

Samoorganizujące się sieci mieszankowe w reprezentacji fotografii cyfrowych w skali szarości

Patryk Filipiak
Instytut Informatyki, Uniwersytet Wrocławski
patryk.filipiak@ii.uni.wroc.pl

19 sierpnia 2011

Streszczenie

Sieci Kohonena (SOM) są najczęściej wykorzystywanym narzędziem w celu tzw. uczenia bez nadzoru. Dlatego też doczekały się wielu modyfikacji i adaptacji. Niniejsza praca poświęcona jest samoorganizującym się sieciom mieszankowym (SOMN), będącym istotnym rozwinięciem pierwotnej idei Kohonena. Zdolność SOMN do efektywnego uczenia się dowolnego rozkładu statystycznego ukazana została na przykładzie fotografii cyfrowych w skali szarości. Dowolny obraz cyfrowy w skali szarości może być przybliżony za pomocą skończonej mieszanki gaussowskiej, której parametry dobierane są automatycznie w procesie uczenia SOMN. W niniejszej publikacji przedstawiona została grupa przykładów takiego wykorzystania SOMN przy użyciu zaimplementowanej w tym celu aplikacji.

1 Wstęp

Reprezentacja w skali szarości obrazu o rozdzielczości $M \times N$ pikseli sprowadza się do przypisania każdemu z $M \cdot N$ punktów wartości natężenia jego jasności. Wielkość tę zwyczajowo poddajemy dyskretyzacji do wartości całkowitych z przedziału od 0 do 255, co umożliwia przechowanie jej w dokładnie jednym bajcie pamięci komputera.

Określamy funkcję jasności $l : \{0, \dots, M-1\} \times \{0, \dots, N-1\} \rightarrow \{0, \dots, 255\}$, która każdemu pikselowi obrazu $(x, y) \in \{0, \dots, M-1\} \times \{0, \dots, N-1\}$ przyporządkowuje dyskretną wartość natężenia jego jasności.

Niech:

$$L = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} l(x, y).$$

Wówczas funkcja $l' = \frac{L}{L}$ jest dyskretyzacją funkcji gęstości pewnego rozkładu wektora losowego w przestrzeni dwuwymiarowej. Możemy zatem postrzegać obraz w kryteriach rozkładu statystycznego.

Niech \mathbf{x} będzie wektorem losowym w przestrzeni d -wymiarowej $\Omega \subseteq \mathbb{R}^d$ ($d \geq 1$). Mówimy, że **wektor losowy \mathbf{x} ma rozkład w postaci skończonej mieszanki** (ang. *finite mixture distribution*), jeżeli funkcja gęstości jego rozkładu jest następująca:

$$p(\mathbf{x}) = p_1(\mathbf{x})P_1 + \dots + p_K(\mathbf{x})P_K \quad (\mathbf{x} \in \Omega, K \geq 1), \quad (1)$$

gdzie

$$P_i \geq 0, i = 1, \dots, K; \quad \sum_{i=1}^K P_i = 1$$

oraz

$$p_i(\cdot) \geq 0, i = 1, \dots, K; \quad \int_{\Omega} p_i(\mathbf{x}) d\mathbf{x} = 1.$$

Zmienne P_1, \dots, P_K nazywać będziemy **wagami**, zaś funkcje $p_1(\cdot), \dots, p_K(\cdot)$ — **składnikami mieszanki**. Dla mieszanek jednorodnych (tzn. takich, których składniki są funkcjami gęstości tego samego typu) wygodnie będzie zapisać Równanie 1 w postaci

$$p(\mathbf{x}|\Theta) = \sum_{i=1}^K p_i(\mathbf{x}|\theta_i)P_i, \quad (2)$$

gdzie θ_i są wektorami parametrów i -tego rozkładu, zaś $\Theta = (\theta_1, \dots, \theta_K)$.

W niniejszej pracy rozważać będziemy *mieszanki gaussowskie* zadane wzorem

$$\forall_{i=1, \dots, K} \quad p_i(\mathbf{x}|\theta_i) = \frac{1}{(2\pi)^{\frac{d}{2}} \cdot |\Sigma_i|^{\frac{1}{2}}} \cdot \exp \left[-\frac{1}{2}(\mathbf{x} - \mathbf{m}_i)^T \Sigma_i^{-1} (\mathbf{x} - \mathbf{m}_i) \right], \quad (3)$$

gdzie $\theta_i = \{\theta_{i1}, \theta_{i2}\} = \{\mathbf{m}_i, \Sigma_i\}$ to (kolejno) wektor wartości średnich oraz macierz kowariancji rozkładu normalnego.

2 Klasteryzacja za pomocą sieci Kohonena (SOM)

Niech $d \geq 1$ oraz $\{\mathbf{x}; \mathbf{x} = (x_1, \dots, x_d) \in \Omega \subseteq \mathbb{R}^d\}$ będzie próbką z pewnego rozkładu prawdopodobieństwa (ciągłego lub dyskretnego), zaś $\{\mathbf{m}\} = \{\mathbf{m}_i \in \mathbb{R}^d; i = 1, \dots, K\}$ ustalonym zbiorem tzw. *wektorów kotwicowych*.

Teselacją Voronoi'a [1] przestrzeni $\Omega \subseteq \mathbb{R}^d$ nazywamy procedurę jej podziału na K wypukłych podzbiorów $V_i(\{\mathbf{m}\})$ wyznaczonych przez wektory $\mathbf{m}_i \in \mathbb{R}^d$ następująco:

$$\forall_{i=1, \dots, K} \quad V_i(\{\mathbf{m}\}) = \{\mathbf{x} \in \Omega; \forall_{j \neq i} \|\mathbf{x} - \mathbf{m}_i\| < \|\mathbf{x} - \mathbf{m}_j\|\}, \quad (4)$$

gdzie $\|\cdot\|$ oznacza normę euklidesową. Uzyskany podział nazywamy *mozaiką Voronoi'a*.

Zauważmy, jakie znaczenie odgrywa odpowiedni dobór wektorów kotwicowych. Aby wynikowa mozaika Voronoi'a była reprezentatywna dla wyjściowego rozkładu $p(\mathbf{x})$ i użyteczna dla potrzeb klasteryzacji, musimy zadbać o to, by gęstość rozmieszczenia wektorów \mathbf{m}_i była proporcjonalna do gęstości $p(\mathbf{x})$ w danym regionie. Innymi słowy — chcemy, aby:

$$\forall_{i \neq j} \quad \mathbb{P}(\mathbf{x} \in V_i(\{\mathbf{m}\})) = \mathbb{P}(\mathbf{x} \in V_j(\{\mathbf{m}\})).$$

Sieci Kohonena (SOM, od ang. Self-organizing Maps [3, 4]) realizują koncepcję tzw. *samoorganizacji topologicznej*. Nacisk kładziony jest bowiem nie tylko na rozmieszczenie wektorów kotwicowych $\mathbf{m}_i \in \mathbb{R}^d$ traktowanych niezależnie, ale również ich wzajemne położenie.

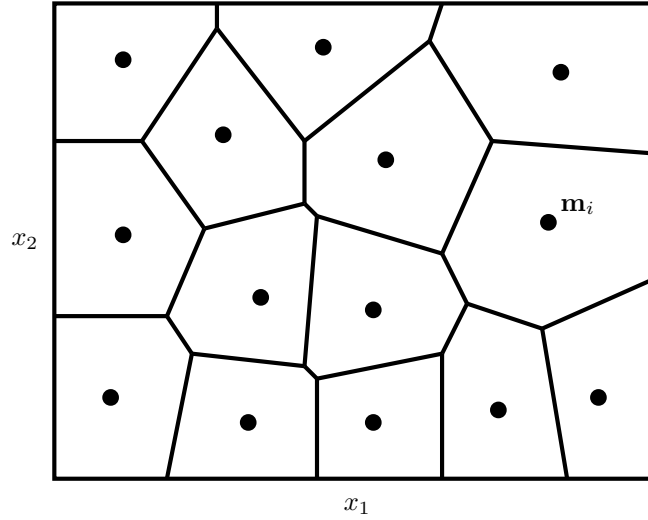
Niech G będzie dowolnym grafem spójnym, którego wierzchołkami są wszystkie K wektory kotwice \mathbf{m}_i . Niech ponadto $d(i, j)$ będzie pewną grafową miarą odległości w G . Określmy funkcję $\tilde{h} : \mathbb{R}_+ \cup \{0\} \rightarrow [0, 1]$ o własnościach:

- (a) \tilde{h} jest ściśle malejąca,
- (b) $\tilde{h}(0) = 1$,
- (c) $\lim_{z \rightarrow \infty} \tilde{h}(z) = 0$.

Dla tak zadanego odwzorowania \tilde{h} zdefiniujemy *funkcję sąsiedztwa* (ang. *neighbourhood function*) $h : \{(i, j); i, j = 1, \dots, K, i \neq j\} \rightarrow \mathbb{R}$ następująco:

$$\forall_{i \neq j} \quad h(i, j) = \tilde{h} \left(\frac{d(i, j)}{\sigma} \right), \quad (5)$$

gdzie $\sigma > 0$ jest pewnym parametrem algorytmu. W każdej iteracji wektor \mathbf{x} leżący w i -tej komórce Voronoi'a porusza wszystkie j -te wektory kotwice, dla których $d(i, j) < \sigma$.



Rysunek 1: Przykładowa mozaika Voronoi'a dla $d = 2$. Rysunek zaczerpnięto z [1].

2.1 Algorytm SOM

- *Dane:* Liczba $K \in \mathbb{N}$, skończony zbiór wektorów losowych $\mathbf{x} \in \Omega \subseteq \mathbb{R}^d$ zgodny z rozkładem prawdopodobieństwa $p(\mathbf{x})$, funkcja sąsiedztwa $h(i, j)$ oraz parametry σ, η .
- *Wynik:* Zbiór wektorów kotwicowych $\{\mathbf{m}\} = \{\mathbf{m}_i \in \mathbb{R}^d; i = 1, \dots, K\}$ tworzących strukturę grafu, reprezentatywny dla $p(\mathbf{x})$.

Wybierz losowo wszystkie wektory $\mathbf{m}_i \in \mathbb{R}^d$, a następnie powtarzaj:

1. Wylosuj wektor $\mathbf{x} \in \Omega$ zgodnie z rozkładem $p(\mathbf{x})$.
2. Znajdź komórkę Voronoi'a zawierającą \mathbf{x} , to znaczy wyznacz indeks i taki, że:

$$\forall_{j \neq i} \quad \|\mathbf{x} - \mathbf{m}_i\| < \|\mathbf{x} - \mathbf{m}_j\|.$$

3. Przesuń wszystkie wektory kotwicowe \mathbf{m}_j w stronę \mathbf{x} według następującej formuły:

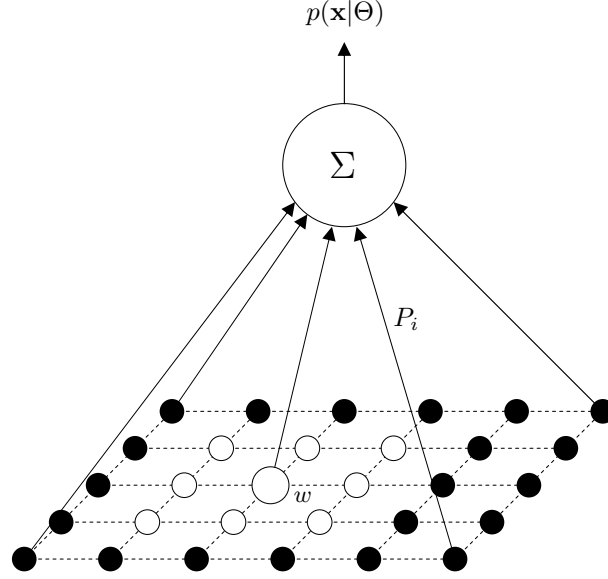
$$\mathbf{m}_j \leftarrow \mathbf{m}_j + \eta(\mathbf{x} - \mathbf{m}_j)h(i, j). \quad (6)$$

3 Samoorganizujące się sieci mieszankowe

Samoorganizujące się sieci mieszankowe (ang. *Self-Organizing Mixture Networks*, w skrócie *SOMN*) [6] są uogólnieniem pojęcia *Bayesian Self-Organizing Maps (BSOM)* [5]. Cechuje je dwuwartstwowa struktura, pozwalająca połączyć zalety sieci Kohonena (SOM) z możliwością uczenia macierzy kowariancji oraz wag dla każdej ze składowych mieszanki.

3.1 Architektura sieci

Na Rysunku 2 przedstawiono schematycznie dwuwartstwową strukturę sieci. Niższa warstwa funkcjonuje na zasadach bardzo zbliżonych do sieci Kohonena. Wyższa natomiast powstaje poprzez sumowanie wszystkich węzłów z uwzględnieniem wag P_i , z jakimi węzły te występują w wynikowej mieszance (por. Równanie 2), dając w wyniku wartość $p(\mathbf{x}|\Theta)$ dla dowolnego $\mathbf{x} \in \Omega$.



Rysunek 2: Schematyczne przedstawienie SOMN zaczerpnięte z [6].

W praktycznym zastosowaniu SOMN do reprezentacji obrazów cyfrowych, przekształcenie funkcji gęstości rozkładu na wartość jasności każdego kolejnego piksela jest niewystarczające. Ignoruje ono bowiem informację o średniej jasności całego obrazu, traktując jednakowo obrazy utrzymane w jasnych jak i ciemnych odcieniach, odwzorowując jedynie kontrast pomiędzy najjaśniejszym i najciemniejszym pikselem. Prowadzić to może do błędnej reprezentacji wyjściowego obrazu (por. Rysunek 3).

Celem uniknięcia powyższego błędu, należy dla wejściowego obrazu o rozdzielczości $M \times N$ wyznaczyć sumę jasności wszystkich $M \cdot N$ pikseli. Pamiętając o wspomnianej wcześniej kwestii rozbieżności w reprezentacji odcieni, tzn. $l(\cdot, \cdot) = 0$ oznacza odcień najciemniejszy, zaś gęstość $p(\cdot|\cdot) = 0$ — odcień najjaśniejszy — obliczymy w istocie sumę różnic postaci:

$$L' = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (255 - l(x, y)).$$

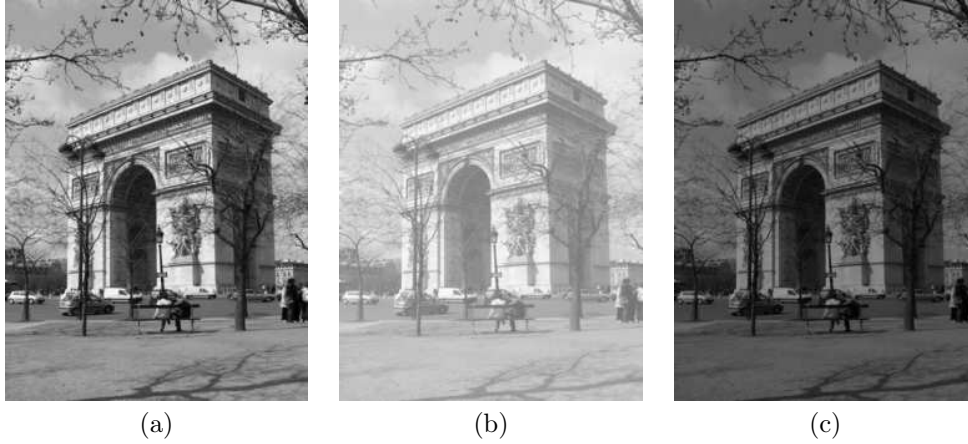
3.2 Uczenie SOMN

Uczenie SOMN jest procesem iteracyjnym. W chwili inicjalizacji sieć składa się z \tilde{K} węzłów, gdzie $\tilde{K} \geq K$. Jeżeli wartość K jest znana *a priori*, przyjmujemy zwykle $\tilde{K} = K$. W przeciwnym razie szacujemy ją z góry, aby uniknąć błędu wynikającego ze zbyt niewielkiej liczby składników mieszanki, co w oczywisty sposób obniżałoby dokładność przybliżenia rozkładu.

W każdym z $T \in \mathbb{N}$ kroków uczenia sieci losujemy próbkę $\mathbf{x}(t)$ zgodnie z przybliżanym rozkładem $p(\mathbf{x})$, w konsekwencji czego nieznacznie modyfikujemy jej stan. Aktualną postać mieszanki oznaczmy jako:

$$\hat{p}(\mathbf{x}|\hat{\Theta}) = \sum_{i=1}^{\tilde{K}} \hat{p}_i(\mathbf{x}|\hat{\theta}_i) \hat{P}_i, \quad (7)$$

gdzie $\hat{\Theta} = (\hat{\theta}_1, \dots, \hat{\theta}_{\tilde{K}})$.



Rysunek 3: Odzworowanie jasności obrazu: (a) obraz wyjściowy, (b) i (c) przykłady niepoprawnego odwzorowania.

Niech $t = 1, \dots, T$ będzie numerem bieżącej iteracji. Wektorowi $\mathbf{x}(t)$ przyporządkowujemy węzeł, dla którego wartość prawdopodobieństwa warunkowego (*a posteriori*) obliczanego według formuły:

$$\hat{P}(i|\mathbf{x}) = \frac{\hat{P}_i \hat{p}_i(\mathbf{x}|\hat{\theta}_i)}{\hat{p}(\mathbf{x}|\hat{\Theta})} \quad (8)$$

jest największa. Węzeł taki zwyczajowo nazywamy *zwycięzcą* (ang. *winner*). Na Rysunku 2 przedstawiony jest jako największe niewypełnione koło oznaczone literą w . Pozostałe niewypełnione koła symbolizują węzły znajdujące się w bezpośrednim sąsiedztwie zwycięzcy.

3.3 Algorytm SOMN

- *Dane:* Liczba $\tilde{K} \in \mathbb{N}$, skończony zbiór wektorów losowych $\mathbf{x} \in \Omega \subseteq \mathbb{R}^d$ zgodny z rozkładem prawdopodobieństwa $p(\mathbf{x})$, odległość grafowa $d(i, j)$, ciąg nierosnący $(a_t)_{t=1}^\infty$ zbieżny do zera oraz skończony ciąg niemalejący $(\delta_t)_{t=1}^T$ o tej własności, że $\delta_T = 0$.
- *Wynik:* Graf węzłów sieci (z parametrami $\hat{P}_i, \hat{\theta}_i$, gdzie $\hat{\theta}_i = \{\hat{\mathbf{m}}_i, \hat{\Sigma}_i\}$ dla $i = 1, \dots, \tilde{K}$) reprezentatywny dla $p(\mathbf{x})$.

Zainicjuj dowolną metodą wszystkie parametry $\hat{P}_i, \hat{\theta}_i$, a następnie dla $t = 1, \dots, T$ powtarzaj:

1. Wylosuj wektor $\mathbf{x} \in \Omega$ zgodnie z rozkładem $p(\mathbf{x})$.
2. Znajdź indeks $i \in \{1, \dots, \tilde{K}\}$ węzła, dla którego wartość:

$$\hat{P}(i|\mathbf{x}) = \frac{\hat{P}_i \hat{p}_i(\mathbf{x}|\hat{\theta}_i)}{\hat{p}(\mathbf{x}|\hat{\Theta})}$$

jest największa.

3. Dla wszystkich węzłów $j \in \{1, \dots, \tilde{K}\}$ takich, że $d(i, j) \leq \delta(t)$ przypisz:

$$\begin{aligned} \hat{\mathbf{m}}_j &\leftarrow \hat{\mathbf{m}}_j + a(t) \hat{P}(j|\mathbf{x}) [\mathbf{x} - \hat{\mathbf{m}}_j], \\ \hat{\Sigma}_j &\leftarrow \hat{\Sigma}_j + a(t) \hat{P}(j|\mathbf{x}) \left\{ [\mathbf{x} - \hat{\mathbf{m}}_j] [\mathbf{x} - \hat{\mathbf{m}}_j]^T - \hat{\Sigma}_j \right\}, \\ \hat{P}_j &\leftarrow \hat{P}_j + a(t) \left[\hat{P}(j|\mathbf{x}) - \hat{P}_j \right]. \end{aligned}$$

4. Dla wszystkich węzłów $j = 1, \dots, \tilde{K}$ wykonaj normalizację:

$$\hat{P}_j(t) \leftarrow \frac{\hat{P}_j(t)}{\sum_{k=1}^{\tilde{K}} \hat{P}_k(t)}.$$

4 Przykłady

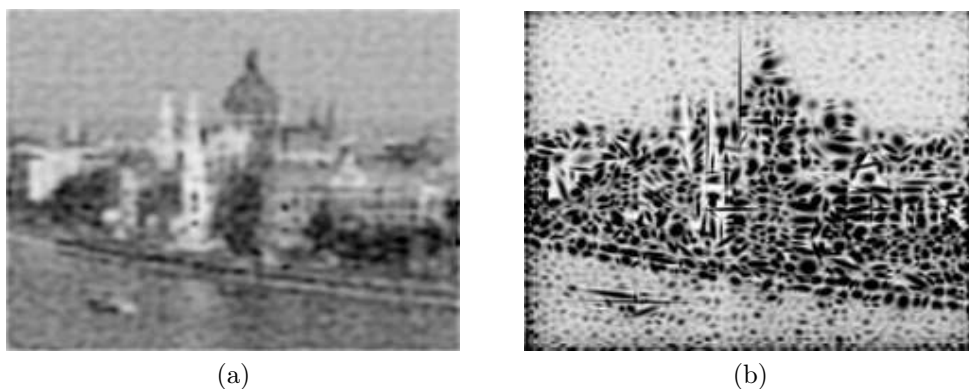
Testy zostały wykonane przy wykorzystaniu programu (stanowiącego integralną część pracy [2]) implementującego opisany wariant algorytmu SOMN. Program ten dostarcza możliwość skonfigurowania następujących parametrów uczenia sieci: Suwak *Learn* pozwala wybrać wartość z przedziału $0,01 \div 1,00$ z dokładnością $0,01$. Wyrazy ciągu współczynników sterujących tempem uczenia dla wektorów średnich $\hat{\mathbf{m}}_i$ oraz macierzy kowariancji $\hat{\Sigma}_i$ określone są wzorem: $a(t) = \text{Cooling factor}(t) \cdot \text{Learn}$. Dla zniwelowania efektu zbyt gwałtownych wahań wag węzłów SOMN w procesie uczenia, wprowadzono dodatkowy parametr *Weight* z przedziału $0,00001 \div 0,00100$ określany z dokładnością $0,00001$. Ciąg współczynników determinujących tempo uczenia wag węzłów wyrażony jest więc wzorem: $\alpha(t) = a(t) \cdot \text{Weight}$.

W pierwszej kolejności posłużymy się fotografią gmachu węgierskiego parlamentu w Budapeszcie:



Rysunek 4: Obraz wejściowy w skali szarości (640×480 pikseli).

Zdjęcie przesyccone jest detalami architektonicznymi, dlatego jego poprawne odwzorowanie wymaga precyzyjnego doboru parametrów. Rysunek 5(a) pokazuje, że 100 tysięcy iteracji to zdecydowanie zbyt mało, by oddać szczegóły budowli. Obraz jest mocno rozmyty i nieczytelny. Z kolei Rysunek 5(b) przedstawia wynik działania SOMN po pięciu milionach iteracji z parametrami: $\text{Learn} = 0,15$ oraz $\text{Weight} = 0,00005$, które w przypadku tego obrazu okazały się być zbyt wysokie. Zważywszy, że uzyskanie wyniku w niniejszym przypadku pochłonęło nieco ponad 7 godzin, widzimy, że właściwe dobranie parametrów staje się procesem czasochłonnym.



Rysunek 5: Sieć rozmiaru 100×100 węzłów: (a) po 100 tys. iteracji, (b) po 5 mln. iteracji ze zbyt wysokimi parametrami uczenia.

Tabela 1 przedstawia zestawienie najlepszych uzyskanych wyników. Rezultat dla sieci 200×200 wygląda satysfakcjonująco, jednak należy zwrócić uwagę, że 5 milionów iteracji algorytmu pochłonęło znacznie ponad jedną dobę.





Liczba iteracji	Rozmiar sieci	
	100×100	200×200
1 000 000	 1 godz. 22 min.	 6 godz. 29 min.
5 000 000	 7 godz. 16 min.	 31 godz. 21 min.

Tabela 1: Zestawienie rezultatów i czasów wykonywania algorytmu SOMN dla podanych rozmiarów sieci oraz liczby iteracji na przykładzie fotografii gmachu węgierskiego parlamentu.

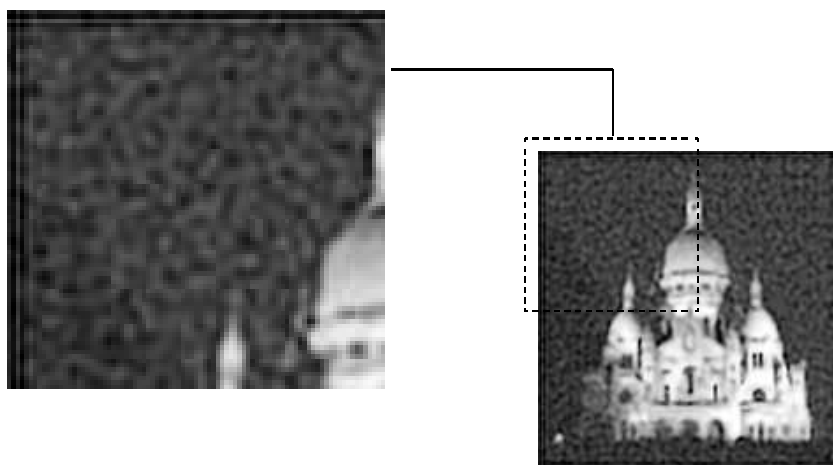
Zupełnie inny charakter cechuje kolejną fotografię przedstawiającą paryską bazylikę Sacré-Cœur (Rysunek 6). Zdjęcie zostało wykonane nocą, zatem odwzorowanie ciemnego nieba skontrastowanego z jasną elewacją kościoła stanowi kompletnie odmienne zadanie dla SOMN w stosunku do poprzedniego przykładu.

Trudność w odwzorowaniu dużych pól jednakowego odcienia skutkuje występowaniem artefaktów, widocznych na poniższym rysunku:

Artefakty szczególnie widoczne są w przypadku sieci o rozmiarach 100×100 węzłów, mniej zaś dla rozmiaru 200×200 (por. Tabela 2).



Rysunek 6: Obraz wejściowy w skali szarości (460×480 pikseli).



Rysunek 7: Artefakty widoczne na dużych polach tego samego odcienia.

Jako ostatni przykład weźmiemy pod uwagę fotografię pomnika Mikołaja Kopernika w Toruniu (Rysunek 8). Podobnie jak w poprzednich przypadkach, najlepsze odwzorowanie detali uzyskujemy po pięciu milionach iteracji algorytmu dla sieci o rozmiarze 200×200 węzłów. Odwzorowanie detali rysów twarzy, jak również liści drzew na drugim planie jest bliskie oryginałowi. Brak rozległych pól jednakowego odcienia powoduje, że na uzyskanym obrazie nie odnajdujemy niepożądanych artefaktów. Zestawienie uzyskanych rezultatów przedstawiono w Tabeli 3.

5 Konkluzje

Niniejsza publikacja oparta jest w znacznej mierze na pracy magisterskiej autora [2] napisanej pod kierunkiem prof. Tomasza Schreiber na Uniwersytecie Mikołaja Kopernika w Toruniu.

Jako dane wejściowe dla procesu klasteryzacji z użyciem SOMN wykorzystane zostały obrazy cyfrowe w skali szarości. Warto zwrócić uwagę, że reprezentacja obrazu przy pomocy sieci neuronów jest bardzo użyteczna dla potrzeb analizowania wzorców (ang. *pattern matching*), eksploracji danych (ang. *data mining*) czy kompresji. Przedstawione przykłady pokazują, że odpowiednio nauczone SOMN mogą być z powodzeniem





Liczba iteracji	Rozmiar sieci	
	100 × 100	200 × 200
1 000 000	 1 godz. 4 min.	 5 godz. 57 min.
5 000 000	 6 godz. 32 min.	 28 godz. 11 min.

Tabela 2: Zestawienie rezultatów i czasów wykonywania algorytmu SOMN dla podanych rozmiarów sieci oraz liczby iteracji na przykładzie fotografii bazyliki Sacré-Cœur.



Rysunek 8: Obraz wejściowy w skali szarości (640 × 480 pikseli).





Liczba iteracji	Rozmiar sieci	
	100 × 100	200 × 200
1 000 000	 1 godz. 31 min.	 6 godz. 27 min.
5 000 000	 7 godz. 8 min.	 31 godz. 55 min.

Tabela 3: Zestawienie rezultatów i czasów wykonywania algorytmu SOMN dla podanych rozmiarów sieci oraz liczby iteracji na przykładzie fotografii pomnika Mikołaja Kopernika.

wykorzystane do reprezentacji obrazów cyfrowych w skali szarości.

Literatura

- [1] A. C. C. Coolen, R. Kühn, and P. Sollich. *Theory of Neural Information Processing Systems*. Oxford University Press, 2005.
- [2] P. Filipiak. Samoorganizujące się sieci mieszkankowe w reprezentacji obrazów cyfrowych. Master's thesis, Uniwersytet Mikołaja Kopernika w Toruniu, 2010.
- [3] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1478, 1990.
- [4] T. Kohonen. *Self-Organizing Maps*. Springer, 2001.
- [5] H. Yin and N. M. Allinson. Bayesian learning for self-organizing map. *Electronic Letters*, 33(4):304–305, 1997.
- [6] H. Yin and N. M. Allinson. Self-organizing mixture networks for probability density estimation. *IEEE Transactions on Neural Networks*, 12(2):405–411, 2001.