Sztuczne sieci neuronowe

Krzysztof A. Cyran
POLITECHNIKA ŚLĄSKA
Instytut Informatyki, p. 311

Wykład 8

PLAN:

- Zastosowanie sieci neuronowych do klasyfikacji statystycznej:

  statystyczny model klasyfikacji

  aproksymacja a'posteriori w sieciach MLP

  probabilistyczne sieci neuronowe

Klasyfikacja statyczna

- Wyniki klasyfikacji w krokach poprzednich nie wpływają na klasyfikację w chwili bieżącej
- Przykładem jest klasyfikacja dokonywana przez WYTRENOWANĄ sieć neuronową

Klasyfikacja statystyczna - motywacja

- Klasyfikowane wektory cech są obarczone zakłóceniami powstałymi w różnego rodzaju przetwarzaniu wstępnym
- Najczęściej zakłócenia są złożeniami niezależnych zakłóceń elementarnych, zatem wynikowe zakłócenia mają normalny rozkład gęstości
- Rezultatem jest częściowe nakładanie się w przestrzeni cech regionów należących do poszczególnych klas: klasyfikacja bezbłędna jest niemożliwym do spełnienia ideałem

Klasyfikacja statystyczna – losowa klasa abstrakcji

- Wyeliminowanie determinizmu klasyfikacji daje podstawy do wprowadzenia pojęcia losowej klasy abstrakcji
- Losowa klasa abstrakcji wyznaczana jest poprzez funkcję $p\,(\mathbf{x} \mid C_k)$ określającą gęstość prawdopodobieństwa przynależności wektora **x** do klasy C_k

Klasyfikacja statystyczna - cele

- Ponieważ dla wielu k może zachodzić jednocześnie: $p\ (\mathbf{x} \mid C_k) > 0$, zatem wiele wektorów cech nie może być zakwalifikowanych jednoznacznie do jednej tylko klasy
- Dlatego celem klasyfikacji statystycznej jest klasyfikacja, której odpowiada najmniejsze prawdopodobieństwo nieprawidłowej decyzji (uwzględniając również koszty błędnych decyzji w postaci dodatkowych wag).

Klasyfikacja statystyczna - oznaczenia

- $p\,(\mathbf{x} \mid C_k)$ – rozkład określający losowe klasy abstrakcji
- $P\,(C_k)$ – prawdopodobieństwo a'priori klasy C_k
- $p\,(\mathbf{x})$ – rozkład prawdopodobieństwa wektora cech **x**
- $P\,(C_k \mid \mathbf{x})$ – prawdopodobieństwo a'posteriori przynależności do klasy C_k po zaobserwowaniu wektora cech **x**

Reguła Bayes'a

- Prawdopodobieństwo koniunkcji zdarzeń zależnych (pierwsza postać):

$$P(C_k, \mathbf{x}) = P(C_k \mid \mathbf{x}) p(\mathbf{x})$$

- Prawdopodobieństwo koniunkcji zdarzeń zależnych (druga postać):

$$P(C_k, \mathbf{x}) = p(\mathbf{x} \mid C_k) P(C_k)$$

- Wniosek (reguła Bayes'a):

$$P(C_k \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_k) P(C_k)}{p(\mathbf{x})}$$

Mianownik we wzorze Bayes'a jest niezależny od klasy

- Ponieważ prawdopodobieństwo zdarzeń niezależnych wyraża się wzorem:

$$p(\mathbf{x}) = \sum_k P(C_k, \mathbf{x}) = \sum_k p(\mathbf{x} \mid C_k) P(C_k)$$

- Zatem mianownik wzoru Bayes'a pełni tylko rolę normalizującą, by prawdopodobieństwa a'posteriori sumowały się do jedności:

$$\sum_k P(C_k \mid \mathbf{x}) = 1$$

Problem określenia reguły klasyfikacji w modelu z niepewnością losową

- Określenie reguły decyzyjnej jest w ogólności zależne od celu klasyfikacji
- Najczęściej przyjmuje się minimalizację strat średnich wynikłych z błędnego zakwalifikowania
- Czasami jednak minimalizować można straty maksymalne, minimalne lub najbardziej prawdopodobne

Cel: minimalizacja strat średnich

Średnia strata wynikła z sytuacji, iż wektor cech **x** został zakwalifikowany do klasy C_j , oznaczana jako r_j (**x**), jest dana wyrażeniem:

$$r_j(\mathbf{x}) = \sum_{k=1}^{K} L_{kj} P(C_k | \mathbf{x})$$

gdzie: L_{kj} oznacza stratę powstałą z zakwalifikowania do klasy C_j obrazu faktycznie należącego do klasy abstrakcji C_k.

Cel: minimalizacja strat średnich (cd.)

Stosując wzór Bayes'a otrzymujemy:

$$r_j(\mathbf{x}) = \frac{1}{p(\mathbf{x})} \sum_{k=1}^{K} L_{kj}\, p(\mathbf{x}|C_k) P(C_k)$$

Założenie: straty ze wszystkich błędnych decyzji są sobie równe

- Wówczas funkcję strat L_{kj} można przedstawić jako:

$$L_{kj} = 1 - \delta_{kj}$$

- Gdzie:

$$\delta_{kj} = \begin{cases} 0 \Leftarrow k \neq j \\ 1 \Leftarrow k = j \end{cases}$$

Straty średnie (cd.)

- Podstawiając wzór na L_{kj} do wzoru na $r_j(\mathbf{x})$ dostajemy:

$$r_j(\mathbf{x}) = \frac{1}{p(\mathbf{x})}\left(\sum_{k=1}^{K} p(\mathbf{x}|C_k)P(C_k) - \sum_{k=1}^{K} \delta_{kj}\, p(\mathbf{x}|C_k)P(C_k)\right)$$

- Uwzględnienie wzoru na prawdopodobieństwo całkowite oraz faktu, że δ_{kj} jest różne od zera tylko dla $k = j$, prowadzi do:

$$r_j(\mathbf{x}) = 1 - \frac{1}{p(\mathbf{x})} p(\mathbf{x}|C_j)P(C_j)$$

Wzór końcowy na straty średnie

- Stosując jeszcze raz prawo Bayes'a otrzymujemy ostatecznie:

$$r_j(\mathbf{x}) = 1 - P\left(C_j \middle| \mathbf{x}\right)$$

Klasyfikacja w modelu z niepewnością losową- podsumowanie

$$r_j(\mathbf{x}) = 1 - P(C_j|\mathbf{x})$$

- r_j - funkcja strat średnich powstałych przy zaklasyfikowaniu wektora cech **x** jako należącego do klasy abstrakcji C_j

$$\delta(\mathbf{x}) = C_k \Leftrightarrow \underset{j \neq k}{\forall} P(C_k \mid \mathbf{x}) > P(C_j \mid \mathbf{x})$$

<u>Reguła decyzyjna:</u> wektor cech **x** powinien być zaklasyfikowany do tej klasy abstrakcji C_k, dla której prawdopodobieństwo a'posteriori $P\,(C_k|\mathbf{x})$ jest największe

Cyran K. A., Jaroszewicz L. R., Automatic Image Recognition Based on Diffraction Method and Artificial Neural Network, Stare Jabłonki 2000,

Kryterium odrzucenia

- Ponadto w modelu z niepewnością losową można zdefiniować tzw. kryterium odrzucenia:

$$\max_k P(C_k \mid \mathbf{x}) \begin{cases} \geq \theta & \Rightarrow \quad \text{sklasyfiku j } \mathbf{x} \\ < \theta & \Rightarrow \quad \text{odrzu ć } \mathbf{x} \end{cases}$$

Podstawowy problem klasyfikacji statystycznej a SN

- Ponieważ reguła decyzyjna opiera się na porównaniu prawdopodobieństw a posteriori przynależności wektorów cech do klas, zatem aby ją stosować należy znać te prawdopodobieństwa
- Jak zostanie pokazane za chwilę, właściwie nauczone sieci neuronowe potrafią aproksymować te prawdopodobieństwa z wzorców uczących.

Sieci neuronowe w klasyfikacji statystycznej - założenia

- Niech klasyfikator neuronowy uczony jest dużą liczbą przykładów uczących w postaci par ($\mathbf{x}$, C_j) gdzie $\mathbf{x}$ jest wektorem wejściowym (wektorem cech) a C_j prawidłową klasą abstrakcji odpowiadającą temu wektorowi. Indeks $j = 1, ..., K$, odpowiada numerowi klasy abstrakcji, a K jest ilością klas do rozpoznania.
- Niech pary ($\mathbf{x}$, C_j) mają rozkład prawdopodobieństw p ($\mathbf{x}$, C_j)
- Niech y_k ($\mathbf{x}$) będzie wartością pojawiającą się na k-tym neuronie wyjściowym. Oczywiście zachodzi również: $k = 1, ..., K$.

Sieci neuronowe w klasyfikacji statystycznej - założenia (cd.)

- Wymagane odpowiedzi sieci oznaczane przez T_{kj} spełniają zależność:

$$T_{kj} = \begin{cases} 1 & \Leftrightarrow \quad k = j \\ 0 & \Leftrightarrow \quad k \neq j \end{cases}$$

Sieci neuronowe w klasyfikacji statystycznej – problem uczenia

- Uczenie jest minimalizacją sumy funkcjonału błędu średniokwadratowego liczonej po wszystkich klasach i wszystkich wektorach **x** proporcjonalnie do ich rozkładu prawdopodobieństw. Jest zatem minimalizacją wyrażenia *E* danego wzorem:

$$E = \int_{\mathbf{x}} \left(\sum_{j} p\left(\mathbf{x}, C_{j}\right) \sum_{k} \left(T_{kj} - y_{k}\left(\mathbf{x}\right)\right)^{2} \right) d\mathbf{x}$$

Sieci neuronowe w klasyfikacji statystycznej – uczenie (cd.)

- Przedstawmy funkcję błędu *E* jako:

$$E = \int_{\mathbf{x}} \left(\sum_{k} \left(\sum_{j} p\left(\mathbf{x}, C_j\right)\left(T_{kj} - y_k\left(\mathbf{x}\right)\right)^2 \right) \right) d\mathbf{x} = \int_{\mathbf{x}} \left(\sum_{k} E_{\mathbf{x}k} \right) d\mathbf{x}$$

- Gdzie:

$$E_{\mathbf{x}k} = \sum_{j} p\left(\mathbf{x}, C_j\right)\left(T_{kj} - y_k\left(\mathbf{x}\right)\right)^2$$

Sieci neuronowe w klasyfikacji statystycznej – uczenie (cd.)

- Ponieważ E_{xk-} jest dodatnie dla dowolnego **x** oraz k, zatem minimalizując go minimalizuje się również E.
- Rozdzielając wyrażenie na E_{xk} na dwie części dla: $j = k$ oraz $j \neq k$ dostajemy:

$$E_{\mathbf{x}k} = p\left(\mathbf{x}, C_k\right)\left(1 - y_k\left(\mathbf{x}\right)\right)^2 + \sum_{j \neq k} p\left(\mathbf{x}, C_j\right) y_k^2\left(\mathbf{x}\right)$$

Sieci neuronowe w klasyfikacji statystycznej – uczenie (cd.)

- Kolejno przekształcamy to wyrażenie:

$$E_{\mathbf{x}k} = p(\mathbf{x}, C_k)\left(1 - 2y_k(\mathbf{x}) + y_k^2(\mathbf{x})\right) + \left(p(\mathbf{x}) - p(\mathbf{x}, C_k)\right)y_k^2(\mathbf{x})$$

$$E_{\mathbf{x}k} = p(\mathbf{x}, C_k) - 2p(\mathbf{x}, C_k)y_k(\mathbf{x}) + p(\mathbf{x})y_k^2(\mathbf{x})$$

$$E_{\mathbf{x}k} = p(\mathbf{x})\left(p(C_k|\mathbf{x}) - 2p(C_k|\mathbf{x})y_k(\mathbf{x}) + y_k^2(\mathbf{x})\right)$$

$$E_{\mathbf{x}k} = p(\mathbf{x})p(C_k|\mathbf{x}) + p(\mathbf{x})\left(-p^2(C_k|\mathbf{x}) + p^2(C_k|\mathbf{x}) - 2p(C_k|\mathbf{x})y_k(\mathbf{x}) + y_k^2(\mathbf{x})\right)$$

Sieci neuronowe w klasyfikacji statystycznej – uczenie (cd.)

- Otrzymujemy ostatecznie:

$$E_{\mathbf{x}k} = p(\mathbf{x})p(C_k|\mathbf{x})(1 - p(C_k|\mathbf{x})) + p(\mathbf{x})(p(C_k|\mathbf{x}) - y_k(\mathbf{x}))^2$$

- Widać, że E_{xk} jest minimalizowane gdy:

$$y_k(\mathbf{x}) = p(C_k|\mathbf{x})$$

SSN przybliżają na wyjściach prawdopodobieństwa a'posteriori

SSN generuje na wyjściu y_k dla wektora wejściowego $\mathbf{x}$ błąd $E_{\mathbf{x}k}$ dany przez:

$$E_{\mathbf{x}k} = p(\mathbf{x})p(C_k|\mathbf{x})(1 - p(C_k|\mathbf{x})) + p(\mathbf{x})(p(C_k|\mathbf{x}) - y_k(\mathbf{x}))^2$$

Ponieważ pierwszy składnik sumy nie zależy od y_k, błąd jest minimalizowany gdy:

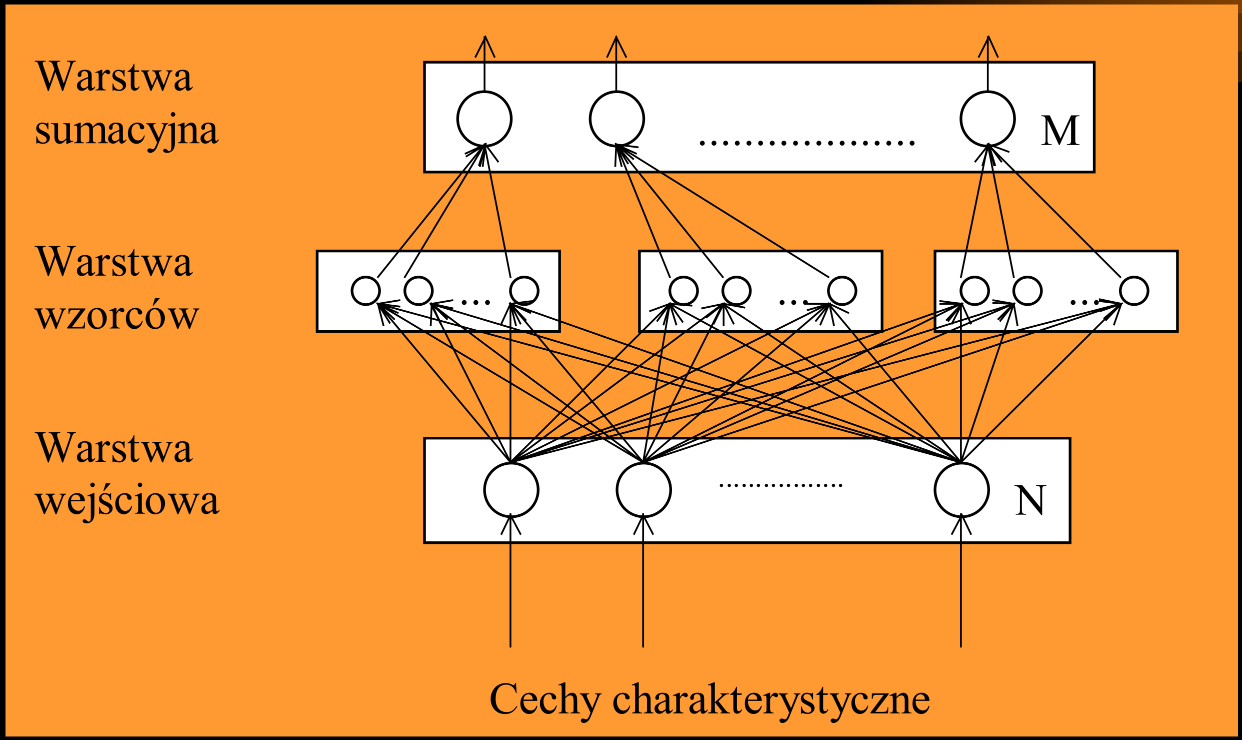
$$y_k(\mathbf{x}) = p(C_k|\mathbf{x})$$

Z drugiej strony uczenie SSN minimalizuje błąd $E_{\mathbf{x}k}$,, dlatego: *właściwie wytrenowane SSN przybliża prawdopodobieństwa a'posteriori poszczególnych klas abstrakcji C_k*

Probabilistyczne sieci neuronowe

- Są dedykowane do zadań klasyfikacji statystycznej
- Uczenie jednoprzebiegowe
- Są specjalizowanymi sieciami RBF dedykowanymi do generowania estymatorów jądrowych gęstości warunkowych prawdopodobieństw wykorzystywanych w klasyfikacji statystycznej

Probabilistyczne sieci neuronowe - struktura



Probabilistyczne sieci neuronowe - estymacja jądrowa

Oznaczenia:

zbiór $V_j = \{\mathbf{x}(s) \in \Re^N, 1 \le s \le S_j\}$ wektorów cech należących do klasy C_j

$$\hat{p}\left(\mathbf{x} \middle| C_j\right) = \frac{1}{S_j} \sum_{\mathbf{x}(s) \in V_j} \frac{1}{h(s,j)^N} K\left(\frac{\mathbf{x} - \mathbf{x}(s)}{h(s,j)}\right)$$