Rafał Latkowski
rl167288@students.mimuw.edu.pl

SIECI HOPFIELDA — TEORIA I ZASTOSOWANIA

Praca pod kierunkiem prof. Andrzeja Skowrona
A.Skowron@mimuw.edu.pl

Warszawa 1999

Streszczenie

Praca opisuje budowę sieci Hopfielda na przykładzie prostego zastosowania. W rozdziale pierwszym zaprezentowany jest krótki rys historyczny ilustrujący miejsce sieci Hopfielda w nauce o sztucznych sieciach neuronowych. Rozdział drugi prezentuje budowę sieci, oraz klasyczny opis probabilistyczny wraz ze skróconym wyprowadzeniem pojemności sieci oraz klasyfikacją stanów fałszywych. Trzeci rozdział opisuje próbę zastosowania sieci do przykładowego zadania. Zawiera dwie próby zastosowań opierających się na wiedzy z poprzedniego rozdziału oraz alternatywne wyprowadzenie warunku stabilności wraz z rozwiązaniem zadania.

Spis rzeczy

Rozdział 1

Wstęp

Początek zainteresowania sieciami neuronowymi sięga lat 40. i jest wynikiem przełomowej pracy McCullocha i Pittisa prezentującej matematyczny model neuronu, jednakże wtedy jeszcze nie było możliwe modelowanie sztucznych sieci neuronowych. W latach 60., za sprawą konstrukcji Perceptronu przez Rosenblatta i dowodu zbieżności jego reguły uczenia, wiązano niewspółmiernie duże nadzieje z możliwością postępu w tej dziedzinie wiedzy. Jednakże zapał ten został na kilkanaście lat ostudzony pracą Minsky'ego i Paperta wykazującą ograniczenia ówczesnej wiedzy, uniemożliwiające zastosowania sieci do wielu prostych problemów. Odrodzenie zainteresowania sieciami neuronowymi nastąpiło w roku 1982, w którym ukazała się przełomowa praca Hopfielda prezentująca zupełnie nowy model sieci neuronowych. Sieci te, opierające się na pomyśle symetrycznych połączeń rekurencyjnych, umożliwiły zastosowanie pojęcia funkcji energetycznej, dzięki czemu stało się możliwe przetłumaczenie bogactwa osiągnięć fizyki statystycznej na język sieci neuronowych. Dlatego w porównaniu do klasycznych jednokierunkowych sieci neuronowych (perceptronów) sieci Hopfielda mają dużo ciekawsze i obszerniejsze podstawy teoretyczne.

Podstawowym zastosowaniem sieci Hopfielda jest realizacja pamięci asocjacyjnej — pamięci adresowanej treścią, które posłuży jako ilustracja przy prezentowaniu tego modelu. Bardzo dobre rezultaty odniesiono również na polu rozwiązywania przybliżonego kombinatorycznych problemów NP-zupełnych. Rozszerzony model takich sieci zwany maszynami Boltzmana ma potencjalnie bardzo szeroki zakres zastosowań, jednakże nie doczekał się jeszcze licznych implementacji ze względu na znaczny stopień złożoności obliczeniowej procesów uczenia.

Rozdział 2

Sieci Hopfielda

W tym rozdziale zaprezentowany zostanie podstawowy model sieci neuronowych zaproponowany przez Hopfielda, na przykładzie standardowego zadania - pamięci asocjacyjnej. Pamięć — pamięć adresowana treścią ma odtwarzać wzorce uprzednio wprowadzone na podstawie podobieństwa. Załóżmy, że w takiej pamięci umieścimy bitmapy zdjęć kilku osób. Sieć taka na podstawie zaszumionego obrazu jednego ze zdjęć powinna odtworzyć jego oryginał, tj. najbardziej przypominające go zdjęcie[1] uprzednio zapamiętane w sieci. Cechą charakterystyczną pamięci asocjacyjnej jest to, że jej zawartość nie jest ponumerowana tak jak ma to miejsce w typowej pamięci klasycznych komputerów, ale jest „adresowana zawartością" (CAM - content adressable memory). Taka implementacja pamięci nie przechowuje zawartości w wydzielonych częściach (komórkach pamięci), ale równomiernie rozprasza po całym jej obszarze[2] (pamięć hologeniczna).

2.1 Budowa sieci

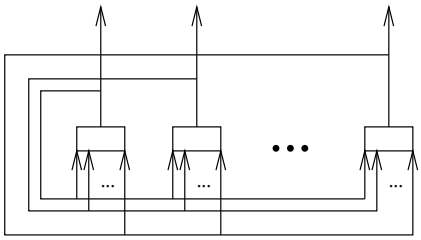
Podstawowy model bazuje na jednostkach nieliniowych oznaczanych S_i o wartościach ze zbioru $\{-1, +1\}$, oraz skokowej funkcji aktywacji $S_i := sign(h_i)$, (przyjmuje się, że $sign(0) = +1$). Sieć składa się z jednej warstwy neuronów połączonych każdy z każdym (zobacz rys. 2.1) za pomocą współczynników wagowych w_{ij} oznaczających połączenie jednostki j-tej do i-tej.

Oto wzór opisujący zachowanie się pojedynczego neuronu

$$S_i = sign(\sum_j w_{ij} S_j - b_i) \tag{2.1}$$

[1] W sensie odległości Hamminga.

[2] Objawia się to równie ż z stopniowym spadkiem, a nie załamaniem, jakości takiej sieci podczas usuwania kolejnych jednostek.



Rysunek 2.1: Tak należy wyobrażać sobie model połączeń w sieci Hopfielda.

Ponieważ jednak próg aktywacji nie będzie potrzebny do przetwarzania losowych wzorców możemy uprościć ten wzór do postaci

$$S_i = sign(\sum_j w_{ij} S_j) \tag{2.2}$$

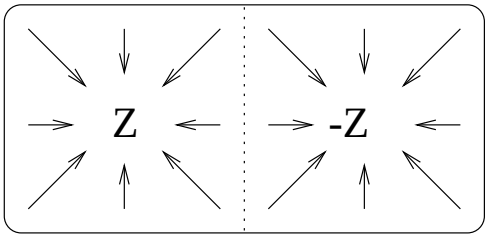
Kolejnym aspektem jaki trzeba ustalić jest sposób aktualizacji stanów. W sieciach jednokierunkowych (bez sprzężenia zwrotnego) występował naturalny częściowy porządek jednostek który można było dowolnie rozszerzyć do porządku liniowego aktualizacji jednostek, a łączna aktualizacja całej sieci nazywana była turą symulacji. W tym modelu nie można sensownie mówić o kolejności jednostek. Można wyróżnić dwie strategie aktualizacji stanów:

- synchronicznie - zmieniając stan wszystkich jednostek jednocześnie. Taka strategia wymaga centralnego zegara lub taktowania przy implementacjach sprzętowych.
- asynchronicznie - po jednej jednostce w danym kroku. W każdym kroku losowo wybieramy jednostkę i-tą i aktualizujemy jej stan. Jest to naturalny sposób działania niezależnie zaimplementowanych fizycznie (np. elektronicznie) jednostek sieci.

Bardziej naturalne i tutaj zaprezentowane będzie asynchroniczne aktualizowanie stanów. Ze względu na sprzężenie zwrotne należy się spodziewać, że sieć będzie się zachowywała w sposób dynamiczny, określony przez stan początkowy oraz współczynniki wagowe połączeń. Oczekiwać będziemy, że sieć po pewnym czasie ustabilizuje się i jednostki nie będą się już przełączać. Ogólnie, tutaj i dalej rozważane będą tylko takie sieci które dążą do stabilnego stanu końcowego (co jak później zostanie pokazane zapewnia symetryczność połączeń oraz $w_{ii} = 0$).

2.1.1 Przypadek jednego wzorca

Przypuśćmy, że chcemy w takiej sieci zapamiętać jeden wzorzec **Z**. Będzie to N elementowy ciąg (wektor) o wartościach ze zbioru $\{-1, +1\}$. Chcemy, aby



Rysunek 2.2: Przedstawienie przestrzeni stanów sieci jako płaszczyzny daje możliwość intuicyjnego naszkicowania obszarów przyciągania w przypadku jednego wzorca.

sieć po zaprezentowaniu jej takiego wzorca (czyli ustawieniu początkowego stanu jednostek zgodnego z **Z**) zachowywała swój stan, tj.:

$$\forall_i \;\; sign(\sum_{j=1}^{N} w_{ij} Z_j) = Z_i \tag{2.3}$$

ponieważ wówczas postępowanie zgodnie z regułą aktualizacji jednostek nie daje żadnych zmian. Zobaczmy, że jest to spełnione gdy przyjmiemy

$$w_{ij} \sim Z_i Z_j \tag{2.4}$$

Dla ułatwienia dalszych obliczeń przyjmijmy współczynnik proporcjonalności $\frac{1}{N}$

$$w_{ij} = \frac{1}{N} Z_i Z_j \tag{2.5}$$

Spójrzmy na całkowite pobudzenie wejściowe jednostki i-tej

$$h_i = \sum_{j=1}^{N} w_{ij} S_j \tag{2.6}$$

Jeśli teraz za S_i podstawimy Z_i i rozpiszemy wzór 2.6 to otrzymamy

$$h_i = \sum_{j=1}^{N} \frac{1}{N} Z_i Z_j Z_j = Z_i \tag{2.7}$$

Widać zatem, że stan **Z** jest stabilny, a także nawet jeśli pewna (niewielka) liczba bitów wzorca początkowego jest błędna to zostaną one zdominowane przez większość bitów prawidłowych tak, że znak h_i nie ulegnie zmianie. Oznacza to, że sieć będzie korygować błędy. Możemy to sobie wyobrazić, że wzorzec **Z** stanowi punkt przyciągania (atraktor) w przestrzeni stanów sieci. Oprócz tego jest jeszcze jeden atraktor, a mianowicie taki w którym wszystkie stany są odwrócone (-**Z**). spójrzmy na 2.7. Jeśli początkowy stan sieci ma ponad połowę bitów przeciwnych wtedy zamiast do oryginalnego wzorca, będzie przyciągany do wzorca odwróconego (zobacz rys. 2.2).

2.1.2 Przypadek wielu wzorców

Przypuśćmy teraz, że mamy p wzorców losowych i oznaczmy je $Z^1 ... Z^p$ Najprostszym sposobem, jaki można zastosować do ustalenia odpowiednich współczynników, jest superpozycja składników dla pojedynczych wzorców. Okazuje się, że właśnie taki wzór jest skuteczny.

$$w_{ij} = \frac{1}{N} \sum_{m=1}^{p} Z_i^m Z_j^m \tag{2.8}$$

Wzór ten nazywany jest „regułą Hebba" lub „uogólnioną regułą Hebba" ponieważ zmiany wag są proporcjonalne do korelacji aktywności presynaptycznej i postsynaptycznej neuronu. Jednakże odbiega on trochę od biologicznego realizmu oryginalnego pomysłu Hebba, gdyż daje dodatnią zmianę wagi gdy żaden z neuronów nie jest w stanie pobudzenia (+1), a także podczas dokładania nowych wzorców połączenie może zmienić swój znak z dodatniego na ujemny, co jest niemożliwe w przypadku rzeczywistym. Można zmodyfikować ten wzór tak, aby uniknąć tych niezgodności, tym bardziej, że komplikuje to konstrukcję niektórych implementacji fizycznych (np. elektronicznych przy pomocy wzmacniaczy operacyjnych).

Aby przekonać się o słuszności doboru współczynników należy zbadać stabilność sieci, czyli

$$\forall_i \ \ sign(h_i^n) = Z_i^n \tag{2.9}$$

co zgodnie z wzorem daje

$$Z_i^n = sign(\sum_{j=1}^{N} w_{ij} Z_i^n) = sign(\sum_{j=1}^{N} \frac{1}{N} \sum_{m=1}^{p} Z_i^m Z_j^m Z_i^n) \tag{2.10}$$

wyłączmy z sumy składnik $n = m$

$$Z_i^n = sign(Z_i^n + \frac{1}{N} \sum_{j=1}^{N} \sum_{m \neq n}^{p} Z_i^m Z_j^m Z_i^n) \tag{2.11}$$

Jak widać, jeśli tylko składnik

$$\frac{1}{N} \sum_{j=1}^{N} \sum_{m \neq n}^{p} Z_i^m Z_j^m Z_i^n \tag{2.12}$$

jest zerem, wtedy natychmiast dostajemy, że wzorzec jest stabilny. Ponadto jeśli moduł z tego składnika jest mniejszy od jedynki, to i tak Z_i^n zdominuje znak wyrażenia. Nazwijmy ten składnik **przesłuchem** albo **szumem**. Badanie przesłuchu pozwoli nam stwierdzić kiedy sieć funkcjonuje dobrze. Gdy przesłuch jest dostatecznie mały (lub tego samego znaku co bit wzorca) niewielkie odchylenia od oryginalnego wzorca zostaną skorygowane przez sieć tak samo, jak to miało miejsce w przypadku jednego wzorca.

$P(C_i^n > 1)$	p_{max}/N
0,001	0,105
0,0036	0,138
0,01	0,185
0,05	0,37
0,1	0,61

Tablica 2.1: Pojemność pamięci

2.2 Pojemność sieci

Ustalmy znak wyrażenia 2.12 mnożąc je przez $-Z_i^n$

$$C_i^n = -Z_i^n \frac{1}{N} \sum_{j=1}^{N} \sum_{m \neq n}^{p} Z_i^m Z_j^m Z_i^n \tag{2.13}$$

Jeśli C_i^n jest większe od $+1$ wtedy składnik szumu przeważa we wzorze 2.11 i i-ty bit wzorca n jest niestabilny. Szacując $P(C_i^n > 1)$, tj. prawdopodobieństwo zdarzenia, że wybrany bit jest niestabilny dostaniemy wyrażenie określające maksymalną pojemność sieci.

W celu dalszych obliczeń załóżmy, że rozpatrywane przez nas wzorce Z^k, to niezależne zmienne losowe, takie, że $P(Z_i^k = -1) = P(Z_i^k = +1) = \frac{1}{2}$. Wartości C_i^n zależą wprost od wzorców Z_j^m. $P(C_i^n > 1)$ zależy od liczby neuronów N i liczby wzorców p. Przy założeniach $N > 1$ i $p > 1$ mamy, że C_i^n to $\frac{1}{N}$ sumy około Np niezależnych zmiennych losowych, z których każda jest równa +1 lub -1. Chociaż C_i^n ma rozkład dwumianowy o wartości oczekiwanej 0 i wariancji $\sigma^2 = \frac{p}{N}$, to dla dużych Np możemy zastosować przybliżenie rozkładem Gaussa o tej samej średniej i wariancji ([3] s.160-163).

$$P(C_i^n > 1) = \frac{1}{\sqrt{2\pi}\sigma} \int_1^{\infty} e^{-x^2/2\sigma^2} dx \tag{2.14}$$

gdzie $\sigma^2 = \frac{p}{N}$. Jeśli zdefiniujemy funkcję błędu

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{2.15}$$

wtedy możemy to prościej napisać

$$P(C_i^n > 1) = \frac{1}{2}(1 - erf(\sqrt{N/2p})) \tag{2.16}$$

Numerycznie możemy stwierdzić jaki stosunek p/N jest wymagany, aby uzyskać zadane prawdopodobieństwo błędu (tj. tego, że przesłuch przekroczy 1). W

tablicy 2.1 pokazano maksymalną pojemność pamięci (wyrażoną w p/N) wymaganą w celu otrzymania zadanego prawdopodobieństwa błędu na jednym bicie.

Oczywiście można badać też pojemność (czyli p/N) uzależniając ją od poprawnego odtwarzania większości wzorców (zamiast większości bitów, jak powyżej). Każdy wzorzec posiada N bitów, dlatego, aby otrzymać N bitów poprawnych z prawdopodobieństwem α

$$(1 - P(C_i^n > 1))^N > \alpha \tag{2.17}$$

co można przybliżyć rozwinięciem dwumianowym do

$$P(C_i^n > 1) < \frac{1-\alpha}{N} \tag{2.18}$$

Widać zatem, że $p/N \to 0$ gdy $N \to \infty$. To pozwala na zastosowanie rozwinięcia asymptotycznego funkcji błędu

$$1 - erf(x) \to \frac{e^{-x^2}}{\sqrt{\pi}x} \qquad \text{gdy } x \to \infty \tag{2.19}$$

W rezultacie uzyskujemy

$$\frac{1}{2}\frac{e^{-N/2p}}{\sqrt{\pi}\sqrt{N/2p}} \tag{2.20}$$

Po zlogarytmowaniu nierówności 2.18 mamy

$$-\ln 2 - \frac{N}{2p} - \frac{1}{2}\ln\pi - \frac{1}{2}\ln\left(\frac{N}{2p}\right) < \ln(1-\alpha) - \ln N \tag{2.21}$$

Przy $N \to \infty$ możemy uwzględniać tylko główny składnik

$$\frac{N}{2p} > \ln N \tag{2.22}$$

co daje $p_{max} = N/2\ln N$.

Idąc dalej można by żądać, aby wszystkie bity były odtwarzane z zadanym prawdopodobieństwem. Musimy wtedy otrzymać Np bitów poprawnych. Widać, że wtedy 2.22 zmienia się w

$$\frac{N}{2p} > \ln(Np) \tag{2.23}$$

co daje $p_{max} = N/4\ln N$, ponieważ $\ln(Np) < \ln N^2$.

Podsumowując, pojemność p_{max} jest proporcjonalna do N, jeśli dopuszczamy mały procent bitów w każdym wzorcu, a proporcjonalna do $N/\ln N$, gdy nalegamy, aby większość lub prawie wszystkie wzorce były odtwarzane idealnie.

2.3 Funkcja energetyczna

Jednym z najważniejszych osiągnięć pracy Hopfielda było wprowadzenie do teorii sieci neuronowych pojęcia funkcji energetycznej. Dla naszych sieci będzie to funkcja

$$H = -\frac{1}{2}\sum_{i,j} w_{ij} S_i S_j \tag{2.24}$$

Najważniejszą własnością funkcji energetycznej jest to, że zawsze maleje (lub pozostaje stała) gdy układ ewoluuje zgodnie z regułą 2.11. W ten sposób zapamiętane wzorce leżą w minimach lokalnych powierzchni funkcji energetycznej. Układ ewoluując porusza się po tej powierzchni na podobieństwo ruchu cząstki pod wpływem siły ciężkości ciągnącej w dół i tarcia nie pozwalającego jej podskakiwać. Obszary przyciągania odpowiadają dolinom, czyli obszarom przechwytywania wokół minimum. Układ startujący w określonej dolinie dąży do najniższego punktu tej doliny.

W wielu dziedzinach istnieje funkcja stanu, która zawsze maleje podczas ewolucji dynamiki lub która musi być minimalizowana, aby znaleźć stan stabilny lub optymalny. W wielu dziedzinach mamy sytuację odwrotną. Funkcja rośnie lub też musi być maksymalizowana. Najbardziej ogólna nazwa, pochodząca z teorii układów dynamicznych, to funkcja Lapunowa. Inne nazwy to hamiltonian w mechanice statystycznej, funkcja kosztu lub funkcja celu w teorii optymalizacji lub funkcja przystosowania w programowaniu ewolucyjnym.

Dla sieci neuronowych funkcja energetyczna istnieje, gdy wagi połączeń są symetryczne $w_{ij} = w_{ji}$. W rzeczywistych sieciach jest to założenie nieuzasadnione, ale badanie przypadku symetrycznego jest uzasadnione, ponieważ dzięki istnieniu funkcji energetycznej można przetłumaczyć na język sieci neuronowych osiągnięcia innych dziedzin nauki, a w szczególności fizyki.

Ponieważ $S_i \cdot S_i = +1$ oraz $w_{ij} = w_{ji}$ funkcję energetyczną 2.24 możemy zapisać tak:

$$H = C - \sum_{i<j} w_i j \cdot S_i \cdot S_j \tag{2.25}$$

Teraz łatwo jest wykazać, że reguła dynamiczna 2.2 może tylko zmniejszyć energię. Niech S_i' będzie nową wartością S_i

$$S_i' = sign(\sum_j w_{ij} S_j) \tag{2.26}$$

Oczywiście, jeśli $S_i' = S_i$, to energia nie zmienia się. W przeciwnym przy-

padku $S_i' = -S_i$ i możemy wyliczyć zmianę energii.

$$\begin{aligned} H' - H &= -\sum_{j\neq i} w_{ij} S_i' S_j + \sum_{j\neq i} w_{ij} S_i S_j = \\ &= 2S_i \sum_{j\neq i} w_{ij} S_j = \\ &= 2S_i \sum_{j} w_{ij} S_j - 2w_{ii} \end{aligned} \tag{2.27}$$

Pierwszy wyraz jest ujemny na podstawie reguły aktualizacji 2.26, a drugi jest ujemny, ponieważ reguła Hebba 2.8 daje $w_{ii} = p/N$ dla każdego i. A zatem energia maleje przy każdej zmianie S_i, zgodnie z postulatem.

Skł adniki sprzężenia zwrotnego w_{ii} można pominąć zarówno na podstawie reguły Hebba (możemy po prostu przyjąć, że $w_{ii} = 0$), jak też według funkcji energetycznej. Łatwo można sprawdzić, że nie wnoszą one istotnej różnicy do stabilności wzorców, ale silnie wpływają na dynamikę i liczbę stanów fałszywych, dlatego lepiej je pomijać. Możemy wydzielić składnik sprzężenia zwrotnego z wyrażenia określającego regułę aktualizacji 2.2

$$S_i := sign(w_{ii} S_i + \sum_{j\neq i} w_{ij} S_j) \tag{2.28}$$

Gdyby w pewnym stanie w_{ii} było większe niż $\sum_{j\neq i} w_{ij} S_j$, wówczas oba stany $S_i = +1$ i $S_i = -1$ były by stabilne. Może to wytworzyć dodatkowe stabilne stany fałszywe w sąsiedztwie pożądanego atraktora, zmniejszając obszar przyciągania. Jeśli $w_{ii} = 0$ to problem ten nie występuje.

2.4 Od funkcji energetycznej do reguł y Hebba

Idea funkcji energetycznej, jako czegoś, co ma być minimalizowane w stanach stabilnych daje nam alternatywną drogę do wyprowadzenia reguły Hebba 2.8. Funkcja energetyczna w jednoznaczny sposób wyznacza wartości współczynników sieci.

Zacznijmy od przypadku jednego wzorca. Chcemy, żeby energia była minimalna, gdy występuje największa część wspólna pomiędzy konfiguracją sieci a zapamiętanym wzorcem Z. Wybieramy więc

$$H = -\frac{1}{2N}\left(\sum_i S_i Z_i\right)^2 \tag{2.29}$$

Analogicznie w przypadku wielu wzorców możemy spróbować umieścić każdy z nich w minimum lokalnym funkcji sumując po wszystkich wzorach

$$H = -\frac{1}{2N}\sum_{m=1}^{p}\left(\sum_i S_i Z_i^m\right)^2 \tag{2.30}$$

Po wymnożeniu otrzymujemy

$$H = -\frac{1}{2N}\sum_{m=1}^{p}\left(\sum_i S_i Z_i^m\right)\left(\sum_j S_j Z_j^m\right) = -\frac{1}{2}\sum_{i,j}\left(\sum_{m=1}^{p} Z_i^m Z_j^m\right) S_i S_j \quad (2.31)$$

co ma dokładnie taką samą postać jak nasza oryginalna funkcja energetyczna, jeśli w_{ij} jest obliczona według reguły Hebba 2.4. To podejście do poszukiwania wag w_{ij} jest ogólnie użyteczne. Jeśli możemy napisać funkcję energetyczną, której minimum jest rozwiązaniem danego problemu, to możemy wyznaczyć właściwe wartości połączeń w_{ij}

2.5 Klasyfikacja stanów fał szywych

Reguła Hebba 2.4 daje układ dynamiczny, który ma atraktory — minima lokalne funkcji energetycznej — w pożądanych punktach Z_i^m. Są one niekiedy nazywane **stanami odtwarzanymi**. Okazuje się jednak, że nie są to jedyne atraktory.

Oprócz wzorców Z_i^m, także stany odwrócone $-Z_i^m$ tworzą minima o takiej samej energii jak oryginalne wzorce. Reguła aktywacji (2.2) i funkcja energetyczna (2.25) są idealnie symetryczne $S_i \leftrightarrow -S_i$ dla każdego i. Nie jest to jednak kłopotliwe dla odtwarzanych stanów i moglibyśmy się zgodzić na odwrócenie wszystkich pozostałych bitów, gdy pewien szczególny „bit znaku" jest ujemny.

Inne stabilne stany to **stany mieszane**, które nie są żadnym pożądanym wzorcem, ale za to odpowiadają liniowej kombinacji nieparzystej liczby wzorców. Najprostsze są symetryczne kombinacje trzech zapamiętanych wzorców

$$Z_i^{mix} = sgn(+ - Z_i^{m_1} + -Z_i^{m_2} + -Z_i^{m_3}) \quad (2.32)$$

Możliwa jest każda z ośmiu kombinacji, ale przeanalizujemy przypadek, gdy wszystkie znaki są dodatnie. Pozostałe przypadki są podobne.

$$h_i^{mix} = \frac{1}{N}\sum_{(j,m)} Z_i^m Z_j^m Z_j^{mix} = \frac{1}{2}Z_i^{m_1} + \frac{1}{2}Z_i^{m_2} + \frac{1}{2}Z_i^{m_3} + \text{przesłuch} \quad (2.33)$$

W ten sposób warunek stabilności jest rzeczywiście spełniony dla stanu mieszanego 2.32. Podobnie można tworzyć kombinacje dowolnej nieparzystej liczby wzorców. Sieć nie wybiera kombinacji parzystej liczby wzorców, ponieważ w pewnych miejscach sumują się do zera, podczas gdy jednostki sieci muszą przyjmować stany $+ - 1$.

Oprócz tego dla dużych p istnieją minima lokalne nieskorelowane ze skończoną liczbą oryginalnych wzorców Z_i^m. Są one niekiedy nazywane **stanami szkł a spinowego** z powodu ścisłej analogii do modeli szkła spinowego w mechanice statystycznej.

Okazuje się, że pamięć nie działa idealnie. Występują w niej nie tylko pożądane stany odpowiadające zapamiętanym wzorcom, ale także wszystkie wymienione wyżej minima lokalne. Druga i trzecia klasa są ogólnie nazywane stanami fałszywymi. Ponieważ cechuje je mały obszar przyciągania, wpadamy do jednego z nich tylko wtedy, gdy startujemy blisko niego. Są rozmaite sposoby (np. symulowane wyżarzanie), które pozwalają zredukować lub usunąć minima fałszywe.

Rozdział 3

Badanie zastosowań

3.1 Sformuł owanie problemu

Wszystkie rozważania teoretyczne przeprowadzane były dla wzorców o losowej strukturze. Interesującym byłoby zaobserwowanie jak takie sieci radzą sobie w „warunkach bojowych". W idealnym modelu każdy bit wzorca był niezależny od pozostałych. W rzeczywistości zazwyczaj wzorce są w pewien sposób skorelowane, tak, że wymykają się opisowi jako niezależne zmienne losowe. Dlatego właśnie bardzo ciekawe jest przebadanie sięci w rzeczywistych zastosowaniach i wynikające z tego wnioski.

Zadanie „Telegazeta"

Należy zapamiętać i poprawnie odtwarzać N wzorców składających się z N znaków.

Wł asności zadania

Zauważmy najpierw że „rzeczywista" telegazeta posiada ok. 1000 ekranów po $25 \cdot 40 = 1000$ znaków na ekranie, co odpowiada $N = 1000$. Na znaki prawdziwej telegazety składają się litery, cyfry i znaki semigraficzne, w kilkudziesięciu zestawach kolorystycznych[1]. Niemniej jednak nas głównie interesować będzie zapamiętywanie liter, ale pojęcie znaku wymaga dalszego uściślenia i jest ono oddzielnie specyfikowane przy każdej z zastosowanych metod.

Zadanie pomimo swej prostoty sprawiło nieoczekiwane trudności przy realizacji go za pomocą sięci Hopfielda, nie dające się wywnioskować z probabilistycznego szacowania zaprezentowanego wcześniej.

[1] tj. kolor znaku razem z kolorem tł a, tak ˙ze „kolor" przezroczysty

3.2 Naturalna reprezentacja

Nasuwającym się podejściem do zakodowania znaków w postaci bitów jest przyjęcie ich naturalnej reprezentacji w postaci kodów ASCII .

Jak to wcześniej zilustrowano sieć Hopfielda nie preferuje bardziej wzorca zapamiętanego od jego stanu odwróconego. W związku z tym przyjęto, że najważniejszy bit znaku będzie znacznikiem odwrócenia i jego zanegowanie będzie odnosiło skutek podczas wyświetlania, kiedy to wszystkie bity znaku należy odwrócić. Nie jest to może nazbyt realistyczne założenie, należałoby raczej wprowadzić jeden bit znaku dla całej sieci, ale za to bardzo proste w implementacji i tak samo skuteczne przy badaniu stabilności wzorców.

3.2.1 Opis eksperymentu

Na eksperyment składają się 3 programy:

1. Program 1 — badający stabilność w sieci z 2 wzorcami.
2. Program 2 — badający zbieżność w sieci z 2 wzorcami.
3. Program 3 — badający stabilność w sieci z 3 wzorcami.

W każdym z tych programów zastosowano wzorce składające się z małych liter, o długości 10 znaków, co daje liczbę 80 jednostek w sieci.

Program 1

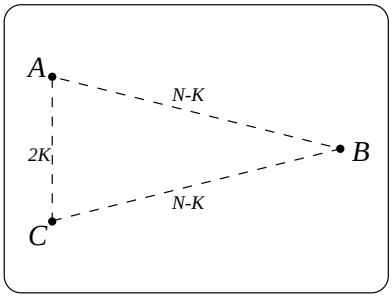
Program demonstruje działanie sieci Hopfielda z zapamiętanymi 2 wzorcami (reprezentowanymi naturalnie). Symulacja sieci rozpoczyna się od stanu wzorca pierwszego.

Okazuje się, że sieć spoczywa stabilnie w tym stanie. Ze względu na symetrię zadania (dla dwóch wzorców taka symetria występuje) można twierdzić, że obydwa wzorce są stabilne.

Program 2

Program demonstruje działanie sieci Hopfielda z zapamiętanymi 2 wzorcami (reprezentowanymi naturalnie). Symulacja sieci rozpoczyna się od stanu losowego.

Okazuje się, że sieć zbiega do jednego ze stanów zapamiętanych wzorców. Jednakże czasami „utyka” w minimach lokalnych funkcji energetycznej sieci. Ogólnie można powiedzieć, że sieć wykazuje się zbieżnością do zapamiętanych stanów, jeżeli tylko wystartuje dostatecznie blisko (w sensie odległości Hamminga) rozwiązania.



Rysunek 3.1: Tak można zilustrować przykład trzech wzorców w przestrzeni N bitowej. Wzorce rozpatrywane jako atraktory funkcji energetycznej powinny tworzyć wokół siebie obszary przyciągania.

Program 3

Program demonstruje działanie sieci Hopfielda z zapamiętanymi 3 wzorcami (reprezentowanymi naturalnie). Symulacja sieci rozpoczyna się od stanu wzorca pierwszego.

Sieć nie pozostaje w stanie pierwszego wzorca. Co więcej, nie zbiega do żadnego innego wzorca. W takiej sieci żaden z wzorców nie jest stabilny, a stabilne są jedynie pewne mieszaniny tych trzech wzorców.

3.2.2 Wnioski

Sieć nie zachowuje się tak, jak byśmy tego oczekiwali. Już przy $p = 3$ i $N = 80$ (tj. $p = 0,038 \cdot N$) nie można w żaden sposób skorzystać z sieci jako pamięci asocjacyjnej.

Przykł ad.

Rozważmy wzorce A, B i C w sieci o N jednostkach (tj. trzy wzorce N bitowe). Niech dla ustalonego $K < \frac{N}{2}$

$$\begin{array}{rcl} |A-B| & = & N-K \\ |A-C| & = & 2K \\ |B-C| & = & N-K \end{array} \tag{3.1}$$

gdzie $|x-y|$ oznacza odległość Hamminga, czyli liczbę bitów różniących obydwa wzorce (ciągi bitowe).

Przy odpowiednim ponumerowaniu jednostek sieci, własność 3.1 możemy za-

pisać w postaci jawnej:

$$\begin{array}{rcl} A & - & \text{dowolny wzorzec} \\ B_i & = & \begin{cases} A_i & : \ i \leq K \\ -A_i & : \ i > K \end{cases} \\ C_i & = & \begin{cases} -A_i & : \ i \leq K \\ A_i & : \ K < i \leq N-K \\ -A_i & : \ i > N-K \end{cases} \end{array} \tag{3.2}$$

Przyjrzyjmy się teraz stabilności wzorców (por. 2.10). Rozważmy stabilność wzorca A na pierwszych K bitach.

$$A_i = sign(\sum_{j=1}^{N} w_{ij} A_j) \qquad \text{gdzie } i \in \{1, \ldots, K\} \tag{3.3}$$

Ponieważ 3.2 jawnie zadaje nam wszystkie bity wzorców możemy więc obliczyć w_{ij}.

$$w_{ij} = \frac{1}{N}(A_i \cdot A_j + B_i \cdot B_j + C_i \cdot C_j) \tag{3.4}$$

Razem dostajemy

$$\begin{array}{rcl} A_i & = & sign(\frac{1}{N}\sum_{j=1}^{N}(A_i \cdot A_j + B_i \cdot B_j + C_i \cdot C_j) \cdot A_j) \\ & = & sign(\sum_{j=1}^{N} A_i \cdot A_j \cdot A_j + \sum_{j=1}^{N} B_i \cdot B_j \cdot A_j + \sum_{j=1}^{N} C_i \cdot C_j \cdot A_j) \end{array} \tag{3.5}$$

Na podstawie 3.2 możemy zrobić dla $i \leq K$ tabelkę wartości B_iB_j i C_iC_j:

	$j \leq K$	$K < j \leq N-K$	$j > N-K$
B_iB_j	A_iA_j	$-A_iA_j$	$-A_iA_j$
C_iC_j	A_iA_j	$-A_iA_j$	A_iA_j

Zatem 3.5 możemy zapisać w postaci

$$\begin{array}{rl} A_i = sign(& \sum_{j=1}^{N} A_iA_jA_j \\ + & \sum_{j=1}^{K} A_iA_jA_j - \sum_{j=K+1}^{N} A_iA_jA_j \\ + & \sum_{j=1}^{K} A_iA_jA_j - \sum_{j=K+1}^{N-K} A_iA_jA_j + \sum_{j=N-K+1}^{N} A_iA_jA_j) \end{array} \tag{3.6}$$

Po uproszczeniu

$$\begin{array}{c} A_i = sign(A_i(N + K - (N - K) + K + (N - 2K) + K)) \\ \Updownarrow \\ 6K - N > 0 \end{array} \tag{3.7}$$

Widać teraz, że jeśli tylko $K < \frac{N}{6}$ wtedy wzorzec A nie jest stabilny. Zauważmy, że zachodzi to dla dowolnie dużego N, gdzie szacowania probabilistyczne były dokładne.

3.2.3 Dyskusja modyfikacji

Źródła niepowodzeń można doszukiwać się w dużej wariancji odległości Hamminga pomiędzy dwoma wybranymi reprezentacjami znaków. Oczywiście można spróbować przekodować dane za pomocą np. kodu Graya, który jest takim pożądkiem liniowym ciągów bitowych, że dwa kolejne różnią się na jednym bicie. Niestety nadal odległość pomiędzy reprezentacją dwóch bardziej odległych liter może być dowolną liczbą ze zbioru $\{1, \ldots, 8\}$.

Ogólnie mówiąc, żadne wstępne kodowanie przestawieniowe nie zapewni nam małej wariancji odległości Hamminga pomiędzy losowo wybranymi reprezentacjami znaków.

3.3 Rzadka reprezentacja

Binarna reprezentacja znaków, która posiada oczekiwane własności, to np. reprezentacja rzadka. Jest to dosyć marnotrawna reprezentacja kodująca znak za pomocą jednego bitu ustawionego a pozostałych skasowanych. Jej inna nazwa to „1 z A" dla A elementowego alfabetu, np. „1 z 26", gdyż tylko jeden bit jest ustawiony w danym ciągu reprezentującym znak.

W celu jak największej przejrzystości kodu źródłowego zaniedbuje się tutaj problem odwróconych wzorców. Wpływa to ilościowo na częstość odtwarzania wzorca startując z losowego stanu początkowego, ale ze stałym współczynnikiem $\frac{1}{2}$ i nie przeszkadza w poprawnej klasyfikacji stanów stabilnych oraz przyciągających.

Uwaga.

Zauważmy, że dwie reprezentacje rzadkie „1 z 26" dowolnych znaków mogą być odległe o 0 (gdy takie same znaki) lub o 2 (gdy znaki są różne) w sensie odległości Hamminga. Zatem przy $N = 260$ i dziesięciu znakach we wzorcu odległość pomiędzy dwoma wzorcami jest liczbą parzystą z przedziału $[0, 20]$, czyli nie większą niż $0.077 \cdot N$.

3.3.1 Opis eksperymentu

Na eksperyment składają się 3 programy:

1. Program 4 — badający stabilność w sieci z 2 wzorcami.
2. Program 5 — badający zbieżność w sieci z 2 wzorcami.
3. Program 6 — badający stabilność w sieci z 3 wzorcami.

W każdym z tych programów zastosowano wzorce składające się z małych liter, o długości 10 znaków. Kodowane są tylko znaki ze zbioru {„a” ,. . . , „z”}, co oznacza 26 jednostek na znak i łączną liczbę 260 jednostek sieci.

Program 4

Program demonstruje działanie sieci Hopfielda z zapamiętanymi 2 wzorcami (reprezentowanymi rzadko — „1 z 26”). Symulacja sieci rozpoczyna się od stanu wzorca pierwszego.

Okazuje się, że sieć spoczywa stabilnie w tym stanie. Ze względu na symetrię zadania (dla dwóch wzorców taka symetria występuje) można twierdzić, że obydwa wzorce są stabilne.

Program 5

Program demonstruje działanie sieci Hopfielda z zapamiętanymi 2 wzorcami (reprezentowanymi rzadko — „1 z 26”). Symulacja sieci rozpoczyna się od stanu losowego.

Okazuje się, że sieć zbiega do jednego ze stanów zapamiętanych wzorców. Jednakże czasami „utyka” w minimach lokalnych funkcji energetycznej sieci. Ogólnie można powiedzieć, że sieć wykazuje się zbieżnością do zapamiętanych stanów, jeżeli tylko wystartuje dostatecznie blisko (w sensie odległości Hamminga) rozwiązania.

Należy zwrócić uwagę, że gdy w reprezentacji ustawiony jest więcej niż jeden bit, na ekranie pojawia się znak odpowiadający ostatniemu z ustawionych bitów. Dlatego częste występowanie „zzzz. . . ” w wyniku należy interpretować jako stan odwrócony lub mieszany. Nie należy sądzić, że w sieci stabilnym stanem jest właśnie „zzzz. . . ”.

Program 6

Program demonstruje działanie sieci Hopfielda z zapamiętanymi 3 wzorcami (reprezentowanymi rzadko — „1 z 26”). Symulacja sieci rozpoczyna się od stanu wzorca pierwszego.

Sieć nie pozostaje w stanie pierwszego wzorca. Co więcej, nie zbiega do żadnego innego wzorca. W takiej sieci żaden z wzorców nie jest stabilny, a stabilne są jedynie pewne mieszaniny tych trzech wzorców.

Niestety nie pomaga tutaj zrównoważenie odległości wzorców od siebie.

3.3.2 Wnioski

Pomimo zapewnienia takiej samej odległości Hamminga od wszystkich wzorców (w przykładowych danych wszystkie wzorce mają na ustalonej pozycji różne znaki) nie udało się ominąć problemu braku stabilności sieci z trzema zapamiętanymi wzorcami.

Należy zwrócić uwagę, że szacowania probabilistyczne zaprezentowane w drugim rozdziale zakładały równoprawdopodobny rozkład dwóch wartości bitów w reprezentacji wzorca.

3.3.3 Dyskusja modyfikacji

Oczywiście pod uwagę zostało wzięte założenie o równoprawdopodobnym rozkładzie +1 i -1 w binarnej reprezentacji wzorca, czyli napisu. Pomysł ten został przetestowany w **programie 7**.

Program 7

Program wzorowany jest na **programie 6**. Jedyną modyfikacją jest to, że każdy przetwarzany bit w reprezentacji binarnej jest zanegowany wtedy, gdy numer kolejny odpowiadającego neuronu jest parzysty. Innymi słowy każdy co drugi bit został zanegowany, co odpowiada $P(+1) = \frac{1}{2} + \epsilon$, gdzie $\epsilon \in [-\frac{1}{26}, +\frac{1}{26}]$.

Niestety także tutaj obserwacja zachowania się programu nie wskazuje na jakąkolwiek poprawę stabilności wzorca.

3.4 Model matematyczny

Po prezentacji pesymistycznie nastawiających wyników eksperymentalnych nadszedł wreszcie czas na systematyczne zanalizowanie zaistniałej sytuacji. Jak się wydaje szacowania za pomocą metod rachunku prawdopodobieństwa nie przynoszą zadowalającej klasyfikacji dobrego uwarunkowania sieci. Należy zauważyć, że sieci Hopfielda pojęciowo dają sie zcharakteryzować za pomocą zbiorów dyskretnych i jako takie dają się wyliczać niejako „na palcach”.

Przyjmijmy model podobny do zastosowanego poprzednio. Rozważać będziemy stabilność dowolnego wzorca Z^0 w zależności od pozostałych wzorców $Z^1, \ldots, Z^\mu$ oddalonych odpowiednio o $l_1, \ldots, l_\mu$ od Z^0.

Lemat 3.4.1

Niech Z^0 i Z^k będą dowolnymi wzorcami z przestrzeni N bitowej.
Jeśli

$$l_k = |Z^k - Z^0| \tag{3.8}$$

wtedy

$$\sum_{j=1}^{N} Z_i^k \cdot Z_j^k \cdot Z_j^0 = \begin{cases} (N - 2l_k)Z_i^0 & gdy \quad Z_i^0 = Z_i^k \\ (2l_k - N)Z_i^0 & gdy \quad Z_i^0 = -Z_i^k \end{cases} \tag{3.9}$$

Dowód

A) $Z_i^0 = Z_i^k$
Wtedy możemy napisać

$$\sum_{j=1}^{N} Z_i^k \cdot Z_j^k \cdot Z_j^0 = \sum_{j=1}^{N} Z_i^0 \cdot Z_j^k \cdot Z_j^0 \tag{3.10}$$

Następnie rozbijamy zbiór indeksów na dwa rozłączne podzbiory I_+ oraz I_- takie, że $I_+ \cup I_- = \{1, \ldots, N\}$ oraz $\forall_{j \in I_+} Z_j^k = Z_j^0$ i $\forall_{j \in I_-} Z_j^k = -Z_j^0$.

$$\begin{aligned} \sum_{j=1}^{N} Z_i^0 \cdot Z_j^k \cdot Z_j^0 &= \sum_{j \in I_+} Z_i^0 \cdot Z_j^0 \cdot Z_j^0 - \sum_{j \in I_-} Z_i^0 \cdot Z_j^0 \cdot Z_j^0 = \\ = Z_i^0 \cdot (card(I_+) - card(I_-)) &= Z_i^0 \cdot ((N - l_k) - (l_k)) = \\ &= (N - 2l_k) \cdot Z_i^0 \end{aligned} \tag{3.11}$$

B) $Z_i^0 = -Z_i^k$
Analogicznie do poprzedniego przypadku — wystarczy postawić minus przed każdym składnikiem równości, gdyż teraz Z_i^k ma przeciwny znak.

Twierdzenie 3.4.1

Mamy sieć o N jednostkach i $\mu+1$ zapamiętanych wzorcach $\{Z^0, \ldots, Z^\mu\}$ takich, że $l_k = |Z^k - Z^0|$.
Wzorzec Z^0 jest stabilny wtedy i tylko wtedy, gdy

$$\forall_{1 \le i \le N} \; N + \sum_{k=1}^{\mu} \varepsilon_i^k (N - 2l_k) \ge 0 \tag{3.12}$$

gdzie $\varepsilon_i^k \in \{-1, +1\}$ i jest zdefiniowane

$$\varepsilon_i^k = Z_i^0 \cdot Z_i^k \tag{3.13}$$

Dowód

Weźmy wzór określający stabilność pojedynczego bitu sieci (por. 2.10).

$$Z_i^0 = sign(\sum_{j=1}^{N} w_{ij} Z_i^0) = sign(\sum_{j=1}^{N} \frac{1}{N} \sum_{k=1}^{\mu} Z_i^k Z_j^k Z_i^0) \tag{3.14}$$

Z **lematu 3.4.1** wiemy, że dla ustalonego k

$$\sum_{j=1}^{N} Z_i^k \cdot Z_j^k \cdot Z_j^0 = \varepsilon_i^k (N - 2l_k) Z_i^0 \tag{3.15}$$

Zatem 3.14 możemy napisać tak:

$$\begin{gathered} Z_i^0 = sign(\frac{1}{N} \sum_{k=0}^{\mu} \sum_{j=1}^{N} Z_i^k \cdot Z_j^k \cdot Z_j^0) \\ \Updownarrow \\ Z_i^0 = sign(\frac{1}{N}(N \cdot Z_i^0 + \sum_{k=1}^{\mu} \varepsilon_i^k (N - 2l_k) Z_i^0)) \\ \Updownarrow \\ Z_i^0 = sign(N + \sum_{k=1}^{\mu} \varepsilon_i^k (N - 2l_k)) \cdot Z_i^0 \\ \Updownarrow \\ N + \sum_{k=1}^{\mu} \varepsilon_i^k (N - 2l_k) \geq 0 \end{gathered} \tag{3.16}$$

Ponieważ zachodzi to dla każdego i, twierdzenie jest prawdziwe.

Dalsze rozważania ograniczymy do reprezentacji rzadkiej, gdyż jest bardzo prosta do wymodelowania. Jednakże należy zauważyć, że wyniki jakościowo będą odpowiednie także dla innych reprezentacji.

Fakt 3.4.1

Dla pewnej klasy sieci z dużym prawdopodobieństwem istnieje takie i_+ oraz i_-, że

$$\forall_k \quad \varepsilon_{i_+}^k = +1 \quad \& \quad \varepsilon_{i_-}^k = -1 \tag{3.17}$$

Dowód

Zauważmy, że każde Z_i^k to pojedyncza próba Bernoulliego o $P(Z_i^k = +1) = \frac{1}{26}$. Znak ε_i^k informuje o tym, czy Z_i^k jest równe Z_i^0. Możemy zatem obliczyć prawdopodobieństwo tego, że wszystkie $\mu + 1$ bitów Z_i^k są tego samego znaku, co jest równoważne temu, że wszystkie ε_i^k są dodatnie.

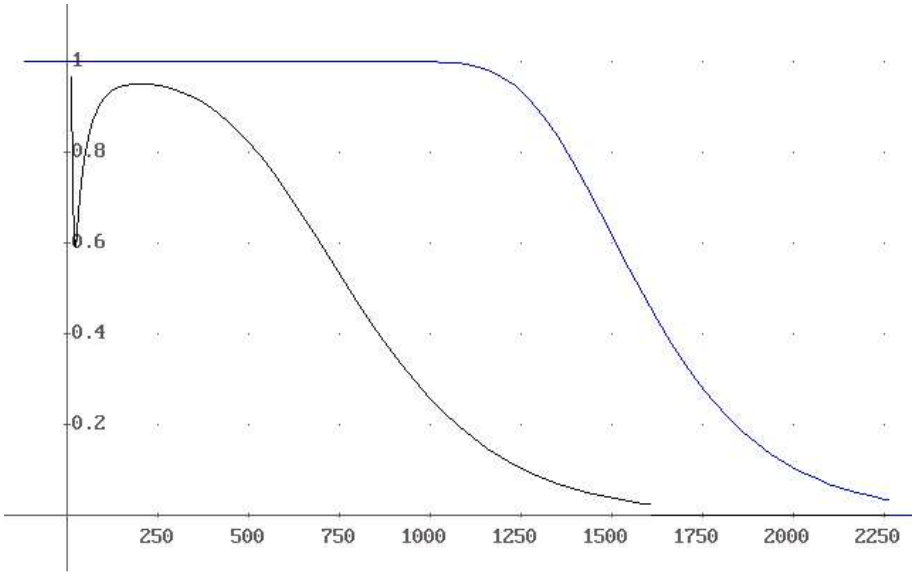
$$\begin{aligned} P(\forall_k\ \varepsilon_i^k = +1) &= P(Z_i^0 = +1) \cdot P(Z_i^1 = +1) \cdot \ldots \cdot P(Z_i^\mu = +1) \\ &+ P(Z_i^0 = -1) \cdot P(Z_i^1 = -1) \cdot \ldots \cdot P(Z_i^\mu = -1) \\ &\Updownarrow \\ P(\forall_k\ \varepsilon_i^k = +1) &= \frac{1}{26} \cdot \left(\frac{1}{26}\right)^\mu + \frac{25}{26} \cdot \left(\frac{25}{26}\right)^\mu \\ &\Updownarrow \\ P(\forall_k\ \varepsilon_i^k = +1) &= \frac{25^{\mu+1} + 1}{26^{\mu+1}} \end{aligned} \tag{3.18}$$

Analogicznie możemy obliczyć prawdopodobieństwo tego, że wszystkie ε_i^k są ujemne:

$$\begin{aligned} P(\forall_k\ \varepsilon_i^k = +1) &= P(Z_i^0 = +1) \cdot P(Z_i^1 = -1) \cdot \ldots \cdot P(Z_i^\mu = -1) \\ &+ P(Z_i^0 = -1) \cdot P(Z_i^1 = +1) \cdot \ldots \cdot P(Z_i^\mu = +1) \\ &\Updownarrow \\ P(\forall_k\ \varepsilon_i^k = +1) &= \frac{1}{26} \cdot \left(\frac{25}{26}\right)^\mu + \frac{25}{26} \cdot \left(\frac{1}{26}\right)^\mu \\ &\Updownarrow \\ P(\forall_k\ \varepsilon_i^k = +1) &= \frac{25^\mu + 25}{26^{\mu+1}} \end{aligned} \tag{3.19}$$

Ponieważ chcemy stwierdzić istnienie takiego bitu wśród N bitów wzorca, że jest on odpowiednio zgodny bądź niezgodny z pozostałymi, mamy N niezależnych prób, podczas których możemy wybrać taki bit. Oczywiście istnienie implikuje że jest co najmniej jeden taki bit, a nie dokładnie jeden.

$$\begin{gathered} P(\exists_i \forall_k\ \varepsilon_i^k = +1) = \\ P(|\{i \in \{1, \ldots, N\} : \forall_k\ \varepsilon_i^k = +1\}| \geq 1) = \\ 1 - P(|\{i \in \{1, \ldots, N\} : \forall_k\ \varepsilon_i^k = +1\}| = 0) = \\ 1 - (1 - P(\forall_k\ \varepsilon_i^k = +1))^N = \\ 1 - (1 - \frac{25^\mu + 25}{26^{\mu+1}})^N \end{gathered} \tag{3.20}$$



Rysunek 3.2: Wykres prezentuje prawdopodobieństwa tego, że $\exists_i \forall_k \ \varepsilon_i^k = +1$ (u góry) oraz $\exists_i \forall_k \ \varepsilon_i^k = -1$ (poniżej) względem liczby bitów wzorca. Przyjęto, że liczba wzorców jest ósmą częścią liczby bitów, czyli zbliżoną do maksymalnej użytecznej pojemności sieci.

oraz analogicznie:

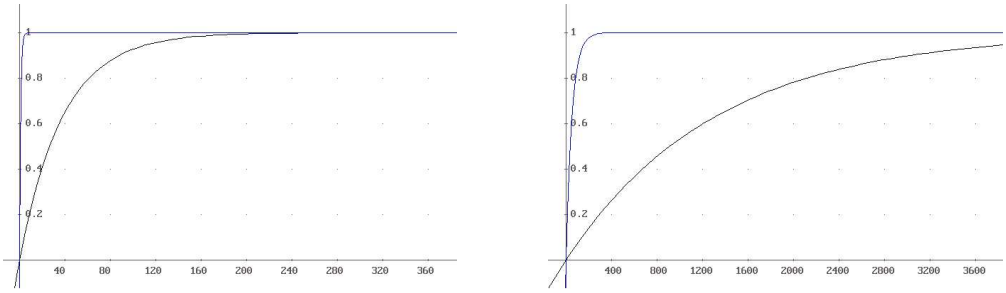
$$\begin{gathered} P(\exists_i \forall_k \ \varepsilon_i^k = -1) = \\ P(|\{i \in \{1, \ldots, N\} : \forall_k \ \varepsilon_i^k = -1\}| \geq 1) = \\ 1 - P(|\{i \in \{1, \ldots, N\} : \forall_k \ \varepsilon_i^k = -1\}| = 0) = \\ 1 - (1 - P(\forall_k \ \varepsilon_i^k = -1))^N = \\ 1 - (1 - \frac{25^{\mu+1} + 1}{26^{\mu+1}})^N \end{gathered} \tag{3.21}$$

Podsumowując prawdopodobieństwa tego, że istnieją i_+ oraz i_-, takie że:

$$\forall_k \quad \varepsilon_{i_+}^k = +1 \quad \& \quad \varepsilon_{i_-}^k = -1 \tag{3.22}$$

są zadane funkcjami liczby bitów i wzorców w sieci:

$$\begin{aligned} P(\exists_i \forall_k \ \varepsilon_i^k = +1) &= 1 - (1 - \frac{25^{\mu} + 25}{26^{\mu+1}})^N \\ P(\exists_i \forall_k \ \varepsilon_i^k = -1) &= 1 - (1 - \frac{25^{\mu+1} + 1}{26^{\mu+1}})^N \end{aligned} \tag{3.23}$$



Rysunek 3.3: Wykresy analogiczne do powyższego, jednakże prezentują prawdopodobieństwa przy ustalonej liczbie wzorców. Po lewej stronie dla 10 wzorców, po prawej dla 100 wzorców.

Zachowanie się tych funkcji obrazuje wykres 3.2. Przyjęto tam, że $\mu = \frac{N}{8}$, co w dobrym przybliżeniu odpowiada załadowaniu sieci wzorcami prawie do jej maksymalnej użytecznej pojemności (wynoszącej $0.138 \cdot N$). Jak widać, dla niedużych sieci (np. pięćset jednostek) prawdopodobieństwo istnienia takich i_+ i i_- jest dosyć znaczne. Co prawda asymptotycznie zbliża się do zera, gdy liczba jednostek wzrasta, ale możemy na to spojrzeć inaczej. Jeśli interesuje nas zapamiętanie w sieci konkretnej liczby wzorców, to wtedy, przy ustalonym μ prawdopodobieństwo jest opisane przez „odwróconą" funkcję wykładniczą monotonicznie rosnącą do jedynki. Prezentują to wykresy na rysunku 3.3.

Na podstawie powyższych obserwacji można powiedzieć, że w każdej małej (względem liczby jednostek) sieci istnieją i_+ oraz i_-, a także w takiej, w której zapamiętano niewielką liczbę wzorców w porównaniu do jej wielkości.

Lemat 3.4.2

Jeśli istnieją takie i_+ oraz i_-, że

$$\forall_k \quad \varepsilon^k_{i_+} = +1 \quad \& \quad \varepsilon^k_{i_-} = -1 \tag{3.24}$$

oraz

$$\forall_{1 \le i \le N} \; N + \sum_{k=1}^{\mu} \varepsilon^k_i (N - 2l_k) \ge 0 \tag{3.25}$$

wtedy

$$\frac{\mu - 1}{2} \cdot N \le \sum_{k=1}^{\mu} l_k \le \frac{\mu + 1}{2} \cdot N \tag{3.26}$$

Dowód

Weźmy i_+.

$$\begin{gathered} N + \sum_{k=1}^{\mu} \varepsilon_{i_+}^k (N - 2l_k) \geq 0 \\ \Updownarrow \\ N + \sum_{k=1}^{\mu} (N - 2l_k) \geq 0 \\ \Updownarrow \\ (\mu + 1) \cdot N \geq 2 \cdot \sum_{k=1}^{\mu} l_k \\ \Updownarrow \\ \tfrac{\mu+1}{2} \cdot N \geq \sum_{k=1}^{\mu} l_k \end{gathered} \tag{3.27}$$

Weźmy teraz i_-

$$\begin{gathered} N + \sum_{k=1}^{\mu} \varepsilon_{i_-}^k (N - 2l_k) \geq 0 \\ \Updownarrow \\ N + \sum_{k=1}^{\mu} (2l_k - N) \geq 0 \\ \Updownarrow \\ 2 \cdot \sum_{k=1}^{\mu} l_k \geq (\mu - 1) \cdot N \\ \Updownarrow \\ \sum_{k=1}^{\mu} l_k \geq \frac{\mu - 1}{2} \cdot N \end{gathered} \tag{3.28}$$

Obie nierówności razem dają nam tezę twierdzenia.

Fakt 3.4.2

Zauważmy, że nasza sieć, opisana w **programach 6** i **7** powinna spełniać warunek 3.26. W naszych programach $\sum l_k$ zawiera się w $\{0, \ldots, 40\}$, gdyż obydwa wzorce mogą się różnić maksymalnie na dwudziestu bitach, natomiast $N = 260$. Widać zatem, że $\sum l_k$ nijak się nie mieści w przedziale $[130, 390]$. Zatem wzorzec nie spełnia warunku stabilności 3.26.

Wnioski

Nasuwającym się pomysłem na poprawę zaistniałej sytuacji jest dodanie progów. Spróbujmy przerachować wszystko z uwzględnieniem progu aktywacji (stałego

dla całej sieci).

Reguła aktualizacji 2.2 przybiera teraz postać

$$S_i = sign(\sum_j w_{ij} S_j + \alpha) \tag{3.29}$$

Zatem pobudzenie w naszym przypadku wynosi

$$h_i = \frac{1}{N} \sum_{k=0}^{\mu} \sum_{j=1}^{N} Z_i^k \cdot Z_j^k \cdot Z_j^0 + \alpha \tag{3.30}$$

Natomiast nierówność 3.26 będąca tezą **lematu 3.4.2**

$$\frac{(\mu - 1) - \alpha \cdot Z_i^0}{2} \cdot N \leq \sum_{k=1}^{\mu} l_k \leq \frac{(\mu + 1) + \alpha \cdot Z_i^0}{2} \cdot N \tag{3.31}$$

Teraz wystarczy zauważyć, że prawa część nierówności odpowiada przypadkowi, gdy bity wszystkich pozostałych wzorców są różne od bitu Z_i^0, natomiast lewa gdy bity te są równe. W ogromnej większości przypadków bity pozostałych wzorców są różne od Z_i^0, gdy ten przyjmuje wartość $+1$, czyli, że wszystkie pozostałe znaki (litery) są różne od tego znaku wzorca. Odmiennie jest, gdy wszystkie bity są równe, to wskazuje na „puste miejsce" reprezentacji rzadkiej, czyli -1. Możemy zatem napisać, oczywiście mając świadomość braku pełnej implikacji, że

$$\frac{(\mu - 1) - \alpha}{2} \cdot N \leq \sum_{k=1}^{\mu} l_k \leq \frac{(\mu + 1) - \alpha}{2} \cdot N \tag{3.32}$$

W przypadku interesującego nas programu parametry wynosiły odpowiednio $N = 260$, $\mu = 2 \sum l_k = 40$

$$1 - \alpha \leq \frac{4}{13} \leq 3 - \alpha \tag{3.33}$$

Czyli $\alpha \in [\frac{9}{13}, 2\frac{9}{13}]$.

Program 8

Jest to zmodyfikowany **program 6**. Wprowadzono do niego progi aktywacji jednostek, dzięki czemu wzorce wreszcie są stabilne. Badania eksperymentalne wykazały, że dobrze jest przyjąć $\alpha \in [0.75, 0.95]$. Przy większych wartościach progu w sieci występowało lawinowe przełączanie się jednostek do stanu wzorca drugiego lub trzeciego. Natomiast wyjście ze stałą poza wyliczony we wniosku przedział powodowało, że stabilne były inne stany, zupełnie nie przypominające wzorców.

3.5 Podsumowanie

Jak zostało to pokazane sieć Hopfielda może być z powodzeniem stosowana jako pamięć homoasocjacyjna. Jednakże zastosowanie ogólnej probabilistycznej wiedzy o działaniu i pojemności takich sieci może nie wystarczyć. Podczas realizacji zadania może się okazać, że sieć nie działa tak, jak byśmy tego oczekiwali. Dlatego właśnie obliczenia „na palcach”, tj. metodami matematyki dyskretnej dają nam wgląd w zachowanie sieci dla naszego konkretnego przypadku. Skutkiem ubocznym takich obliczeń w tym przypadku było wprowadzenie progów (które nie występują w opisie sieci Hopfielda w dostępnej literaturze), co umożliwiło prawidłowe odtwarzanie wzorców.

Literatura

[1] J. Hertz, A. Krogh, R.G. Palmer „Wstęp do teorii obliczeń neuronowych” WNT 1993

[2] R. Tadeusiewicz „Sieci neuronowe” Akademicka Oficyna Wydawnicza PM 1993

[3] W. Feller „Wstęp do Rachunku Prawdopodobieństwa, Tom I” PWN 1977

[4] J.J. Hopfield „Neural Networks and Physical Systems with Emergent Colective Computational Abilities” 1982 Proceedings of the National Academy of Sciences

Spis rysunków

Spis tablic