Współczesne Metody Heurystyczne - Projekt

Piotr Doniec pdoniec@elka.pw.edu.pl
Łukasz Trzaska ltrzaska@elka.pw.edu.pl
Wydział Elektroniki i Technik Informacyjnych,
Politechnika Warszawska,

21 stycznia 2012

1 Wstęp

1.1 Treść zadania

Celem projektu jest implementacja matody symulowanego wyżarzania (ang. simulated annealing - SA) w kilku wariantach do doboru wartości parametrów jąder przekształcenia w zadaniu klasyfikacyjnym. Wyznacznikiem jakości klasyfikacji jest ilość popełnianych błędów. Otrzymane wyniki należy porównać z metodą pełnego przeglądu.

1.2 Symulowane wyżarzanie

Symulowane wyżarzanie to rodzaj algorytmu poszukującego rozwiązania problemu optymalizacyjnego. Jest to algorytm heurystyczny, zatem znalezione rozwiązanie nie jest optymalne, a jedynie do niego zbliżone. Innymi przykładami algorytmów heurytycznych stosowanymi do optymalizacji jest rodzina algorytmów ewolucyjnych, Multistart, Adaptive Random Search.
Symulowane wyżarzanie często wykorzystywane, jest gdy przestrzeń rozwiązań jest dyskretna. Najpowszechniejszym przykładem zadania, w który zastosowanie znajduje SA jest TSP (ang. Travelling Salesman Problem). Idea działania SA pochodzi z metalurgii i po raz pierwszy została opisana w 1953 roku. Pod wpływem wysokiej temperatury cząstki, atomy uwalniają się z pozycji, a oddając energię w czasie kontrolowanego schładzania rozkładają się w sposób bardziej systematyczny, przyjmując mniejszą wewnętrzną energię. Bazując na podobnych założeniach zaproponowano metodę symulowanego wyżarzania do rozwiązywania zadań optymalizacyjnych.
Algorytm znalazł szerokie zastosowanie począwszy od kolorowania i podziału grafu aż do optymalizacji rozlokowania układów elektronicznych, rekonstrukcji obrazu i geofizyki. Powszechność metody wynika głównie z łatwości implementacji w celu rozwiązania konkretnego zadania optymalizacyjnego, braku szczególnych wymagań dotyczących analizowanej / optymalizowanej funkcji a ponadto

dowiedziono, że SA asymptotycznie dąży do globalnego minimum [2]. Dodatową cechą algorytmu jest unikanie pułapek związanymi z minimami lokalnymi . Dzieje się tak, gdyż w określonych przypadkach, SA zezwala na przyjęcie gorszego rozwiązania niż aktualnie najlepsze.

1.2.1 Ogólny opis algorytmu

Algorytm symulowanego wyżarzania łatwo rozpatrywać dzieląc na 4 fazy: generowanie nowego, następnego stanu, sprawdzenie warunku akceptacji, redukcja parametrów sterujących, sprawdzenie warunku stopu.

Algorithm 1 Generyczna postać algorytmu symulowanego wyżarzania

```
while warunek stopu nie osiągnięty do
  for j = 1 → N_c^k do
    Generowanie nowego stanu
    Ewaluacja kryterium akceptacji
  end for
  Uaktualnienie N_c^k
  Redukcja parametru sterującego
end while
```

Generowanie nowego stanu. Jest to bardzo istotny etap. Sposób wytwarzania nowych punktów wpływa w oczywisty sposób na wydajność algorytmu. Po kilku iteracjach algorytmu, powinno się uzyskiwać coraz niższe wartości funkcji. Z tego powodu dopasowanie metody generowania punktów do problemu jest bardzo istotne z punktu widzenia działnia algorytmu. W przypadku TSP, proponowanym sposobem na wyznaczenie kolejnych stanów jest zamiana kolejności w losowo wybranej parze kolejnych miast na drodze komiwojażera.

Punkt startowy algorytmu jest często poprostu losowany. Jednakże, w wielu arytułach pojawiają się sugestie, aby dobrać początkowy stan po uprzedniej analizie problemu.

Kryterium akceptacji. Pozwala ono SA na uniknięcie pułapki związanej z minimum lokalnym, kiedy poszukiwane jest minimum globalne. Dodatkowo w każdej kolejnej iteracji, prawdopodobieństwo tego że nowe, ale gorsze rozwiązanie zostanie zaakceptowane jest coraz mniejsze. Z tego wynika, że pod koniec działania algorytm koncentruje się na poprawianiu precyzji uzyskanego wyniku. Najczęściej stosowanym kryterium akceptacji jest kryterium Metropolisa.

$$t^{k+1} = \begin{cases} y & \text{if } \tau \leq A_{t^k y}(c^k) \\ t^k & \text{wpp.} \end{cases}$$

$$A_{t^k y}(c^k) = min(1, \exp(-\frac{g(t^k) - g(y)}{c^k})$$

, gdzie:
t^k to aktualny stan, oszacowanie globalnego minimum,
y jest nowym punktem, kandydatem,
τ wartość wylosowana z przdziału jednostajnego $U(0,1)$,
$A_{t^k y}(c^k)$ kryterium Metropolisa

Większość vartiantów symulowanego wyżarzania korzysta z kryterium Metropolisa. Tyczy się to również obu modyfikacji opisanych w natępnych punktach.

Redukcja parametru sterującego. Funkcja malejąca c^k jest nazywana parametrem sterującym lub harmonogramem schładzania. W celu uzyskania wydajnego algorytmu początkowa wartość parametru sterującego powinna być wystarczająco wysoka by przeszukać regiony zawierające potencjalne rozwiązanie, ale jednocześnie nie przesadzono, bo spowoduje to spowolnienie algorytmu. Z tego powodu dobra wartość początkowa jest praktycznie niemożliwa do oszacowania bez żadnej analizy problemu.

Warunek stopu. Istnieje wiele różnych warunków stopu do zastasowania w przypadku symulowanego wyżarzania. Wszystkie sprowadzają się do prostego pomysłu, że algorytm powinien się zatrzrymać jeśli „materiał" zamarzł i mimo zmiany temperatury nie dostrzega się zmiany w rozwiązaniu. Przykładowym warunkiem stopu może być określona liczba iteracji lub osiągnięcie dolnej granicy parametru sterującego. Innym pomysłem jest zatrzymanie wyżarzania, kiedy kolejne N iteracji nie przynosi zmiany wyniku.

1.2.2 Wariant Boltzmana

W wariantcie Boltzmanna, nazywanym również SSA (ang. Standard Simulated Annealing), wykorzystywane są kolejno następująca funkcja generowania nowych punktów i harmonogram schładzania:

$$g(\Delta x) = (2\pi T)^{-\frac{D}{2}} \exp(-\frac{(\Delta t^2)}{2T})$$

$$T(k) = \frac{T_0}{lnk}$$

,gdzie
$\Delta t_k = t_{k+1} - t_k$, to różnica między aktualnym i poprzednim stanem
k, kwant czasu

1.2.3 Wariant FSA

Wariant wyróżnia się nie tylko inną metodą generowania nowego stanu, ale także inną funkcją zmiany temperatury. Na bazie FSA powstały kolejne modyfikacje

SA. Najpierw było to VFSA (ang. Very Fast Simulated Annealing) a potem ASA (ang. Adaptive Simulated Annealing). Ostatnia pozycja charakteryzuje się zmianą parametrów (tj. harmonogram chłodzenia, generacja kolejnego stanu) w trakcie działania algorytmu. To sprawia, że algorytm jest bardziej wydajny i mniej wrażliwy na zdefiniowane przez użytkownika parametry.

Równania dla FSA:

$$T(t) = T_0/t$$

$$g(\Delta x) = \frac{T}{(\Delta x^2 + T^2)^{\frac{D+1}{2}}}$$

1.3 SVM

SVM (ang. Supporting Vector Machine) to koncepcja w statystyce oraz informatyce dla metod nadzorowanego uczenia maszyn w analizie danych i rozpoznawania wzorców na użytek algorytmów klasyfikacji i analizy regresji. Standardowo maszyna SVM jest w stanie przydzielić dane do jednej z dwóch klas. Czyni to algorytm SVM nieprobabilistycznym liniowym klasyfikatorem binarnym. Zadając zbiór danych trenujących, przyporządkowanych do jednej z kategorii, algorytm trenujący SVM buduje model, który przypisuje dane przykładowe do jednej z tych kategorii. Model ten jest reprezentacją danych przykładowych jako punktów w przestrzeni, odwzorowanej w taki sposób, aby dane z wydzielonych kategorii były oddzielone wyraźną przestrzenią, tak dużą jak to możliwe. Nowe dane są następnie odwzorowane na tą samą przestrzeń i przypisane do kategorii biorąc pod uwagę część przestrzeni w której się znajdują.

1.3.1 Jądro przekształcenia

Przy przekstrzałceniu danych treningowych na punkty w przestrzeni, która później służy do przypisywania kategorii dla danych testowych, można użyć kilku rodzajów jąder. W praktyce, na podstawie doświadczeń ze środowiskiem rozwiązania zadania MatLab, używa się poniższych:

1. liniowe,
2. kwadratowe,
3. wielomianowe,
4. tangens hiperboliczny,
5. rbf - funkcja charakterystyczna normalnego rozkładu Gausowska,
6. mlp - funkcja mnożnika perceptronu.

Testy zostaną przeprowadzone dla jąder: wielomianowego, tangensa hiperbolicznego oraz rbf. W wynikach zwizualizowane zostaną te dane, które są najciekawsze pod kątem prezentacji różnorodności rozwiązania przedstawionego problemu.

1.3.2 Wieloklasowy SVM

Podział danych na dwie klasy jest często nie wystarczający. Aby rozszerzyć możliwości SVM o kategoryzację do kilku klas, wykorzystuje się kilka, niezależnie trenowanych binarnych SVM i definiuje strategię przydziału nowego wektora do jednej z istniejących klas. Jedną z takich strategii jest „one-versus-all". Metoda polega na utworzeniu, w przypadku M klas, M binarnych maszyn SVM w których podział następuje między jedną z dostępnych klas a pozostałymi (M - 1) klasami. Przydział testowego wektora do klasy odbywa się na zasadzie winner-takes-all. Inne możliwe metody budowy wieloklasowego SVM to „one-against-one" lub „DAGSVM".

2 Założenie projektowe

Cele postawione przed niniejszym projektem:

1. Porównanie wyników dla dwóch rodzajów jąder przekształcenia klasyfikacyjnego algorytmu SVM - zadane jako parametr wejściowy
2. Wykorzystanie metody symulowanego wyżarzania w dwóch wariantach: Bolztmanna oraz FSA - zadane jako parametr wejściowy
3. Implementacja rozwiązania oraz przeprowadzenie i wizualizacja wyników testowania w środowisku MATLAB
4. Na wejściu nadesłane dane trenujące i testowe, na wyjściu przydział wektorów do klas diagram popełnionych błędów
5. Kryterium stopu - określona liczba iteracji zadana jako parametr wejściowy

Testowanie. Działanie dobranych parametrów dla jąder przekształceń przetestowane zostanie na przygotowanych wcześniej danych. Dane wejściowe treningowe, jak również dane testowe program przyjmnie w postaci wektorów liczb rzeczywistych, gdzie ostatnią składową wektora, będzie wartość liczbowa, wyznaczająca kategorię do której on przynależy. Wektory, zapisane będą w pliku tekstowym w postaci liczb zmiennoprzecinkowych o zapisie wykładniczym oddzielonych białym znakiem.

Złożoność obliczeniowa i czas wykonania. Szacowanie złożoności obliczeniowej oraz czasu wykonania dla całego rozwiązania dekomponuje się na złożoność i czas wykonania odpowiednio dla algorytmu wyżarzania oraz algorytmu treningu klasyfikatora SVM, ponieważ w kroku, w którym ewaluowana jest wartość bieżącego znalezionego rozwiązania dla wyżarzania, trzeba będzie wykonać trening i testowanie danych przy użyciu SVM. Będziemy posługiwać się pesymistyczną złożonością obliczeniową dla każdego z przypadków. Dla algorytmu symulowanego wyżarzania jest ona wielomianowa dla każdej składowej problemu, co w przypadku sumy złożoności przy doborze kilku parametrów pozostawia ją wielomianową. W przypadku algorytmu treningowego, parametrem złożoności klasyfikacji, będzie ilość danych wektorów wejściowych. Z dotychczasowej analizy dostępnych źródeł ustaliliśmy, że złożoność ta jest liniowa z ewentualnym doprecyzowaniem ilością wymiarów wektorów. W konkluzji, przyjąć można, że złożoność rożwiązania będzie wilomianowa.

Prezentacja wyników. Wyniki porównania jakości doboru parametrów jąder przekształceń zaprezentowane zostaną na wykresach obrazujących liczbę błędów klasyfikacji w zależności od wybranego jądra z podziałem na rodzaj danych (zredukowane i pełne) oraz podziałem na rodzaj symulowanego wyżarzania FSA (ang. Fast Simulated Annealing) oraz Boltzmann. Czasy wykonania dla

poszczególnych typów jąder przekształceń nie będą zmienną, gdyż kryterium stopu obrane dla algorytmu jest kryterium czasu tzn. czas jest taki sam dla wszystkich porównywanych przypadków.

3 Środowisko realizacji

Ze względu na bardzo dużą ilość matematyki w projekcie zdecydowaliśmy się na realizację projektu w programie MATLAB. Na korzyść wybory wpływa dostępność prostej maszyny SVM zaimplementowanej w Bioinformatics Toolbox oraz funkcji realizującej symulowane wyżarzanie o dużych możliwościach konfiguracyjnych dostępnej w Global Optimization Toolbox. Dzięki temu można skoncentrować się na rozwiązaniu problemu bez konieczności implementowania wszystkich algorytmów składowych.

3.1 Symulowane wyżarzanie

Dostępna w MATLAB funkcja `simulannealbnd` realizująca realizująca symulowane wyżarzanie, jest tak naprawdę jedynie szablonem, zapisana jest w sposób generyczny. Zanim zostanie wykorzystana, należy ją dostosować do własnych potrzeb i do rozwiązywanego problemu. Do ustawiania parametrów wyżarzania korzysta się z `saoptimset`. Najważniejsze parametry z punktu widzenia projektu przedstawia Tablica 1 .

Tablica 1: Parametry funkcji `simulannealbnd`

Opcja	Opis
`AcceptanceFcn`	Uchwyt do funkcji której algorytm użyje jako kryterium akceptacji
`AnnealingFcn`	Uchwyt do funkcji która zostanie użyta do generowania nowego punktu, kandydata
`DataType`	Typ danych
`InitialTemperature`	Początkowa temperatura
`MaxFunEvals`	Maximum number of objective function evaluations allowed
`MaxIter`	Maksymalna liczba iteracji
`PlotFcns`	Wyświetlanie wykresu w czasie wyżarzania
`TemperatureFcn`	Uchwyt do funkcji realizującej harmonogram schładzania
`TimeLimit`	Limit czasu działania algorytmu w sekundach

3.2 SVM

W celu zbudowania klasyfikatora wykorzystującego SVM należy wykorzystać 2 funkcje. Pierwsza z nich `svmtrain` służy trenowaniu klasyfikatora. Jako

parametry przyjmuje wektor danych i wektor klas. Możliwe jest tez podanie dodatkowych parametrów kontrolnych. Jednym z nich jest `kernelfunction` umożliwiający wskazanie funkcji wykonującej przkształcenie. Oprócz predefiniowanych wartości takich jak `linear`, `quadratic`, `polynomial`, można równiez wskazać uchwyt do samodzielnie zdefiniowanej funkcji. Do klasyfikacji danych korzysta się z funkcji `svmclassify`. Parametry wejściowe to klasyfikator - rezultat `svmtrain` oraz dane które chcemy przydzielić do jednej z dwóch klas. Opcjonalnie można wygenerować wykres ilustrujący rezultat klasyfikacji.

4 Testowanie i wnioski

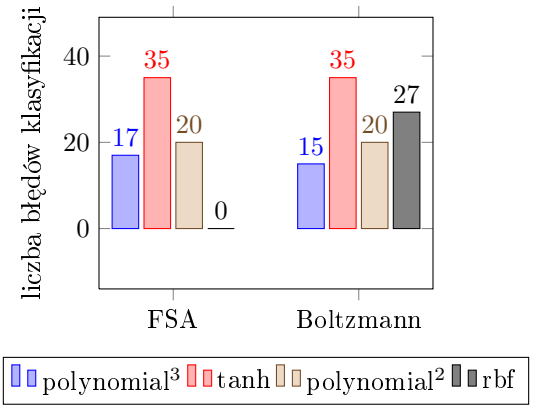
4.1 Kryterium stopu

Dla wykonanych testów, jako podstawowe kryterium stopu obrano czas, aby móc wykonać większą liczbę testów. Arbitralnie, na jeden test obrano limit czasu 300 sekund.

4.2 Błędy a typ jądra

Testy przeprowadzono dla dwóch zbiorów danych: *zawężonych* i *pełnych*.

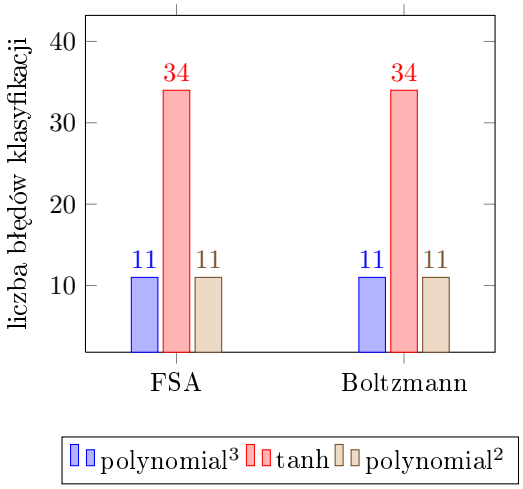
Dane zawężone Dla zróżnicowania badania skuteczności zastosowanej metody, test przeprowadzono najpierw na *danych zawężonych* tzn. klasyfikowano wektory o zredukowanej do 3 liczbie wymiarów. Dane znajdują się w pliku: *rs train7 short.csv*, a testujące w pliku *rs test7 short.csv*. Dane te przedstawiono na poniższym wykresie:



Dla tego zestawu danych w wyniku uzyskano następujące wartości parametrów jąder:

Typ SA	**Polynomial^3**	**Tanh**	**Polynomial^2**	**rbf**
FSA	4.7243, -8.6634	0, 0	-7.6178, 7.9733	-
Boltzmann	8.4878, 9.9313	0, 0	-8.4375, 7.9898	4.4031

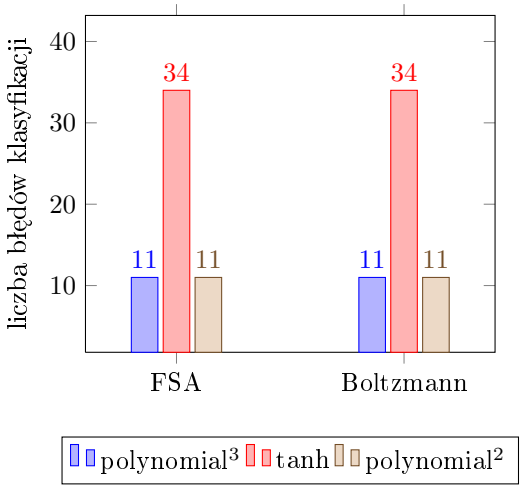
Dane pełne. Wyniki po trenowaniu na zbiorze *rs train5 no headers* i testowaniu na zbiorze *rs test5 no headers*.



Wyniki po trenowaniu na zbiorze *rs train7 no headers* i testowaniu na zbiorze *rs test7 no headers*.

Typ SA	**Polynomial^3**	**Tanh**	**Polynomial^2**
FSA	6.7973, 5.0605	-3.4106, -3.4106	8.2822, -8.2822
Boltzmann	2.0868, -9.7798	5.5106, 8.3447	8.9050, -4.5498

Wyniki po trenowaniu na zbiorze *rs train7 no headers* i testowaniu na zbiorze *rs test7 no headers*.



Dla tego zestawu danych w wyniku uzyskano następujące wartości parametrów jąder:

Typ SA	**Polynomial^3**	**Tanh**	**Polynomial^2**
FSA	1.2699 1.2699	0, 0	-3.9096 -9.7767
Boltzmann	3.4726 -9.3574	0, 0	-1.0261 -4.8345

Literatura

[1] Nello Cristianini, John Shawe-Taylor, *An introduction to Support Vector Machines and other kernel-based learning methods.* Cambridge University Press, 2000

[2] Ana I.P.N Pereira, Edite M.G.P. Fernandes, *A study of simulated annealing variants.* Braganca Polytechnic Institute, Bragaca, Portugal, Minho University, Braga, Portugal.

[3] Naotoshi Seo, *A Comparison of Multi-class Support Vector Machine Methods for Face Recognition.* Department of Electrical and Computer Engineering, The University of Maryland, 2007.

[4] Kou-Yuan Huang,Yueh-Hsun Hsieh, *Very Fast Simulated Annealing for pattern detection and seismic applications.* Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan.

[5] Shih-Wei Lin, Zne-Jung Lee, Shih-Chieh Chen, Tsung-Yuan Tseng, *Parameter determination of support vector machine and feature selection using simulated annealing approach.* Department of Information Management, Chang Gung University, Department of Information Management, Huafan University, Department of Industrial Management, National Taiwan University of Science and Technology, 2007.

[6] *http://www.iue.tuwien.ac.at/phd/binder/node87.html*