Model FSM w zastosowaniu do klasyfikacji.

Rafał Adamczak, Włodzisław Duch

Katedra Metod Komputerowych, Uniwersytet Mikołaja Kopernika,
ul. Grudziądzka 5, 87-100 Toruń, e-mail:{raad, duch}@phys.uni.torun.pl

Streszczenie

Sieci FSM mają prostą strukturę, podobną do sieci RBF, jednak dzięki użyciu separowalnych funkcji transferu, zamiast radialnych, mają szersze zastosowanie. Omówiono architekturę, inicjalizację, konstruktywistyczny algorytm uczenia, optymalizację sieci, obroty zlokalizowanych funkcji transferu, rozpoznawanie wektorów w czasie klasyfikacji oraz możliwości zastosowań do ekstrakcji reguł.

1. Wprowadzenie

W dążeniu do zrozumienia ludzkiej inteligencji i umysłu wyróżnić można dwa paradygmaty. Pierwszy z nich, sztuczna inteligencja, bazuje na przetwarzaniu symbolicznym, a więc źródłem są tu procesy poznawcze wysokiego poziomu, drugi, sieci neuronowe wywodzą się z neurodynamiki i inspirowane są przez strukturę neuronową mózgu. Sieci neuronowe poprzez swoją odmienność w podejściu do problemu inteligencji i umysłu wnoszą nowe możliwości do dziedziny sztucznej inteligencji. Dotychczas sieci neuronowe wydają się być najlepszym rozwiązaniem dla zadań poznawczych niskiego poziomu, takich jak problemy widzenia czy rozpoznawania mowy, lub też dla prostych zadań klasyfikacyjnych, przez co są one w pewien sposób ograniczone w ich możliwościach realizacji predefiniowanych struktur wiedzy i w użyciu tych struktur w sekwencyjnym procesie przyczynowym. Nie ma wątpliwości, że wyższe funkcje poznawcze są rezultatem aktywności mózgu, a więc powinno być możliwe ich odtworzenie przez sieci neuronowe [2], [4], [15]. Jasnym jest, że obecny rozwój sieci neuronowych jest powiązany z brakiem modularności i raczej niewielką złożonością modelu niż z ich wewnętrznym ograniczeniem jako modelu. Dużo już wiadomo o szczegółach procesów neuronowych odpowiedzialnych za działanie mózgu i neurodynamikę. Obecnie nawet niektóre działy psychologii korzystają z osiągnięć neurodynamiki [16].

Czy jest możliwe zrozumienie procesów myślowych bezpośrednio z procesów neuronowych w mózgu? Wydaje się, że nie. Nawet w chemii i fizyce koncepcje fenomenologiczne, które nie są łatwo redukowalne do fundamentalnych oddziaływań wciąż są w użyciu. Teorie makroskopowe w zasadzie są redukowalne do mikroskopowych, w praktyce jednak bardziej owocne jest przybliżenie fenomenologiczne do złożonych systemów. Języki neurologii i psychologii są zupełnie różne. Wydaje się jednak, że musi istnieć teoria pozwalająca uzyskać koncepcje psychologiczne jako aproksymacje neurodynamicznego podejścia do działania mózgu. Głównymi celami takiej teorii są:

1. Wprowadzenie aproksymacji do neurodynamiki, przy spełnieniu faktów neurobiologicznych, prowadzącej do nowej (matematycznej) koncepcji bezpośrednio opisującej stany poznawcze.
2. Użycie tych koncepcji jako języka do zbudowania teorii systemu poznawczego.
3. Zastosowanie tej teorii do wyjaśnienia cech procesu poznawczego człowieka, takich jak identyfikacja, asocjacja, generalizacja, różne stany umysłu.
4. Konstrukcja systemu adaptacyjnego odpowiadającego specyfikacjom, systemu, który będzie uczył się z przykładów i ogólnych praw, wnioskował, używał naturalnego języka, i który będzie spełniał inne funkcje poznawcze

Zgodnie z duchem Allana Newella "Unified theories of cognition" teoria powinna być wspomagana przez oprogramowanie pozwalające zweryfikować przesłanki i modele przez nią utworzone. System SOAR stworzony przez Newella i jego współpracowników bazuje na tworzeniu reguł, ale nie ma nic wspólnego z neurobiologią. Może być użyteczny w modelowaniu pewnych funkcji poznawczych ale nie pomoże nam zrozumieć powiązania tych procesów z dynamiką zachodzącą w mózgu. System FSM [1] jest inspirowany na działaniu mózgu i dostarcza możliwości nie tylko modelowania ale również rozumienia procesów poznawczych i ich relacji z dynamiką mózgu.

Modele funkcji mózgu wymagają szeregu aproksymacji. W pierwszym kroku wprowadza się model neuronu jako urządzenia elektrycznego [15] przez co dokonuje się drastycznego uproszczenia z punktu widzenia procesów biochemicznych i bioelektrycznych. Ta aproksymacja pozwala użyć w modelu tylko kilku parametrów takich jak: próg wzbudzenia i wagi synaptyczne. Jest również konieczna do zrozumienia procesów poznawczych wyższego poziomu. Procesy poznawcze niskiego poziomu, realizowane najczęściej przez różne mapy topograficzne, definiują cechy wewnętrznej reprezentacji. Te cechy reprezentują wiele typów danych: analogowe sygnały sensorowe, zmienne lingwistyczne, liczby, obrazy. Prawdziwe obiekty umysłu składają się głównie z przetworzonych wstępnie danych sensorowych, reprezentacji obrazkowej, działania percepcyjnego. Obiekty umysłu znajdują się w przestrzeni umysłu, która jest dla nich zbiornikiem. Naturalną praktyczną realizacją tej idei jest modularna sieć neuronowa z węzłami specjalizującymi się w opisie grup obiektów w przestrzeni umysłu. Węzły takiej sieci nie reprezentują pojedynczych neuronów ale raczej uśrednioną aktywność zbioru komórek neuronowych. Tego typu sieć może być traktowana jako sieć neuronowa bazująca na zlokalizowanym przetwarzaniu, lub też jako rozmyty system ekspertowy, którego wiedza zapisana jest w postaci zbiorów rozmytych.

2. FSM

Model FSM (Feature Space Mapping) oparty jest na estymacji gęstości rozkładu prawdopodobieństwa prezentowanych danych. Patrząc z różnych perspektyw można go uważać za sieć neuronową podobną do sieci z radialnymi funkcjami bazowymi (chociaż funkcje nie musza być radialne, tylko separowalne), system neurorozmyty (aktywność węzłów sieci z separowalnymi funkcjami transferu można przeanalizować używając koncepcji zbiorów rozmytych), rodzaj systemu opartego na śladach pamięci (memory-based system), system samoorganizujący się a nawet jądro systemu eksperckiego, wykorzystującego wiedzę rozmytą lub oferującego przydatne heurystyki w procesie szukania rozwiązań. Składowe wektora wejściowego i wyjściowego definiują razem przestrzeń cech. Na podstawie danych treningowych, stanowiących prototypy, w przestrzeni cech tworzy się rozmyte obiekty reprezentujące łączny rozkład prawdopodobieństwa dla wektorów wejściowych i wyjściowych. Rozkład ten może być użyty do aproksymacji nieznanych zależności lub do klasyfikacji.

Zastosowanie funkcji separowalnych pozwala na przeszukiwanie przestrzeni cech pomimo występowania wartości brakujących. Umożliwia to wyszukanie węzła najbardziej odpowiadającego aktualnym danym wejściowym, jak również dopełnienie brakującej wartości poprzez uzupełnienie faktu na podstawie teorii opisywanej przez wyszukany węzeł. Wiedza zapisana w przestrzeni cech jest więc używana w procesie przeszukiwania w taki sam sposób jak wiedza heurystyczna używana jest przez eksperta do zmniejszenia przestrzeni poszukiwań.
Model FSM może być również zastosowany do odkrywania wiedzy, praw, które mogą zostać wydedukowane z przykładów. Prawa te mogą być reprezentowane przez funkcje gaussowskie, wówczas rozmycie funkcji gaussowskiej odpowiadającej danemu faktowi jest miarą zaufania do tego faktu, gdzie zaufanie bazuje na liczbie zaprezentowanych przykładów.
Topograficzna struktura przestrzeni cech może podlegać ograniczeniom związanym z symboliczną wiedzą aprioryczną (zależności pomiędzy elementami wektora wejściowego). Szkielet tej struktury tworzy się korzystając z metod inicjalizacji opartych na klasteryzacji, a szczegóły ustalają się w wyniku procesu uczenia. W pewnym sensie jest to więc meta-model, gdyż możliwa jest różna specyfikacja jego matematycznych aspektów. Pierwotną inspiracją do jego stworzenia była próba wprowadzenia pośredniego poziomu opisu funkcji poznawczych jako przybliżenia do neurodynamiki [2]: wysoka aktywność neuronów wysyłających impulsy opisywana jest przez dużą gęstość prawdopodobieństwa w tym obszarze przestrzeni, który odpowiada kombinacji cech koniecznych do wywołania takiej aktywności. Probabilistyczny punkt widzenia jest bardzo przydatny do opisu sieci neuronowych [3].

W tej pracy ograniczymy się tylko do zastosowania modelu FSM do zagadnień klasyfikacji pod nadzorem. Przez pojęcie klasyfikacji rozumie się dzielenie dowolnego zbioru elementów na grupy, do których zalicza się elementy różniące się, ale podobne tj. mające własności wyróżniające daną grupę. Zbiór elementów należących do jednej grupy nazywany jest klasą, a każdy element klasy – obiektem. W klasyfikacji pod nadzorem struktura kategorii jest znana, czyli dysponuje się charakterystyką klas, z których pochodzą obiekty. Klasyfikatory oparte na sieciach neuronowych na podstawie zadanego zbioru N przypadków, zwanego ciągiem treningowym ($\mathbf{X}_i$, C_i), i= 1..N, gdzie $\mathbf{X}$ jest m-wymiarowym wektorem cech opisującym dany obiekt, natomiast C_i jest odpowiednio zakodowaną etykietą klasy, dopasowują parametry adaptacyjne **W (**proces ten nazywa się umownie „uczeniem") w taki sposób, by dokonać poniższego, przybliżonego mapowania

$$f(\mathbf{X}_i; \mathbf{W}) = C_i$$

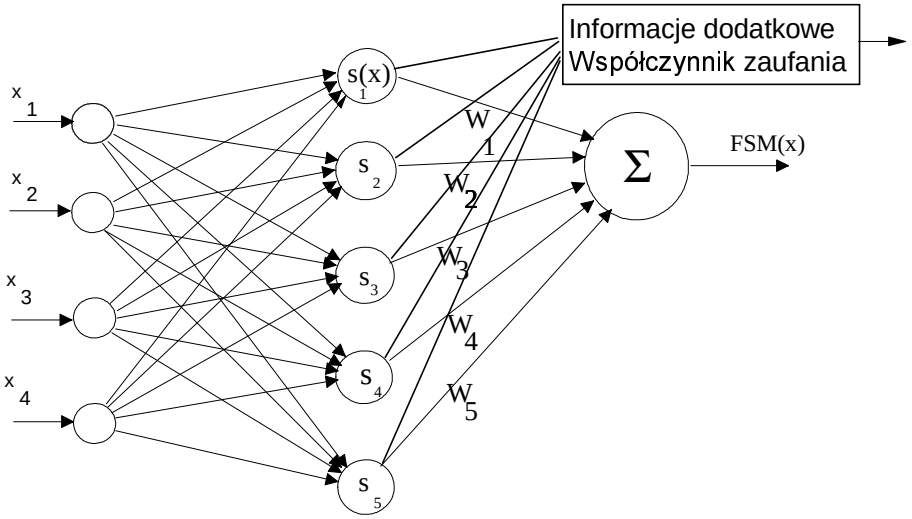
gdzie f jest funkcją, którą realizuje sieć. Jest to odwzorowanie przybliżone, gdyż nie wymagamy zwykle by sieć dawała na wyjściu dokładnie wartość C_i a jedynie wartość pozwalająca wyraźnie odróżnić C_i od C_j. Ilość parametrów adaptacyjnych jak i algorytm uczenia zależą od architektury sieci.

3. Architektura sieci FSM.

Poważnym problemem wielu algorytmów realizujących modele sieci neuronowych jest taki dobór architektury sieci (liczby ukrytych neuronów, liczby warstw ukrytych, topologii połączeń) aby uzyskać jak najlepszą generalizację. Istnieje wiele technik optymalizacji architektury prowadzących do tego celu. Obszerna klasa metod oparta jest na algorytmach konstruktywistycznych, próbujących wbudować optymalizację architektury sieci w algorytm uczenia. Stosuje się trzy techniki konstruowania sieci:

1. Rozpoczyna się z siecią posiadającą dużą (nadmiarową) liczbę węzłów, które w miarę możliwości są usuwane podczas uczenia.
2. Tworzy się sieć od zera tzn. na początku istnieje tylko warstwa wejściowa i wyjściowa, natomiast warstwy ukrytej nie ma wcale, lub też jest, ale z niewielką liczbą węzłów. W trakcie uczenia stopniowo dostawiane są kolejne neurony oraz ewentualnie kolejne warstwy ukryte. Do tej kategorii zaliczyć można takie modele sieci jak korelacja kaskadowa, sieć RAN jak i wersję modelu FSM (jeśli nie stosujemy wstępnej inicjalizacji).
3. Sieć dopasowuje swoją strukturę do napływających danych, zmieniając złożoność przez dodawanie, usuwanie i łączenie neuronów. Sieć FSM należy do tej kategorii.

W sieci FSM występują tylko trzy warstwy neuronów: wejściowa, ukryta i wyjściowa. Liczba węzłów w warstwie wejściowej jest ustalona na początku procesu uczenia i jest równa wymiarowi wektorów wejściowych. W warstwie wyjściowej wystarczy tylko jeden węzeł (można też używać jednego węzła na jedną klasę), którego zadaniem jest określenie na podstawie wzbudzeń neuronów w warstwie ukrytej, do której klasy należy wektor wejściowy **X**. Węzły warstwy ukrytej połączone są ze wszystkimi neuronami warstwy wejściowej oraz z węzłem wyjściowym. Sieć FSM może mieć strukturę modułową, składającą się z kilku podsieci specjalizujących się w klasyfikacji danych wejściowych określonych przez cechy jednego typu [4], ale dla celów tutaj przedstawionych nie jest ona wykorzystywana.



Architektura sieci FSM

Wszystkie węzły w warstwie ukrytej realizują dowolną funkcję faktoryzowalną $G(\mathbf{X};\mathbf{D},\sigma)$. Warunek faktoryzowalności pozwala traktować każdy z wymiarów jako niezależny od pozostałych. Faktoryzowalne funkcje transferu maja postać:

$$G(\mathbf{X};\mathbf{R},\sigma)=\prod_i G_i(X_i;R_i,\sigma_i)$$

gdzie każdy ze składników sparametryzowany jest przez położenie i dodatkowy parametr, który dla funkcji zlokalizowanych pełni rolę dyspersji. Cechą charakterystyczną funkcji zlokalizowanych jest to, że wartość maksymalną osiągają one w swoim centrum, w miarę oddalania się od centrum wartość ta maleje do zera. Najbardziej znanym przykładem tego rodzaju funkcji jest funkcja gaussowska. Chociaż teoretycznie nie ma innych ograniczeń na funkcje

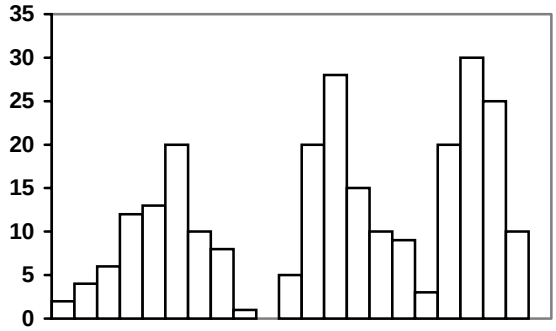
aktywacji w FSM, w praktyce jednak na razie stosowane były tylko funkcje zlokalizowane, o wartościach przeskalowanych do przedziału [0,1]. Zmiana wartości aktywacji w miarę oddalania się od centrum może być również skokowa co ma miejsce w przypadku użycia funkcji prostokątnej. Dotychczas w FSM stosowane były następujące funkcje aktywacji: gaussowskie, trapezoidalne, bicentralne [5], prostokątne i trójkątne.

Najlepsze rezultaty zarówno pod względem liczby powstałych węzłów jak i generalizacji uzyskiwane były dla funkcji gaussowskiej. Wybór funkcji aktywacji zależny jest od celu, który chcemy uzyskać, np. do stworzenia klasyfikatora, którego działanie możemy w prosty sposób prześledzić najlepiej nadają się reguły logiczne, które w FSM można uzyskać stosując funkcje prostokątne. Używając natomiast funkcji gaussowskiej można również uzyskać reguły, są to jednak reguły rozmyte, których interpretacja jest znacznie trudniejsza od klasycznych reguł logicznych. Każdy węzeł ukryty charakteryzowany jest przez położenie **R**, i rozmycie σ, które potrzebne są do wyliczenia funkcji aktywacji. Poza tym występują jeszcze wielkości, które nie mają wpływu na wartość wzbudzenia, ale są potrzebne w trakcie uczenia do określenia wielkości zmian pozostałych parametrów. Są to: masa m określająca liczbę wektorów klasyfikowanych przez dany węzeł i czas powstania τ_n, czyli numer epoki, w której dany węzeł powstał.

4. Inicjalizacja

Ponieważ algorytm uczenia w FSM jest algorytmem konstruktywistycznym to można przyspieszyć proces uczenia poprzez wstępne oszacowanie liczby węzłów ukrytych i odpowiednie ich zainicjowanie [6]. Oszacowanie to powinno być raczej optymistyczne tzn. liczba węzłów utworzona podczas inicjalizacji powinna być zaniżona. W przeciwnym wypadku wstępna inicjalizacja prowadzi do zbytniego rozdrobnienia i nawet po użyciu optymalizacji nie uzyska się dobrych rezultatów.

Dzięki stosowaniu funkcji zlokalizowanych do ustalenia liczby węzłów ukrytych jak i ich zainicjowania można użyć metod opartych na grupowaniu. W wyniku klasteryzacji otrzymujemy zbiór skupisk, prototypów, w środku których ustawiane są węzły sieci. Położenie każdego z tak powstałych neuronów obliczane jest jako średnia z położeń (dla każdego wymiaru osobno) wszystkich wektorów wpadających do danego skupiska (klastra). Rozmycia tych prototypów ustalane są na podstawie wielkości odpowiadających im klastrów. Gdy któreś z rozmyć ma wartość zero (wszystkie wektory mają tę samą wartość jednej z cech) przyjmowana jest minimalna wartość niezerowa. Z uwagi na to, że funkcja aktywacji jest iloczynem po poszczególnych wymiarach, jeśli choć jedna ze składowych ma wartość 0 to cała funkcja aktywacji jest zerowa. Tak ustalone rozmycia zostają następnie proporcjonalnie zwiększone, po to, aby sieć w początkowej fazie uczenia od razu klasyfikowała, bez względu na poprawność, wszystkie wektory z ciągu treningowego. Dzięki temu unika się przypadkowego wstawiania węzłów na początku uczenia.
Jednym z algorytmów stosowanych w FSM do inicjalizacji sieci jest drzewo decyzyjne budowane na podstawie histogramu utworzonego dla każdego wymiaru osobno.

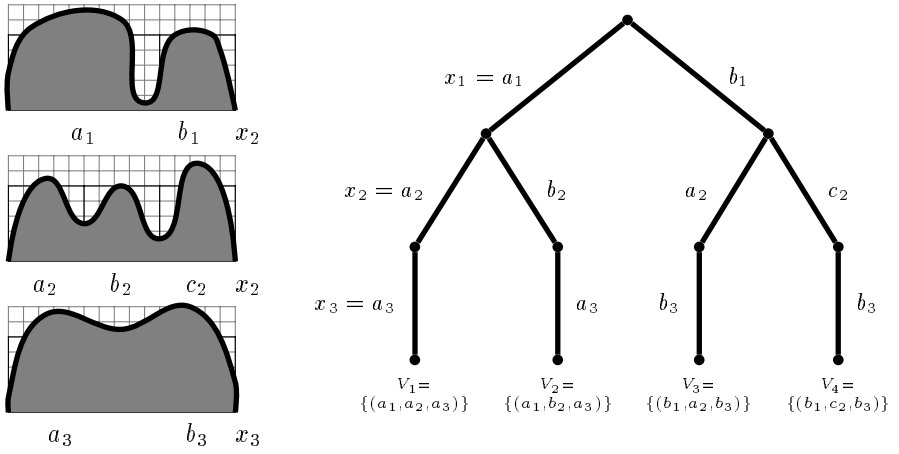


Dany wymiar dzielony jest na *k* przedziałów (oś X na rysunku). W każdym przedziale zliczana jest liczba wektorów, która do niego wpada (oś Y na rysunku). Każdy z przedziałów może być łączony z sąsiednimi przedziałami tworząc w ten sposób skupisko w przestrzeni jednowymiarowej. Jeżeli w danym wymiarze występuje wiele skupisk to będą one oddzielone pustymi przedziałami lub też między kolejnymi przedziałami wystąpi bardzo duża różnica w zawartości wektorów (przedstawiono to na powyższym rysunku, gdzie można wyróżnić 3 skupiska). Załóżmy, że klaster rozpoczyna się od *i* tego przedziału i zajmuje *l* przedziałów, każdy o szerokości *s*. Wówczas pozycja oraz rozmiar skupiska w oryginalnej przestrzeni może zostać przedstawiona w następujący sposób

$$C_x = X_{\min} + s(i + \frac{1}{2})$$

$$\sigma_x = \frac{1}{2} ls$$

$$s = \frac{| X_{\max} - X_{\min} |}{k}$$

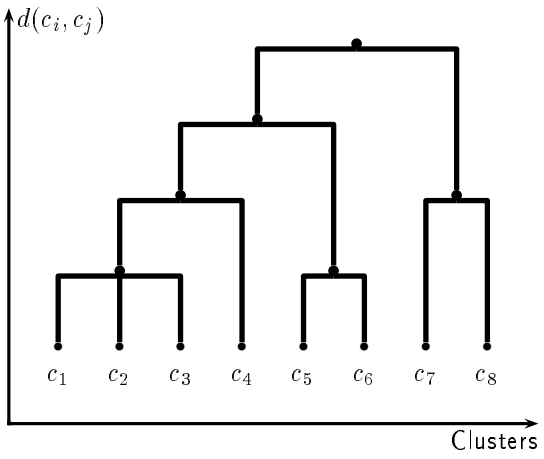


Drzewo decyzyjne

Każde z tych jednowymiarowych skupisk może zostać następnie zrzutowane na wiele rozseparowanych *N*-wymiarowych skupisk. Podczas inicjalizacji wektory analizowane są w każdym wymiarze niezależnie tak więc w pierwszym wymiarze dany wektor może należeć np. do

skupiska C_1^i, w drugim wymiarze do skupiska C_2^j aż wreszcie w *N*-tym wymiarze C_N^l. Każdy taki *N*-wymiarowy klaster może być przedstawiony jako łańcuch niżej wymiarowych skupisk $C_1^i -> C_2^j -> ... -> C_N^l$, stąd tego typu procedura tworzy drzewo decyzyjne. W liściach drzewa zliczane są wektory, które należą do danego *N*-wymiarowego skupiska. Gdy w przypadku któregokolwiek z wymiarów liczba klastrów jest zbyt duża, zmniejszona zostaje rozdzielczość *k* co powoduje zmniejszenie liczby klastrów dla danego wymiaru. Gdy wszystkie klastry są już utworzone obliczana jest odległość między nimi. Te skupiska między którymi odległość jest mniejsza od zadanego progu są łączone w jeden.

W powyższym drzewie decyzyjnym rozdzielczość histogramów jest stopniowo zwiększana. Innym rozwiązaniem opartym na metodzie dendrogramów, które czasem prowadzi do lepszych rezultatów, jest stopniowe zmniejszanie rozdzielczości. Początkowo każdy wektor ciągu treningowego jest oddzielnym skupiskiem. Wyznaczona zostaje macierz odległości między poszczególnymi skupiskami, i na jej podstawie najbliższe skupiska są ze sobą łączone. Cała procedura trwa tak długo aż liczba skupisk jest wystarczająco mała, lub też aż odległość między skupiskami jest większa od zadanej progowej. Poniższy rysunek przedstawia schematycznie ideę dendrogramów.



Dendrogram

Wadą metody dendrogramów jest konieczność tworzenia macierzy odległości co w przypadku dużej liczby wektorów treningowych prowadzi do znacznego spowolnienia procesu grupowania oraz dużego zapotrzebowania na pamięć. W takiej sytuacji dobrym rozwiązaniem jest zmniejszenie rozdzielczości danych.
Obliczenia z reguły wykonywane są dla liczb zmiennoprzecinkowych mających 4 bajtową reprezentację, co dla danych przenormalizowanych do przedziału [0,1] daje rozdzielczość $r = 10^{-7}$. Dla rozdzielczości r wszystkie dane z przedziału $[a, a+r]$ mają tą samą wartość.

Przed dokonaniem obcięcia dokładności wszystkie dane z ciągu treningowego przeskalowywane są do maksymalnego zakresu $[0,2^{32}]$

$$X_r = C(X_r - X_{\min}); C = \frac{2^{32}}{X_{\max} - X_{\min}}$$

Aby uzyskać rozdzielczość $r = \frac{1}{2^k}$ musimy zostawić k znaczących bitów w X_r. Można to wykonać przez przesunięcie bitów

$$X(k) = X_{\min} + \frac{1}{C}\left[Round\left(\frac{X_r}{2^{33-k}} \right)\right] * 2^{33-k}$$

Dla $k = 1$ wszystkie dane $\mathbf{X}$ są mapowane do dwóch klastrów, dla k=32 otrzymujemy dane oryginalne. Liczba znaczących bitów jest zwiększana tak długo aż osiągnięta zostanie dopuszczalna liczba skupisk.
W fazie grupowania dąży się do tego aby powstałe klastry opisywały jedynie oddzielne, wyraźne skupiska dla danej klasy, a nie ich złożoną strukturę wewnętrzną. Zapewnia to, że stosowane oszacowanie będzie „optymistyczne". Algorytm inicjalizacji automatycznie wyznacza liczbę skupisk dla danego problemu. Proces grupowania może być przeprowadzony dla każdej klasy osobno, co pozwala znaleźć skupiska charakterystyczne dla tej klasy oraz zainicjować sieć w taki sposób, aby każda z klas była reprezentowana przez co najmniej jeden węzeł. Można też proces grupowania przeprowadzić dla wszystkich klas jednocześnie. W tym przypadku podczas grupowania wykorzystuje się dodatkową informację (przynależność do klasy) i budując skupisko, analizuje się jego dokładność tzn. liczbę wektorów z interesującej nas klasy w stosunku do wszystkich wektorów należących do tego skupiska. W momencie gdy jakość skupiska znacznie się pogarsza przerywa się łączenie grup do tego skupiska. Jest to bardzo wygodny punkt stopu dla metody dendrogramów.
Tak więc proces uczenia, który następuje po inicjalizacji odbywa się zawsze dla sieci, która posiada już jakieś węzły ukryte.

5. Algorytm Uczenia

Na wejście sieci podawany jest wektor wybierany losowo z ciągu treningowego. Wybór dokonywany jest w taki sposób, aby każdy z wektorów w danej serii pojawił się dokładnie raz. Gdy cały ciąg treningowy zostanie już pokazany, procedura zostaje powtórzona. Każda taka seria nazywana jest epoką. Proces uczenia trwa tak długo, aż spełnione zostanie kryterium zakończenia opisane poniżej. Cały proces uczenia podzielić można na dwie fazy:

Faza 1: Modyfikacja parametrów istniejących prototypów tj. położeń neuronów, ich rozmyć, mas i czasów powstania.
Faza 2: Dostawianie nowych neuronów.

5.1. Faza I

Jeśli wektor wejściowy jest dostatecznie podobny do istniejącego węzła wystarczy zmodyfikować parametry, opisujące ten węzeł. Podobieństwo wyznaczane jest na podstawie odległości euklidesowej między wektorem wejściowym a poszczególnymi neuronami w sieci lub też na podstawie pobudzenia węzłów. W przypadku gdy dwa węzły reprezentujące różne klasy mają tą samą wartość funkcji aktywacji (w przypadku funkcji gaussowskich może się to zdarzać rzadko, jednak w przypadku funkcji prostokątnych bardzo często), wybierany jest ten, który należy do innej klasy niż wektor wejściowy, przez co w obliczaniu końcowej klasyfikacji mamy dla takiego wektora wejściowego błąd. Za najbardziej podobny uznawany jest ten węzeł, który jest najbliższy (w sensie euklidesowym, czyli $\min_k(\| \mathbf{X} - \mathbf{R}_k \|)$) do wektora wejściowego **X**. Jeżeli najbliższy jest węzeł z tej samej klasy co wektor wejściowy, dokonujemy optymalizacji jego parametrów. Jeżeli jest to neuron reprezentujący inną klasę niż wektor

wejściowy poszukujemy węzła, który ulega największemu wzbudzeniu. Jeśli ten węzeł należy do klasy wektora wejściowego to przeprowadzamy jego optymalizację, w przeciwnym razie rozpoczyna się druga faza uczenia, czyli dostawianie nowego neuronu.

W obliczaniu funkcji aktywacji skalowanie danych nie ma wpływu na wartość tej funkcji ponieważ każdy z wymiarów rozpatrywany jest niezależnie. Dzięki temu wkład każdego z wymiarów do funkcji aktywacji zawsze należy do przedziału [0,1]. Ponieważ jednak do określenia podobieństwa między węzłem a wektorem wejściowym używana jest odległość euklidesowa to przeskalowanie danych może mieć wpływ na proces uczenia, gdyż nowe węzły mogą być wstawiane w innym miejscu niż przed przeskalowaniem. Dlatego też przed rozpoczęciem uczenia dokonuje się standaryzacji danych.

Adaptacja parametrów polega na takim ich dopasowaniu, by zwiększyć wartość wzbudzenia węzła. Dokonujemy tego poprzez zmianę położenia węzła tak, aby w każdym wymiarze z osobna zmniejszyć odległość między nim a wektorem wejściowym:

$$R_i = R_i + \frac{\kappa}{m}\left(X_i - R_i\right)$$

Jednocześnie zwiększamy rozmycie węzła proporcjonalnie do odległości z wektorem wejściowym (podobnie jak położenie, dla każdego wymiaru osobno) oraz do aktualnego wzbudzenia tego węzła, i odwrotnie proporcjonalnie do czasu jego życia $\tau - \tau_n$:

$$\sigma_i = \sigma_i + \gamma \frac{(1 - G(X; R, \sigma)) \mid X_i - R_i \mid}{1 + (\tau - \tau_n)/\Lambda}$$

Czas życia jest to różnica aktualnego numeru epoki τ i czasu powstania węzła τ_n (czyli numeru epoki, w której węzeł powstał). Im starszy węzeł, tym trudniej jest zmieniać jego parametry adaptacyjne. Oczywiście w trakcie uczenia numer epoki (τ) ulega ciągłemu zwiększeniu, tak więc różnica $\tau - \tau_n$ jest zawsze dodatnia lub, jeśli węzeł powstał w aktualnej epoce, równa jest 0. Parametry uczenia κ, γ dopasowywane są w sposób automatyczny indywidualnie dla każdego węzła

$$\gamma = \frac{\Gamma}{1 + (\tau - \tau_n)/\Lambda} \;\; ; \;\; \kappa = \frac{\mathrm{K}}{1 + (\tau - \tau_n)/\Lambda}$$

Parametr Λ określa szybkość zmniejszania się parametrów uczenia (parametr ten ustalony jest na początku uczenia i właściwie we wszystkich dotychczasowych zastosowaniach miał zawsze taką samą wartość, 100), natomiast Γ i K są początkowymi wartościami parametrów uczenia, które trzeba ustalić przed rozpoczęciem uczenia.

Ponieważ zmiana rozmycia jest proporcjonalna do $(1-G)$, natomiast wartość maksymalnego wzbudzenia wynosi 1, to im większa wartość wzbudzenia tym mniejsze zwiększenie rozmycia. Stąd również ograniczenie co do funkcji aktywacji aby jej wartości były z przedziału [0,1]. Jeśli węzeł klasyfikuje poprawnie nowy wektor to jego masa zostaje zwiększona o jeden: $m = m+1$. Na początku nowej epoki masa wszystkich węzłów jest ustalana na wartość 1.

W celu uniknięcia zbyt silnego nakładania się węzłów reprezentujących różne klasy na siebie, w czasie optymalizacji podejmowana jest również próba zmniejszenia podobieństwa do węzła będącego największym zagrożeniem. Węzeł, który był optymalizowany (czyli węzeł najbardziej podobny do wektora wejściowego) zostaje wyłączony z obliczeń, i poszukiwany jest

węzeł najbardziej wzbudzający się. Jeżeli jest on z innej klasy niż wektor wejściowy należy zmniejszyć jego podobieństwo poprzez zmniejszenie przypisanego mu rozmycia:

$$\sigma_i = \sigma_i - \upsilon\, G\left(X; R, \sigma\right) \frac{|X_i - R_i|}{1 + \left(\tau - \tau_n\right)/\Lambda}$$

Wartość parametru υ dla dużych wbudzeń (G>0.6) wynosi 1.5 dla małych 1, w przypadku gdy rozmycie przyjmuje ujemną wartość, brana jest wartość bezwzględna.

Czasami może się zdarzyć, że rozmycie ulega gwałtownemu zmniejszeniu, dlatego też w celu ułatwienia ponownego późniejszego dopasowania parametrów odmładzamy dany neuron, czyli zwiększamy jego czas powstania.

$$\tau_n = \tau_n + \left(\tau - \tau_n\right)/2$$

Jeżeli jest on z tej samej klasy to nie trzeba dokonywać dalszych zmian. Jest to równoznaczne ze zgodą na to, by węzły reprezentujące tę samą klasę nakładały się na siebie. Można więc stwierdzić, że podczas tej fazy uczenia następuje samoorganizacja [6],[7] z nadzorem czyli z uwzględnieniem informacji o klasach.

5.2. Faza II

Nowy węzeł wstawiany jest do sieci wtedy, gdy dla istniejącego wektora wejściowego zarówno najbliższy w sensie euklidesowym jak i najbardziej wzbudzający się węzeł są z innej klasy niż wektor na wejściu sieci. Współrzędne położenia nowego węzła są takie same jak wektora wejściowego. Natomiast jego rozmycie ustawiane jest na podstawie położenia najbliższego neuronu. Tak więc podstawowe parametry nowo powstałego węzła mają postać:

$$R_i = X_i \quad m = 1 \quad \tau_n = \tau$$

Liczba tworzonych węzłów w jednej epoce jest ograniczona i równa jest liczbie klas znajdujących się w zbiorze terningowym. Takie ograniczenie umożliwia maksymalny rozrost węzłom istniejącym. Jak pokazuje doświadczenie, bez ograniczenia powstaje większa liczba węzłów, a co za tym idzie jakość generalizacji jest mniejsza, oraz wydłuża się czas uczenia.

Po każdej epoce sprawdzana jest jakość klasyfikacji *Q* każdego z istniejących węzłów, określona przez stosunek liczby poprawnie sklasyfikowanych wektorów do liczby wszystkich wektorów, które „wpadają" w dany węzeł. Sprawdzenie klasyfikacji polega na wyszukaniu wśród istniejących węzłów takiego, którego wzbudzenie jest największe, a jednocześnie jest większe od zadanej wartości minimalnej (progu). Jeżeli wzbudzenia wszystkich węzłów są poniżej zadanego progu to dany wektor wejściowy nie jest klasyfikowany. Parametr wartości minimalnej jest na początku uczenia duży, przez co tylko wektory blisko centrum neuronu brane są pod uwagę. W trakcie trwania uczenia parametr ten jest automatycznie zmniejszany.

Jeżeli dany neuron klasyfikuje poniżej wymaganej dokładności wówczas następuje zmniejszenie jego rozmycia proporcjonalnie do procentowej wartości błędu.

$$\sigma = \sigma - \lambda(1 - Q)\sigma$$

Wartość parametru λ jest mała (0.3) dla nowych węzłów, gdyż te dopiero adaptują się, natomiast w przypadku starszych węzłów parametr ten ma dużą wartość (0.9).

Z uwagi na to, że powstawanie nowych węzłów, jak również optymalizacja węzłów istniejących, zależne są od kolejności podawania wektorów treningowych na wejście sieci, za każdym razem, gdy sieć uczona jest na tych samych danych otrzymuje się konfigurację sieci, które różnią się od siebie. W większości przypadków te różnice są niewielkie, co łatwo zauważyć na zbiorze testowym, gdyż wynik w poszczególnych próbach na zbiorze testowym jest bardzo podobny. Gdy zbiór treningowy jest jednak niewielki wariancja modelu może być duża, co przejawia się dużą różnicą w klasyfikacji na zbiorze testowym dla poszczególnych prób.

Prostym sposobem na rozwiązanie tego problemu jest zastosowanie tzw. komitetu sieci. Metoda ta polega na tym, że na danym zbiorze treningowym uczy się sieć wielokrotnie. Mając *M* modeli sieci klasyfikacji wektorów ze zbioru testowego dokonuje się przez głosowanie większościowe, wybierając klasę, która najczęściej występuje, lub przez ważoną kombinację rozwiązań z poszczególnych sieci [9]. Metoda ta powoduje znaczne zmniejszenie wariancji modelu i, jak pokazuje doświadczenie, doskonale nadaje się do sieci FSM.

5.3. Obroty

W celu lepszego dopasowania węzłów sieci do danych w FSM stosuje się neurony, których funkcje transferu w *n* wymiarowej przestrzeni dają możliwość obrotu realizowanych przez nie hiperpowierzchni stałych wartości, czyli granic decyzji, wyznaczonych przez neuron. Wykonanie pełnego obrotu jest obliczeniowo bardzo kosztowne i powoduje gwałtowne zmniejszenie się szybkości uczenia. Dlatego też zastosowano obroty będące złożeniem obrotów dwuwymiarowych. Dla przykładu przedstawiony zostanie obrót w przestrzeni trójwymiarowej. Najpierw dokonujemy obrotu układu współrzędnych dla pierwszych dwóch wymiarów o kąt α, w wyniku czego z wektora (X,Y) otrzymujemy wektor (X',Y'):

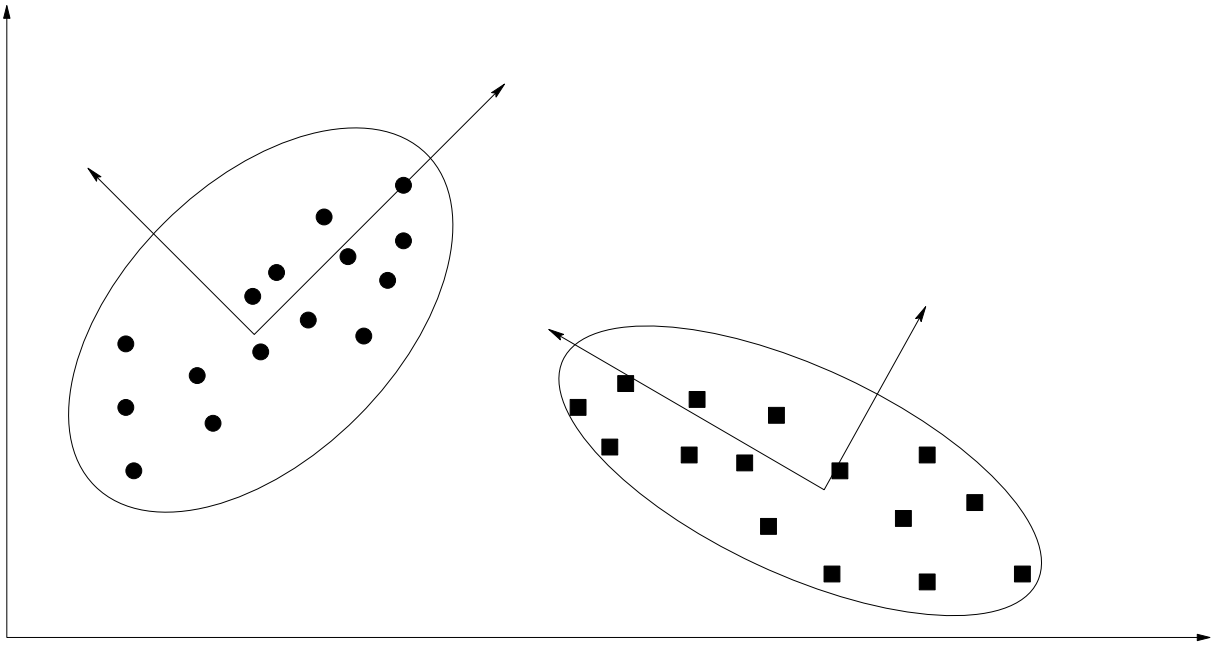
$$\begin{pmatrix} X' \\ Y' \end{pmatrix} = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix}$$

W następnym kroku

$$X = X' \; ; Y = Y'$$

i powtarzamy operacje tym razem uwzględniając kąt β położenia danego węzła między trzecim wymiarem i płaszczyzną (X,Y). W ogólnym przypadku cała procedura powtarzana jest dla wszystkich kolejnych wymiarów. Kąty obrotu dla poszczególnych węzłów ustalane są podczas uczenia dla każdego węzła niezależnie.

Zmiana kątów nachylenia neuronu do odpowiednich osi następuje za każdym razem, gdy wykonywana jest adaptacja położenia i innych parametrów tego węzła. Najpierw wyznaczane są poszczególne kąty nachylenia θ wektora mającego swój początek w środku węzła (a więc jest to położenie tego węzła) a koniec w miejscu, gdzie znajduje się aktualny punkt wejściowy (w



Przykład obróconych dwóch neuronów

wielowymiarowej przestrzeni wektor wejściowy może być traktowany jako punkt). Następnie aktualne kąty neuronu ϕ są przesuwane w kierunku wyznaczonych wartości.

$$\phi = \phi + \gamma(\theta - \phi)$$

parametr γ jest to ten sam parametr, który wystąpił podczas uczenia rozmyć

Jeżeli w danych występują wyraźnie nachylone skupiska to większość kątów wyznaczonych dla poszczególnych wektorów wpadających do tego węzła będzie do siebie podobnych, przez co łatwo jest uzyskać optymalny kąt. Jednak gdy nie ma wyróżnionego kierunku dla skupiska, to następuje ciągła zmiana wartości kąta ϕ co może powodować trudności w uczeniu się. Dla skupisk wyznaczonych podczas inicjalizacji sieci kąty znajdowane są za pomocą prostej $X_i = a_i X_1$ dopasowanej na podstawie wszystkich wektorów należących do tego węzła, stąd poszczególne parametry

$$a_i = \frac{\sum_k X_1^k X_i^k}{\sum_k X_i^k X_i^k}$$

Współczynniki te równe są tangensowi kąta pomiędzy główną osią skupiska a osią X_1. Tak więc poszczególne kąty można wyznaczyć ze wzoru

$$\phi_k = \arctan(a_k)$$

Nowy węzeł, który powstaje podczas uczenia ma początkowo kąty zerowe, ponieważ tylko jeden wektor należy do tego węzła i jest on jego środkiem. Inicjalizacja z uwzględnieniem niezależnych obrotów dla poszczególnych węzłów jest bardziej precyzyjna niż zastosowanie analizy czynników głównych (PCA) lub innych globalnie określonych transformacji danych. Nawet jeśli w czasie uczenia parametry obrotu nie podlegają adaptacji warto wprowadzić macierze obrotu w opisany powyżej sposób. Inne metody tworzenia obróconych konturów decyzji opisano w pracy [5].

5.4. Optymalizacja

W końcowym etapie uczenia tzn. gdy osiągnięta jakość klasyfikacji zgodna jest z pożądaną, następuje optymalizacja polegająca na odrzuceniu węzłów, które mają bardzo małe rozmiary tzn. do których wpada niewielka liczba (np. 1 lub 2) wektorów z ciągu treningowego. Powstają one na skutek żądania zbyt wysokiej dokładności klasyfikacji lub też niewystarczającego rozmycia dużych węzłów istniejących w sieci. Jeżeli przyczyną tworzenia małych węzłów jest zbyt szybkie ich tworzenie to po ich odrzuceniu próba douczenia istniejących węzłów spowoduje zwiększenie ich rozmyć. W przeciwnym przypadku ponownie zostaną utworzone małe węzły. Próba odrzucania i douczania powtarzana jest kilkakrotnie, po czym etap uczenia jest zakończony. Jakość klasyfikacji po tym etapie może być mniejsza od żądanej ponieważ część węzłów została odrzucona.

Etap kolejny to końcowa optymalizacja sieci, po której nie następuje już douczanie. Odrzucane są węzły, których likwidacja nie powoduje zmiany poziomu aktualnej klasyfikacji. Aby uniknąć sprawdzania każdego węzła z osobna, tzn. odrzucania go i sprawdzania klasyfikacji na całym zbiorze treningowym (co byłoby bardzo kosztowne) dla każdego wektora treningowego obliczana jest aktywacja wszystkich węzłów i zapamiętywana wartość aktywacji węzła najbardziej wzbudzającego się. Ten węzeł zostaje wyłączony. Wyszukiwany jest kolejny węzeł najbardziej wzbudzający się i zapamiętywana jest wartość jego wzbudzenia. Mamy więc macierz z liczbą elementów dwa razy większą niż liczba wektorów w ciągu treningowym, natomiast wartości tych elementów równe są maksymalnym wzbudzeniom dla odpowiednich wektorów treningowych. W macierzy tej usuwamy jeden węzeł. Dla wektora, dla którego węzeł ten miał największą wartość aktywacji wstawiana jest do macierzy wartość zero. Następnie na podstawie pozostałych wartości wyliczana jest jakość klasyfikacji. Jeżeli jakość klasyfikacji nie pogorszyła się to analizowany węzeł jest usuwany, w przeciwnym przypadku wstawiane są do macierzy poprzednie wartości.

5.5. Selekcja cech

Dla każdego z węzłów ukrytych można również zastosować **eliminację cech**. W tym celu wybieramy kolejno węzły i dla każdego z nich odrzucamy z ciągu treningowego wektory, które do danego węzła wpadają, czyli które są przez niego klasyfikowane. Możemy to uczynić, gdyż wyłączenie cechy w danym węźle, z uwagi na to, że stosujemy funkcje separowalne, może być traktowane jak wstawienie do iloczynu wartości 1, a więc maksymalnej wartości aktywacji dla danego wymiaru. Może to spowodować tylko zwiększenie aktywacji węzła dla wszystkich wektorów treningowych. Nie może się więc zmienić klasyfikacja wektorów, które były poprawnie klasyfikowane, bo dla tych wektorów ten węzeł i tak miał największe wzbudzenie. Dlatego obliczamy macierz wzbudzeń tylko dla pozostałych wektorów treningowych. Każda z cech dla danego węzła zostaje kolejno odrzucana. Powoduje to wzrost funkcji aktywacji tego węzła. Jeśli aktywacja ta dla wektorów z innej klasy niż optymalizowany węzeł nie przekracza aktywacji węzłów poprawnie je klasyfikujących, zapisanych w tablicy aktywacji, to taka cecha nie jest potrzebna.

Procedury odrzucania węzłów oraz eliminacji cech powtarzane są tak długo aż żaden węzeł nie może już zostać odrzucony. Alternatywna metoda selekcji cech, polegająca na modyfikacji funkcji błędu, opisana została w pracy [13].

5.6. Rozpoznawanie

Testowanie sieci FSM polega na wyszukiwaniu dla danego wektora wejściowego węzła, który ulega największemu wzbudzeniu. Wektor wejściowy jest przypisany do klasy, którą reprezentuje najlepszy neuron. Jeżeli jednak żaden z węzłów nie ulega wzbudzeniu wówczas wektor wejściowy nie zostaje sklasyfikowany.

Do oszacowania jakości klasyfikacji służą dwa parametry: wartość wzbudzenia oraz współczynnik zaufania. Wartość wzbudzanie *G* jest z zakresu [0,1]; im jest większa tym wektor wejściowy bliższy jest istniejącemu prototypowi, który powstał podczas uczenia.

Współczynnik zaufania natomiast można traktować jako prawdopodobieństwo poprawnego zaklasyfikowania wektora wejściowego *X* do danej klasy

$$p_k = \frac{\sum_C G_{Ci}(X_k; D_{Ci}, \sigma_C)}{\sum_i \sum_C G_{C_i}(X_k; D_{C_i}, \sigma_{C_i})}; \quad \sum_k p_k = 1$$

W liczniku sumowanie przebiega po wszystkich węzłach reprezentujących klasę zwycięzcy, w mianowniku suma biegnie po wszystkich węzłach ukrytych w sieci. Węzły przechowywać mogą również informacje symboliczne, które mogą być przedstawiane na wyjściu.

6. Przykładowe rezultaty

Do testowania sieci FSM użyto standardowych zagadnień klasyfikacyjnych, wziętych z repozytorium baz danych Uniwersytetu Kalifornijskiego w Irvine [8], gdzie znaleźć można dokładniejszy opis danych. Wszystkie poniższe przykłady przedstawiają wyniki, które powstały w wyniku uśrednienia kilku rezultatów dla danej bazy.
Wszystkie dane zostały najpierw zestandaryzowane. W poniższych tabelach podana została procentowa poprawność klasyfikacji na zbiorze testowym, lub też jeśli zbiór testowy nie występował przedstawiony jest wynik z 10 krotnej kroswalidacji. W przypadku gdy istniały zebrane wyniki różnych klasyfikatorów dla danego zbioru, FSM został porównany z najlepszymi z nich. Jeśli takie wyniki nie istniały dokonaliśmy porównania FSM z najlepszym modelem MLP, jaki udało się znaleźć w literaturze (była to zwykła wsteczna propagazcja, Rprop, Quickprop lub – dla danych „hypothyroid" – genetyczna optymalizacja parametrów), metody najbliższych sąsiadów *k*-NN [9] oraz drzewa decyzji C4.5 [10]. We wszystkich tabelach rezultat otrzymany z sieci FSM przedstawiony jest w ostatniej kolumnie wszystkie pozostałe rezultaty są w kolejności malejącej dokładności klasyfikacji. We wszystkich przypadkach wyniki osiągnięte za pomocą FSM są zbliżone do najlepszych. Więcej wyników znaleźć można na stronie internetowej: http://www.phys.uni.torun.pl/kmk/projects/datasets.html

- Dane DNA zawierają 2000 wektorów w zbiorze treningowym i 1186 wektorów w zbiorze testowym. Występuje 180 binarnych atrybutów i trzy klasy, o następującym rozkładzie procentowym: w zbiorze treningowym 23.2%, 24.3%, 52.5%, w zbiorze testowym 25.6%, 23.6%, 50.8%.

	Radial	Dipol92	Alloc80	QuaDisc	Discrim	**FSM**
Dokładność na zbiorze testowym	95.9%	95.2%	94.3%	94.1%	94.1%	**94.3%**

- Dane „jonosfera" zawierają 351 wektorów, z czego pierwszych 200 używa się do trenowania, natomiast pozostałe do testowania. Dane te reprezentują sygnał radarowy odbity od jonosfery. Każdy z wektorów zawiera 34 atrybuty o wartościach ciągłych. Występują dwie klasy, pierwsza świadcząca o występowaniu w jonosferze pewnych struktur i druga, świadcząca o braku tych struktur. Obie klasy w zbiorze treningowym są prawie równoliczne.

	k-NN	MLP	C4.5	**FSM**
Dokładność na zbiorze testowym	98.7%	96.0%	94.9%	**97.7%**

- Dane „hypothyroid" zawierają 3772 wektory w zbiorze treningowym i 3428 w zbiorze testowym. Są to dane z dwóch kolejnych lat badań przesiewowych, zawierające informacje o niedoczynności, nadczynności lub normalnie działającej tarczycy. Procentowy rozkład tych trzech klas jest następujący: w pliku treningowym 2.5%, 5.1%, 92.4%, testowym 2.1%, 5.2%, 92.7%. Występuje 21 atrybutów, 15 binarnych i 6 ciągłych.

	C4.5	C-MLP2LN	CART	**FSM**
Dokładność na zbiorze testowym	99.5%	99.36%	99.36%	**99.1%**

- Dane „satimage" składają się z intensywności pikseli znajdujących się w sąsiedztwie 3x3, otrzymanych z czterech zdjęć satelitarnych w różnych zakresach spektralnych, stąd występuje 36 atrybutów mających wartości z przedziału 0-255. W bazie jest 4435 wektorów treningowych i 200 testowych. Dane podzielone są na sześć klas (chodzi o różne powierzchnie odbijające) mających następującą procentową liczebność w poszczególnych zbiorach: zbiór treningowy 24.2%, 10.8%, 21.6%, 9.4%, 10,6%, 23.4%, testowym 23.1%, 11.2%, 19.9%, 10.5%, 11.8%, 23.5%.

	k-NN	LVQ	Dipol92	Radial	**FSM**
Dokładność na zbiorze testowym	90.6%	89.5%	88.9%	87.9%	**89.8%**

- Dane „sonar" zawierają 208 wektorów, które podzielone równo na zbiór treningowy i testowy. Występują dwie klasy. Każdy z wektorów posiada 60 ciągłych atrybutów. Celem jest odróżnienie sygnału sonaru odbitego od metalu od sygnału odbitego od skały.

	MLP+BP	k-NN	C4.5	**FSM**
Dokładność na zbiorze testowym	90.4%	84.2	65.4	**88.8%**

- Dane „shuttle" zawierają 43500 wektorów w zbiorze treningowym i 14500 w zbiorze testowym. Występuje siedem klas mających następujący udział procentowy w poszczegól-

nych zbiorach: zbiór treningowy 78.41%, 0.09%, 0.3%, 15.51%, 5.65%, 0.01%, 0.03%, testowym 79.16%, 0.09%, 0.27%, 14.86%, 5.58%, 0.03%, 0.01%. Dane posiadają 9 ciągłych atrybutów.

	NewId	BayTree	Cn2	Cal5	**FSM**
Dokładność na zbiorze testowym	99.99%	99.98%	99.97%	99.97%	**99.97%**

- Dane „letters" zawierają wektory opisujące 26 dużych liter z alfabetu języka angielskiego. Występuje 20000 wektorów, z czego 16000 przeznaczonych zostało do trenowania sieci a pozostałe 4000 do testowania. Dane te utworzone zostały z 20 różnych fontów. Do każdej litery spośród tych 20 fontów dodany został szum. Utworzono czarno białe prostokątne obrazy, które następnie przedstawiono w postaci wektora składającego się z 16 cech numerycznych. Wszystkie cechy zostały przeskalowane tak aby wartości tych cechy były całkowite w zakresie od 0 do 15.

	Alloc80	k-NN	LVQ	QuaDisc	**FSM**
Dokładność na zbiorze testowym	93.6%	93.2%	92.1%	88.7%	**92.2%**

- Dane „galaxies", zawierają opis galaktyk z katalogu ESO-LV [12]. Celem jest wykonanie automatycznej klasyfikacji, która porównywalna jest do klasyfikacji eksperta. Zbiór ten zawiera ponad 5000 wektorów. Sieć trenowana jest na zbiorze zawierającym 1700 elementów i testowana na zbiorze 3517 wektorów. Każdy wektor składa się z 13 cech, i może należeć do jednej z dwóch klas: galaktyki wczesnego typu, lub też galaktyki późnego typu.

	k-NN	C4.5	MLP+BP	**FSM**
Dokładność na zbiorze testowym	92.82%	92.4%	89.6%	**93%**

- Dane „hepatitis" dotyczą chorób wątroby otrzymane z Tokijskiego uniwersytetu medycznego [13]. Dane te zwierają 536 przypadków, w tym 163 użyte zostały do testowania a pozostałe do trenowania sieci. Każdy przypadek opisany został przez 9 cech. Występują 4 klasy.

	k-NN	C4.5	MLP	**FSM**
Dokładność na zbiorze testowym	82.8%	75.5%	68%	**82.2%**

- Dane „segmentation" zawierają 210 wektorów w zbiorze treningowym i 2100 w zbiorze testowym. Występuje 7 klas, każda z klas w zbiorze treningowym posiada 30 wektorów a w zbiorze testowym 300. Wszystkie wektory opisane są przez 19 cech o wartościach ciągłych.

	k-NN	C4.5	MLP	**FSM**
Dokładność na zbiorze testowym	90.33%	89.8%	88.33%	**91.2%**

- Dane „breast cancer wisconsin" zawierają 699 przypadków, wśród których występuje 458 chorych oraz 245 zdrowych przypadków. Każdy przypadek opisywany jest przez 9 cech mających wartość w zakresie od 1-10. Dla 16 przypadków jedna wartość jest brakująca. W miejsca wartości brakującej wstawiona została wartość średnia dla danej cechy.

	IncNet	k-NN	FDA	**FSM**
Dokładność obliczona z 10-krotnej kroswalidacji	97.1%	97.1%	96.8%	**96.5%**

- Dane „appendictis" [14] zawierają 106 przypadków, opisanych za pomocą 8 cech. Występują dwie klasy mające następujący udział procentowy: 80.2%, 19.8%

	k-NN	MLP	C4.5	**FSM**
Dokładność obliczona z 10-krotnej kroswalidacji	89.3%	83.9%	83.5%	**84.2%**

7. Podsumowanie

Chociaż opisany tu algorytm wydawać się może początkowo skomplikowany i trudny w użyciu w rzeczywistości wykonuje się niemal automatycznie, nie wymagając ingerencji użytkownika w proces uczenia się lub wybierania architektury.
Sieć startuje z prototypów utworzonych podczas grupowania, a następnie dodaje, jeśli jest to konieczne, nowe neurony. Dodanie nowych neuronów uzależnione jest od odległości między wektorem znajdującym się na wejściu sieci i aktualnymi prototypami utworzonymi przez sieć oraz od reakcji sieci na ten wektor (wzbudzenia istniejących węzłów). Początkowe prototypy znajdowane są za pomocą wstępnej inicjalizacji opartej na grupowaniu. Inicjalizacja taka pozwala wystartować z optymalnych prototypów co powoduje często przyspieszenie procesu uczenia oraz utworzenie sieci o niewielkiej liczbie węzłów. Tak więc cała architektura sieci zależy tylko i wyłącznie od danych, które zostają użyte do jej uczenia.
Poza architektura, istotnym czynnikiem wpływającym na jakość klasyfikacji jest zdolność wytwarzania przez model różnorodnych obszarów decyzyjnych przy utrzymanej sensownej złożoności tego modelu. Własność ta umożliwia lepsze dopasowanie się klasyfikatora do danych, a co za tym idzie osiągnięcie lepszej generalizacji.
W sieci FSM zmianę kształtu obszarów decyzyjnych można uzyskać poprzez stosowanie różnych funkcji aktywacji oraz możliwość wykonania obrotów w wielowymiarowej przestrzeni. Stosowanie różnych funkcji aktywacji otwiera również różne drogi analizy danych np. interpretację za pomocą reguł logicznych (wystarczy w tym celu zastosować funkcje prostokątne, przechodząc stopniowo od funkcji bicentralnych lub gaussowskich), czy też interpretację opartą o logikę rozmytą (funkcja trójkątna, gaussowska). Ta giętkość tworzenia różnych obszarów decyzyjnych umożliwia zastosowanie sieci FSM w wielu dziedzinach co potwierdzają przedstawione wyniki.

Zastosowania tego modelu do aproksymacji, jako pamięci asocjacyjnej oraz jako funkcji heurystycznej w problemach optymalnego spełniania wielu ograniczeń [1], nie zostały jeszcze podjęte.

Podziękowania: Za wsparcie finansowe w ramach grantu nr 8 T11F 014 14 jesteśmy wdzięczni Komitetowi Badań Naukowych.

Literatura

[1] Duch W., Diercksen G. H. F. (1995): *Feature Space Mapping as a universal adaptive system.* -- Computer Physics Communications, Vol 87, pp. 341-371.

[2] Duch W. (1997): *Platonic model of mind as an approximation to neurodynamics.* [w:] Brain-like computing and intelligent information systems (S-i. Amari, N. Kasabov, Ed.). – Singapore Springer, rozdz. 20, s. 491-512.

[3] Bishop C. (1995): *Neural network for pattern recognition.* – Oxford: Clarendon Press.

[4] Duch W. (1996): *From cognitive models to neurofuzzy systems - the mind space approach.* -- Systems Analysis-Modeling-Simulation, Vol. 24, pp. 53-65.

[5] Duch W., Jankowski N. (1999): *Survey of neural transfer functions* -- Neural Computing Surveys, Vol. 2, pp. 163-213.

[6] Duch W., Adamczak R., Jankowski N. (1997): *Initialization of adaptive parameters in density networks* -- Third Conference on Neural Networks and Their Applications, Kule, październik 1997, s. 99-104 .

[7] Kohonen T. (1995): *Self-organizing maps.* -- Heidelberg Berlin: Springer-Verlag.

[8] Blake C. L. and Merz C. J. (1999), *UCI repository of machine learning databases*. -- http://www.ics.uci.edu/~mlearn/MLRepository.html

[9] Duch W., Grudziński K., *The weighted k-NN method with selection of features and its neural realization.* -- 4th Conference on Neural Networks and Their Applications, Zakopane, Maj 1999, s. 191-196.

[10] Quinlan J. R. (1993): *C4.5: Programs for machine learning.*-- Morgan Kaufmann, San Mateo, CA.

[11] Duch W., Adamczak R., Grąbczewski K. (2000): *Methodology of extraction, optimization and application of crisp and fuzzy logical rules.* -- IEEE Transactions on Neural Networks special issue on Data Mining.

[12] Lahav O., Naim A., Sodre L. Jr. and Storne-Lombardi M. C. (1995): *Neural Comptutation as a tool for galaxy classification: methods and examples*. -- Institute od Astronomy, Cambridge, Technical report CB3 OHA.

[13] Duch W., Adamczak R., Grąbczewski K., Żal G., Hayashi Y. (1999): *Fuzzy and crisp logical rule extraction methods in application to medical data.* -- Computational Intelligence and Applications (P.S. Szczepaniak. Ed.) Springer, Studies in Fuzziness and Soft Computing, Vol. 23.

[14] Weiss S.M., Kapouleas I. (1990): *An empirical comparison of pattern recognition, neural nets and machine learning classification methods*. [w:] Readings in Machine Learning (Shavlik J.W. and Dietterich T.G. Ed.). -- Morgan Kauffman Publ, CA.

[15] Churchland P.S., Sejnowski T.J. (1992): *The computational brain*. -- MIT, Bradford Book.

[16] Thelen E., Smith L.B. (1994): *A Dynamic Systems Approach to the Development of Cognition and Action*. -- MIT Press.

[17] Breiman L. (1998): *Bias-variance, regularization, instability and stabilization*. -- Neural Networks and Machine Learning (Bishop C. M. Ed.), s. 27–56. Springer-Verlag.