okazuje się, że w praktyce można ograniczyć rozważania do takiego przypadku. Wykażemy to na gruncie rozważań o własnościach typowych. Podamy najprostszą spośród używanych definicji własności typowej [15, str. 107] (nieco inne podejście do tego problemu można znaleźć np. w [60, str. 99]).

DEFINICJA 4.3.1. Niech X będzie przestrzenią topologiczną. Mówimy, że pewna własność jest *typowa* w przestrzeni X, jeśli posiada tę własność pewien otwarty i gęsty podzbiór przestrzeni X.

Niech X będzie rodziną wszystkich M-elementowych zbiorów $\{Y_M\}_j$ złożonych z wektorów z przestrzeni $\mathbb{R}^N$, przy czym $M \leq N$. Liniowa niezależność wektorów należących do $\{Y_M\}_j$ jest własnością typową, ponieważ mając danych M wektorów N-wymiarowych, możemy z nich utworzyć macierz $N \times M$, której kolumny tworzą dane wektory. Liniowa niezależność danej rodziny wektorów implikuje, że utworzona macierz ma kwadratową nieosobliwą podmacierz $M \times M$. Liniowa niezależność wynika więc z powszechnie znanego faktu, że zbiór macierzy nieosobliwych wymiaru $M \times M$ jest otwarty i gęsty w zbiorze wszystkich macierzy rzeczywistych wymiaru $M \times M$.

Dla zastosowań praktycznych oznacza to, że możliwość przypadkowego utworzenia takiego ciągu jest zaniedbywalnie mała.

4.3.3. *Słabo zaburzone liniowe sieci neuronowe.* Zarówno dynamika potoków liniowych, jak i problem istnienia topologicznego sprzężenia pomiędzy generowanymi przez nie kaskadami (por. poprzedni podrozdział) są zagadnieniami szczegółowo zbadanymi. W zagadnieniach tego typu wyniki dotyczące problemów liniowych często w jakimś stopniu przenoszą się również na niewielkie zaburzenia tych problemów. Tak więc na przykład twierdzenie Hartmana (por. np. [30, str. 114, tw. 5.15]) mówi o globalnym topologicznym sprzężeniu kaskad generowanych przez słabo zaburzone operatory liniowe w przestrzeni Banacha, natomiast twierdzenie Grobmana–Hartmana (por. [30, str. 117, tw. 5.25]) mówi o globalnym sprzężeniu potoków bę-

dących słabymi zaburzeniami potoków liniowych. Wniosek z twierdzenia Grobmana–Hartmana ([30, str. 118, tw. 5.26]) mówi o lokalnym sprzężeniu w pobliżu hiperbolicznego punktu równowagi układu dynamicznego i jego lokalnej linearyzacji.

Z problemami tymi związane jest również twierdzenie udowodnione przez Fečkana [17] o sprzężeniu kaskady generowanej przez dyskretyzację potoku generowanego przez słabo zaburzone równanie liniowe z kaskadą generowaną przez metodę Eulera dla tego równania. Twierdzenie to można sformułować następująco (wersja oryginalna jest nieco silniejsza):

TWIERDZENIE 4.3.1. *Niech $(\phi, \mathbb{R}^n)$ będzie układem dynamicznym generowanym przez równanie*

$$\dot{x} = \mathbf{A}x + g(x), \tag{18}$$

gdzie A jest macierzą rzeczywistą $n \times n$ nie posiadającą urojonych wartości własnych, natomiast $g \in C^1(\mathbb{R}^n, \mathbb{R}^n)$ oraz dla każdego $x \in \mathbb{R}^n$, $\sup |g(x)| < \infty$ i $|Dg(x)| \leq b$, gdzie b jest wystarczająco małą stałą dodatnią. Wtedy, dla dostatecznie dużego q, istnieją $h_0 > 0$ i homeomorfizm H_h (który w sposób ciągły zależy od h):

$$H_h : \mathbb{R}^n \to \mathbb{R}^n, \qquad h \in (0, h_0),$$

takie, że na kuli $B_q := \{x : |x| \leq q\}$ zachodzi równość

$$\phi(\cdot, h) \circ H_h(\cdot) = H_h(\cdot) \circ \psi_h(\cdot),$$

gdzie ψ_h jest dane przez metodę Eulera:

$$\psi_h(x) = x + h \cdot \mathbf{A}x + h \cdot g(x).$$

Twierdzenie powyższe można wykorzystać do analizy dynamiki procesu numerycznego realizującego uczenie pewnej klasy sieci neuronowych, które dalej będziemy nazywać słabo zaburzonymi sieciami liniowymi.

DEFINICJA 4.3.2. Neuron nazywamy *słabo zaburzonym neuronem liniowym*, jeśli jego funkcja aktywacji ma postać

$$f : \mathbb{R} \ni s \mapsto f(s) = s + u(s), \tag{19}$$

gdzie funkcja u jest ograniczona, $u \in C^2(\mathbb{R})$ oraz istnieją dostatecznie małe stałe dodatnie c_1, c_2, c_3, c_4 takie, że dla każdego $s \in \mathbb{R}$ prawdziwe są nierówności $|u'(s)| < c_1$, $|u''(s)| < c_2$, $|s \cdot u'(s)| < c_3$, $|s \cdot u''(s)| < c_4$.

Udowodnimy, że proces uczenia neuronu liniowego słabo zaburzonego i, co za tym idzie, sieci jednowarstwowej składającej się z takich neuronów jest modelowany przez równanie różniczkowe (18).

TWIERDZENIE 4.3.3. *Proces uczenia słabo zaburzonego neuronu liniowego metodą gradientową z kwadratową funkcją błędu jest modelowany przez równanie różniczkowe spełniające założenia twierdzenia 4.3.1.*

Dowód. Przypuśćmy, że mamy neuron o M wejściach oraz N-elementowy ciąg uczący $((\vec{x}(1), z(1)), \ldots, (\vec{x}(N), z(N)))$. Wtedy całkowite wzbudzenie neuronu w n-tym kroku nauki jest dane wzorem

$$s(n) = \sum_{m=1}^{M} x_m(n) w_m(n).$$

Jeśli rozważany neuron jest neuronem liniowym słabo zaburzonym, to jego funkcja aktywacji ma postać daną wzorem (19), natomiast kwadratowa funkcja błędu jest dana wzorem

$$E(w_1, \ldots, w_M) = \frac{1}{2} \sum_{n=1}^{N} [s(n) + u(s(n)) - z(n)]^2.$$

Obliczymy k-tą składową gradientu:

$$\begin{aligned} \frac{\partial E}{\partial w_k} &= \sum_{n=1}^{N} \{[s(n) + u(s(n)) - z(n)] \cdot [x_k(n) + u'(s(n) \cdot x_k(n)]\} \\ &= \sum_{n=1}^{N} x_k(n) s(n) - \sum_{n=1}^{N} x_k(n) \cdot z(n) \\ &\quad + \sum_{n=1}^{N} x_k(n) \cdot [u(s(n)) + u'(s(n)) \cdot f(s(n)) - u'(s(n)) \cdot z(n)]. \end{aligned}$$

Gradientowe wektorowe równanie różniczkowe modelujące proces uczenia rozważanego neuronu,

$$\dot{\vec{w}} = -\operatorname{grad} E(\vec{w}),$$

można więc zapisać w postaci

$$\dot{\vec{w}} = -(\mathbf{A}\vec{w} + \mathbf{B} + \tilde{g}(\vec{w})),$$

gdzie macierze $\mathbf{A}$, $\mathbf{B}$ są dane wzorami odpowiednio (15) i (16), a k-ta składowa wektora $\tilde{g}(\vec{w})$ ma postać

$$\tilde{g}(\vec{w})_k = \sum_{n=1}^{N} x_k(n) \cdot [u(s(n)) + u'(s(n)) \cdot f(s(n)) - z(n) \cdot u'(s(n)s)].$$

Jeśli macierz $G(\overline{x}_1, \ldots, \overline{x}_M)$ jest nieosobliwa, to potoki generowane przez równania $\dot{\vec{w}} = \mathbf{A}\vec{w} + \mathbf{B}$ i $\dot{\vec{w}} = \mathbf{A}\vec{w}$ są topologicznie sprzężone. Ponieważ liniowa niezależność wektorów $\overline{x}_1, \ldots, \overline{x}_M$ jest własnością typową, więc nieosobliwość macierzy $G(\overline{x}_1, \ldots, \overline{x}_M)$ jest również typowa oraz, co za tym idzie, typowe jest też topologiczne sprzężenie rozważanych potoków. Sprzężenie jest przechodnie, więc wystarczy rozważyć równanie

$$\dot{\vec{w}} = \mathbf{A}\vec{w} + \tilde{g}(\vec{w}).$$

Macierz pochodnych funkcji $\widetilde{g}$ jest następująca:

$$\frac{\partial \widetilde{g}(\vec{w})_k}{\partial \vec{w}_s} = \sum_{n=1}^{N} x_k(n) \cdot x_s(n) \cdot [u'(s(n)) + s(n)u''(s(n)) + u''(s(n)) \cdot u(s(n)) + u'(s(n)) \cdot (1 + u'(s(n))) - z(n) \cdot u''(s(n))].$$

Jeśli funkcja u będzie spełniała odpowiednie warunki dotyczące ograniczoności jej modułu oraz modułu jej pierwszej i drugiej pochodnej, to tym samym, funkcja $\widetilde{g}$ będzie spełniala założenia zarówno twierdzenia 4.3.1, jak i twierdzenia Grobmana–Hartmana. Oznacza to, że:

- Potok generowany przez równanie

$$\dot{\vec{w}} = -(\vec{w} + \widetilde{g}(\vec{w}))$$

jest globalnie topologicznie sprzężony z potokiem generowanym przez jego część liniową

$$\dot{\vec{w}} = -\mathbf{A}\vec{w}.$$

Fakt ten implikuje m.in. asymptotyczną stabilność procesu uczenia.

- Kaskada generowana przez dyskretyzację równania

$$\dot{\vec{w}} = -(\mathbf{A}\vec{w} + \widetilde{g}(\vec{w}))$$

jest, na dużej kuli, topologicznie sprzężona z kaskadą generowaną przez metodę Eulera dla tego równania (według której działa algorytm komputerowy uczenia sieci neuronowej).

Zastosowanie opisanych wyników do implementacji sieci zbudowanej ze słabo nieliniowych neuronów wymaga oszacowania stałych występujących w twierdzeniu 4.3.1. Oszacowania te zostały dokonane w [11].

Zauważmy, że rozważanie problemu na dostatecznie dużej kuli, a nie w całym $\mathbb{R}^n$ jest wystarczające zarówno w kontekście modelowania komórek nerwowych, jak i implementacji komputerowych. Całkowite wzbudzenie neuronu biologicznego jest ograniczone, gdyż zarówno $|\vec{w}|$, jak $\vec{x}$ są ograniczone. W układach biologicznych jest to konsekwencją zasady funkcjonowania synaps komórki nerwowej. Pod wpływem bodźców nadchodzących do synapsy kolbki synaptyczne wydzielają substancje chemiczne zwane neuromediatorami, np. acetylocholinę, adrenalinę, noradrenalinę, dopaminę i inne. Substancje te oddziałują na błonę komórki, powodując zmianę jej potencjału elektrycznego, przy czym polaryzacja może mieć charakter zarówno pobudzający, jak hamujący. Końcowa polaryzacja błony zależy od sumarycznej ilości pobudzających i hamujących neuromediatorów wydzielonych przez kolbki synaptyczne ([56, str. 39–40]). Neuron może oczywiście uwolnić jedynie ograniczoną ilość neuromediatorów. Natomiast w komputerowym modelowaniu sieci neuronowych ograniczoność modułu wektora $\vec{w}$ wynika z

faktu, że istnieje ograniczenie na moduł liczby, która może być reprezentowana w komputerze. Z tych samych powodów moduł wektora wejściowego $\vec{x}$ jest ograniczony w implementacjach komputerowych. Natomiast jego ograniczoność przy modelowaniu układów biologicznych wynika między innymi z faktu, że komórka ma pewną tolerancję na bodźce zewnętrzne i w przypadku bodźców zbyt silnych zostałaby po prostu zniszczona.

4.3.4. *Nieliniowe sieci neuronowe.* Rozważmy jednowarstwową, nieliniową sieć neuronową zbudowaną z neuronów posiadających po dwa wejścia. Załóżmy, że funkcje aktywacji neuronów są ograniczonymi funkcjami klasy $\mathcal{C}^2(\mathbb{R},\mathbb{R})$ oraz że ich pierwsze i drugie pochodne również są ograniczone. Większość stosowanych w praktyce nieliniowych funkcji aktywacji spełnia powyższe założenia (np. funkcje unipolarne i bipolarne — por. np. [45, str. 38], większość funkcji radialnych — por. np. [45, str. 168]). W sieci jednowarstwowej neurony uczone są niezależnie, możemy więc ograniczyć się do analizy procesu uczenia pojedynczego neuronu.

Do analizy dynamiki użyjemy nastąpującego twierdzenia o sprzężeniu topologicznym.

TWIERDZENIE 4.3.1. *Niech $\mathcal{S}^2$ będzie dwuwymiarową sferą w $\mathbb{R}^3$, natomiast*

$$\phi : \mathcal{S}^2 \times \mathbb{R} \mapsto \mathcal{S}^2$$

układem dynamicznym generowanym przez równanie gradientowe

$$\dot{x} = -\operatorname{grad} E(x), \tag{20}$$

gdzie $E \in \mathcal{C}^2(\mathcal{S}^2,\mathbb{R})$, posiadającym skończoną liczbę punktów stałych; niech wszystkie te punkty będą hiperboliczne. Niech ponadto układ dynamiczny ϕ nie posiada połączeń typu siodło-siodło. Załóżmy również, że $\phi_h : \mathcal{S}^2 \to \mathcal{S}^2$ jest dyskretyzacją układu ϕ, tzn. $\phi_h(x) := \phi(x,h)$, natomiast $\psi_h : \mathcal{S}^2 \to \mathcal{S}^2$ niech będzie odwzorowaniem odpowiadającym metodzie Eulera dla równania (20). Wtedy dla dostatecznie małego $h > 0$ istnieje homeomorfizm $\alpha = \alpha_h : \mathcal{S}^2 \to \mathcal{S}^2$ globalnie sprzęgający dyskretne układy dynamiczne generowane przez ϕ_h i ψ_h, tzn. spełniona jest następująca równość:

$$\phi_h \circ \alpha = \alpha \circ \psi_h. \tag{21}$$

Jak wiadomo, Aksjomat A wraz z warunkiem silnej transwersalności (*strong transversality condition*) jest równoważny strukturalnej stabilności układu dynamicznego ([48, str. 171] oraz [39]). Z drugiej strony, dla układów gradientowych spełnienie Aksjomatu A implikuje, że układ posiada skończenie wiele punktów stałych i wszystkie one są hiperboliczne, natomiast z warunku silnej transwersalności wynika, że układ gradientowy nie ma połączeń typu siodło-siodło. Tak więc strukturalna stabilność rozważanego układu $(\mathcal{S}^2,\phi)$ implikuje założenia twierdzenia 4.3.1. Układy strukturalnie stabilne

natomiast tworzą zbiór otwarty i gęsty w zbiorze gradientowych układów dynamicznych (por. [48, str. 116]).

Twierdzenie 4.3.1 zostało udowodnione dla sfery $\mathcal{S}^2$ ([6], uogólniono również wynik na przypadek dwuwymiarowej rozmaitości zwartej — [9]), którą, poprzez rzut stereograficzny, możemy utożsamiać z uzwarconą płaszczyzną $\overline{\mathbb{R}}^2$. Może ono być użyte do analizy dynamiki procesu uczenia neuronu nieliniowego, jeśli potencjał E w równaniu gradientowym (13) można w sposób ciągły dookreślić (wraz z pierwszą i drugą pochodną) w biegunie północnym sfery $\mathcal{S}^2$. Dla warstwowej sieci neuronowej potencjałem jest funkcja błędu. W założeniach twierdzenia 4.3.1 żądamy, aby potencjał był funkcją klasy $\mathcal{C}^2(\mathcal{S}^2, \mathbb{R})$. Jednakże funkcja kryterialna dana wzorem (14), który dla pojedynczego neuronu z dwuskładnikowym wejściem przybiera postać

$$(22) \quad E(\vec{w}) = \frac{1}{2}\sum_{n=1}^{N}[f(s(n)) - z(n)]^2, \qquad s(n) := x_1(n)\cdot w_1 + x_2(n)\cdot w_2,$$

jest nieciągła w biegunie północnym sfery $\mathcal{S}^2$, gdyż do bieguna północnego możemy zmierzać na różne sposoby (zmierzanie do bieguna północnego na sferze $\mathcal{S}^2$ odpowiada na płaszczyźnie $\mathbb{R}^2$ przejściu granicznemu, przy którym $|\vec{w}| \to \infty$), między innymi w ten sposób, że iloczyn skalarny $\vec{x} \circ \vec{w}$ pozostaje stały. Można jednak funkcję kryterialną tak zmodyfikować, aby w interesującym nas obszarze nie wprowadziło to żadnych zmian w dynamice i żeby możliwe było jej ciągłe dookreślenie w biegunie północnym.

Jak wspomniano w poprzednim podrozdziale, zarówno w biologicznych, jak i sztucznych sieciach neuronowych wektory $\vec{w}$ i $\vec{x}$ są ograniczone. Stąd też przy tworzeniu modeli matematycznych wystarczy rozważać wektory ograniczone co do modułu.

Wybierzmy więc wystarczająco duże $a \in \mathbb{R}$ będące promieniem koła $K((0,0),a) \subset \mathbb{R}^2$, na którym będziemy rozpatrywać proces uczenia. Zmodyfikujemy funkcję kryterialną za pomocą odwzorowania zdefiniowanego następująco:

$$(23) \qquad g(\vec{w}) := \begin{cases} e^{(r-a)^p} & \text{dla } r \geq a, \\ 1 & \text{dla } r \in [0,a), \end{cases}$$

gdzie $\vec{w} = [w_1, w_2]$, $r := |\vec{w}|^2 = w_2^2 + w_2^2$, funkcja g jest klasy $\mathcal{C}^2(\mathbb{R}^2, \mathbb{R})$, a stała p jest dobrana w opisany niżej sposób. Zdefiniujmy funkcję

$$(24) \qquad E^*(\vec{w}) := g(\vec{w})\cdot E(\vec{w}).$$

Dobierając stałą a dostatecznie dużą, zapewnimy sobie, że w interesującym nas obszarze funkcja E nie zostanie zmieniona. Zauważmy, że rozwiązania równania

$$(25) \qquad \dot{\vec{w}} = -\operatorname{grad} E^*(\vec{w})$$

„wchodzą" do koła $K((0,0), r = 2a)$, tzn. $-\operatorname{grad} E^*(\vec{w}) \circ \vec{w}$ przyjmuje dla $|\vec{w}| = \sqrt{2a}$ wartości ujemne. Rzeczywiście,

$$-\operatorname{grad} E^*(\vec{w}) \circ \vec{w}$$
$$= \sum_{i=1}^{2} \left[E(\vec{w}) \cdot \frac{\partial g(\vec{w})}{\partial w_i} \cdot w_i + g(\vec{w}) \cdot \sum_{n=1}^{N} \left[x_i(n) \cdot \frac{df}{ds} \cdot (f(s(n)) - z(n)) \right] \cdot w_i \right].$$

Ponieważ

$$\frac{\partial g(\vec{w})}{\partial w_i} := \begin{cases} 2 \cdot w_i \cdot p \cdot (r-a)^{p-1} \cdot e^{(r-a)^p} & \text{dla } r > a, \\ 0 & \text{dla } r \in [0, a], \end{cases} \tag{26}$$

więc dla $r = 2a$ otrzymujemy

$$-\operatorname{grad} E^*(\vec{w}) \circ \vec{w}$$
$$= -e^{a^p} \cdot \sum_{i=1}^{2} \left[p \cdot a^{p-1} \cdot w_i^2 \cdot E(\vec{w}) + \sum_{n=1}^{N} \left[x_i(n) \cdot \frac{df}{ds} \cdot (f(s(n)) - z(n)) \right] \cdot w_i \right].$$

Człon będący sumą indeksowaną wskaźnikiem n może być ujemny, ale nie zależy od p. Natomiast pierwszy człon pod sumą indeksowaną wskaźnikiem i jest dodatni, ponieważ funkcja E zeruje się co najwyżej dla jednego wektora wag — tam, gdzie przy ustalonym ciągu uczącym E ma minimum globalne, więc a można tak dobrać, aby minimum globalne należało do wnętrza rozważanego koła K. W związku z tym, ponieważ a jest duże, można dobrać tak duże p, że wyrażenie stojące pod sumą indeksowaną wskaźnikiem i będzie dodatnie i, co za tym idzie, całe wyrażenie ujemne.

Przyjmijmy, że w pewnym otoczeniu bieguna północnego sfery $\mathcal{S}^2$, które, na płaszczyźnie $\mathbb{R}^2$, odpowiada wartościom $|\vec{w}|$ większym od, powiedzmy, $\sqrt{a}$, potencjał jest określony przez funkcję typu $V(\vec{w}) = -R^2$, która ma maksimum w biegunie północnym, przy czym R jest odległością od bieguna północnego. Wtedy potok generowany przez równanie

$$\dot{\vec{w}} = -\operatorname{grad} V(\vec{w})$$

rozważane we wspomnianym otoczeniu bieguna północnego ma w biegunie północnym odpychający hiperboliczny punkt stały. Można określić funkcję $\tilde{E} \in C(\mathcal{S}^2, \mathbb{R})$ tak, aby

$$\tilde{E}(\vec{w}) = \begin{cases} E^*(\vec{w}) & \text{dla } |\vec{w}| \leq \sqrt{2a}, \\ V(\vec{w}) & \text{dla } |\vec{w}| \geq \sqrt{3a}. \end{cases}$$

Ponieważ rozwiązania równania (25) „wchodzą" do koła $K((0,0), r = 2a)$, więc koło to jest zbiorem niezmienniczym układu dynamicznego; dlatego, mimo modyfikacji potencjału, w interesującym nas obszarze dynamika układu pozostaje niezmieniona. Tak więc, jeśli potok generowany przez równanie (25) ma w kole $K((0,0), r = 2a)$ skończoną liczbę wyłącznie hiperbolicznych

punktów stałych, to potok generowany przez równanie

$$\dot{\vec{w}} = -\text{grad}\,\widetilde{E}(\vec{w})$$

spełnia założenia twierdzenia 4.3.1.

Możliwość zastosowania twierdzenia 4.3.1 jest ograniczona do skończonej liczby punktów stałych, przy czym wszystkie one muszą być hiperboliczne. W zastosowaniach kluczową rolę odgrywa wspomniany już fakt, że układy strukturalnie stabilne tworzą w zbiorze układów gradientowych zbiór otwarty i gęsty (por. [48, str. 116]) oraz że implikują założenia twierdzenia 4.3.1. Oznacza to, że istnienie skończonej liczby wyłącznie hiperbolicznych punktów stałych jest własnością typową w zbiorze gradientowych układów dynamicznych.

Potoki gradientowe są układami dynamicznymi, których zachowanie jest bardzo regularne. W szczególności, nie mają one punktów okresowych ani nie mogą być chaotyczne. Własności te przenoszą się na dyskretyzację i, dzięki topologicznemu sprzężeniu, również na metodę Eulera, co implikuje asymptotyczną zbieżność procesu uczenia sieci neuronowej modelowanego przez wspomniane kaskady.

5. PODSUMOWANIE

Matematyczna analiza różnych aspektów sieci neuronowych weszła na trwałe do tematyki badawczej związanej z sieciami. Opracowano już wiele zagadnień, niemniej jednak wciąż pojawiają się nowe problemy. Używane są różne narzędzia matematyczne — jako przykłady można wymienić teorię aproksymacji, układy dynamiczne z teorią równań różniczkowych, rachunek prawdopodobieństwa i statystykę matematyczną. Dużą popularność zyskały metody bazujące na bezpośredniej adaptacji modeli wykorzystywanych w fizyce matematycznej ([29], [34]). Obserwuje się zjawisko stosowania coraz to nowych metod matematycznych.

Związek sztucznych sieci neuronowych z biologicznymi pierwowzorami został w niniejszej publikacji jedynie zasygnalizowany. Warto jednak pamiętać o tym ważnym aspekcie badań. W rozdziale 12 monografii [34] poruszono zagadnienie możliwości modelowania wyższych procesów psychicznych, na przykład świadomości, za pomocą sieci neuronowych. Oczywiście, na razie nie ma takich możliwości i otwarte pozostaje pytanie, czy kiedykolwiek się pojawią. Natomiast nie ulega wątpliwości, że warunkiem koniecznym zaistnienia takiej możliwości jest powstanie adekwatnych modeli matematycznych opisujących wyższe procesy psychiczne oraz rozwój istniejących modeli sztucznych sieci neuronowych.

Literatura

[1] F. Alouges, A. Debussche, *On the qualitative behaviour of the orbits of a parabolic partial differential equation and its discretization in the neighbourhood of a hiperbolic fixed point*, Numer. Funct. Anal. Optim. 12 (1991), 253–269.

[2] D. Amit, *Modeling Brain Functions*, Cambridge Univ. Press, Cambridge, 1989.

[3] K. Aihara, T. Takabe, M. Toyoda, *Chaotic neural networks*, Phys. Lett. A. 144 (1990), 333–340.

[4] A. R. Barron, *Universal approximation bounds for superposition of sigmoidal functions*, IEEE Trans. Information Theory 39 (1993), 930–944.

[5] W. J. Beyn, *On the numerical approximation of phase portraits near stationary points*, SIAM J. Numer. Anal. 24 (1987), 103–122.

[6] A. Bielecki, *Topological conjugacy of cascades generated by gradient flows on the two-dimensional sphere*, Ann. Polon. Math. 73 (2000), 37–57.

[7] A. Bielecki, *Topological conjugacy of gradient cascades and learning processes of a nonlinear neuron*, Opuscula Math. 20 (2001), 249–258.

[8] A. Bielecki, *Dynamical properties of learning process of weakly nonlinear and nonlinear neurons*, Nonlinear Anal. Real World Appl. 2 (2001), 249–258.

[9] A. Bielecki, *Topological conjugacy of discrete time-map and Euler discrete dynamical systems generated by a gradient flow on a two-dimensional compact manifold*, Nonlinear Anal. 51 (2002), 1293–1317.

[10] A. Bielecki, *Mathematical model of architecture and learning process of artificial neural networks*, Task Quart. 7 (2003), 93–114.

[11] A. Bielecki, D. Jabłoński, *Estimation of numerical dynamics constants of a weakly nonlinear neuron*, in: Lecture Notes in Computer Sci. 2328, Springer, 2002, 862–869.

[12] W. J. Beyn, J. Lorenz, *Center manifolds of dynamical systems under discretizations*, Numer. Funct. Anal. Optim. 9 (1987), 381–414.

[13] S. Broomhead, D. Lowe, *Multivariable functional interpolation and adaptive network*, Complex Systems 2 (1988), 321–323.

[14] G. Cybenko, *Approximation by superposition of a sigmoidal function*, Math. Control Signals Systems 2 (1989), 303–314.

[15] B. Demidowicz, *Matematyczna teoria stabilności*, WNT, Warszawa, 1972.

[16] M. Fečkan, *Asymptotic behaviour of stable manifolds*, Proc. Amer. Math. Soc. 111 (1991), 585–593.

[17] M. Fečkan, *The relation between a flow and its discretization*, Math. Slovaca 42 (1992), 123–127.

[18] B. Garay, *Discretization and some qualitative properties of ordinary differential equations about equilibria*, Acta Math. Univ. Comen. 62 (1993), 245–275.

[19] B. Garay, *Discretization and Morse–Smale dynamical systems on planar discs*, Acta Math. Univ. Comen. 63 (1994), 25–38.

[20] B. Garay, *On structural stability of ordinary differential equations with respect to discretization methods*, J. Numer. Math. 4 (1996), 449–479.

[21] B. Garay, *On C^j-closeness between the solution flow and its numerical approximation*, J. Diff. Eq. Appl. 2 (1996), 67–86.

[22] B. Garay, *The discretized flow on domains of attraction: a structural stability result*, IMA J. Numer. Anal. 18 (1998), 77–90.

[23] E. Gelenbe, *Stability of the random neural network model*, Neural Comput. 2 (1990), 239–247.

[24] R. Hecht-Nielsen, *Kolmogorov's mapping neural network existence theorem*, w: Proc. Internat. Conf. Neural Networks, Part III, IEEE Press, New York, 1987.

[25] R. Hecht-Nielsen, *Neurocomputing*, Addison-Wesley, Reading, 1990.
[26] K. Hornik, *Approximation capabilities of multilayer feedforward networks*, Neural Networks 4 (1991), 251–258.
[27] K. Hornik, *Some new results on neural network approximation*, Neural Networks 6 (1993), 1069–1072.
[28] K. Hornik, M. Stinchcombe, H. White, *Mulitilayer feedforward networks are universal approximators*, Neural Networks 2 (1989), 359–366.
[29] J. Hertz, A. Krogh, R. G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Welsey, 1991.
[30] M. C. Irwin, *Smooth dynamical systems*, Academic Press, London, 1987.
[31] A. Krzyżak, T. Linder, G. Lugosi, *Nonparametric estimation and classification using radial basis function nets and empirical risk minimization*, IEEE Trans. Neural Networks 7 (1996), 475–487.
[32] P. E. Kloeden, J. Lorenz, *Stable attracting sets in dynamical systems and their one-step discretizations*, SIAM J. Numer. Anal. 23 (1986), 986–996.
[33] A. N. Kolmogorov, *On the representation of continuous functions of several variables by superposition of continuous functions of one variable*, Dokl. Akad. Nauk SSSR 114 (1957), 953–956.
[34] R. A. Kosiński, *Sztuczne sieci neuronowe — dynamika nieliniowa i chaos*, WNT, Warszawa, 2002.
[35] J. Korbicz, A. Obuchowicz, D. Uciński, *Sztuczne sieci neuronowe — podstawy i zastosowania*, Akademicka Oficyna Wyd. PLJ, Warszawa, 1994.
[36] A. Krzyżak, E. Rafajłowicz, *Aproksymacja funkcji za pomocą jednokierunkowych sieci neuronowych*, w: M. Nełęcz (red.), Biocybernetyka i Inżynieria Biomedyczna, tom 6: Sieci neuronowe, W. Duch, J. Korbicz, L. Rutkowski, R. Tadeusiewicz (red. tomu), Akademicka Oficyna Wyd. EXIT, Warszawa, 2000, 1–28.
[37] V. Kurkova, *Kolmogorov's theorem and multilayer neural networks*, Neural Networks 5 (1992), 501–506.
[38] P. Lula, *Jednokierunkowe sieci neuronowe w modelowaniu zjawisk ekonomicznych*, Wydawnictwo Akademii Ekonomicznej w Krakowie, Zeszyty Naukowe — Seria Specjalna: Monografie nr 140, Kraków, 1999.
[39] J. Mańdziuk, *Sieci neuronowe typu Hopfielda. Teoria i przykłady zastosowań*, Akademicka Oficyna Wyd. EXIT, Warszawa, 2000.
[40] D. Moody, J. Darken, *Fast learning in networks of locally-tuned processing units*, Neural Comput. 1 (1989), 281–294.
[41] M. Minsky, S. Papert, *Perceptrons*, MIT Press, Cambridge, 1969.
[42] M. Mrozek, K. P. Rybakowski, *Discretized ordinary differential equations and the Conley index*, J. Dyn. Diff. Eq. 4 (1992), 57–63.
[43] B. Müller, J. Reinhardt, *Neural Networks*, Springer, New York, 1990.
[44] W. S. McCulloch, W. H. Pitts, *A logical calculus of the ideas imminent in nervous activity*, Bull. Math. Biophysics 5 (1943), 115–133.
[45] S. Osowski, *Sieci neuronowe w ujęciu algorytmicznym*, WNT, Warszawa, 1996.
[46] A. J. Owens, D. L. Filkin, *Efficient training of the back propagation network by solving a system of stiff ordinary differential equations*, w: Int. Joint Conf. on Neural Networks, vol. II (Washington, 1989), IEEE, New York.
[47] J. Park, I. W. Sandberg, *Approximation and radial-basis-function networks*, Neural Comput. 5 (1993), 305–316.
[48] J. Palis, W. de Melo, *Geometric Theory of Dynamical Systems*, Springer, New York, 1982.
[49] I. T. Podolak, *Functional graph model of a neural network*, IEEE Trans. Systems Men Cybernetics B 28 (1998), 876–881.

[50] I. T. Podolak, *Feed-forward neural network's sensitivity to input data representation*, Computer Physics Comm. 117 (1998), 62–93.

[51] D. Rutkowska, M. Piliński, L. Rutkowski, *Sieci neuronowe, algorytmy genetyczne i systemy rozmyte*, PWN, Warszawa–Łódź, 1997.

[52] L. Rutkowski (red.), *Sieci neuronowe i neurokomputery*, Wyd. Politech. Częstochowskiej, Częstochowa, 1996.

[53] A. Scott, *Schody do umysłu*, WNT, Warszawa, 1999 (wyd. oryginalne: *Stairway to the Mind*, Springer, New York, 1995).

[54] R. Srzednicki, *Topological invariants and detection of periodic orbits*, J. Diff. Eq. 111 (1994), 283–298.

[55] R. Tadeusiewicz, *Sieci neuronowe*, Akademicka Oficyna Wyd., 1993.

[56] R. Tadeusiewicz, *Problemy biocybernetyki*, PWN, Warszawa, 1994.

[57] R. Tadeusiewicz, *Wstęp do sieci neuronowych*, w: M. Nełęcz (red.), Biocybernetyka i Inżynieria Biomedyczna, tom 6: Sieci neuronowe, W. Duch, J. Korbicz, L. Rutkowski, R. Tadeusiewicz (red. tomu), Akademicka Oficyna Wyd. EXIT, Warszawa, 2000, 1–28.

[58] Y. Takefuji, K. C. Lee, *An artificial hysteresis binary neuron; a model suppressing the oscillatory behaviours of neural dynamics*, Biological Cybernetics 64 (1991), 353–356,

[59] S. G. Waxman, *Regional differentiation of the axon: a review with special reference to the concept of the multiplex neuron*, Brain Res. 47 (1972), 269–288.

[60] S. Wiggins, *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, Springer, New York, 1990.

[61] J. Żurada, M. Barski, W. Jędruch, *Sztuczne sieci neuronowe*, PWN, Warszawa, 1996.

Instytut Informatyki
Uniwersytet Jagielloński
E-mail: uibielec@cyf-kr.edu.pl