

10.1.2 Výstupní neurony

Pro nespojitou výstupní funkci má odezva tvar válece jehož poloměr odpovídá poloměru sféry vlivu $R = 4$. Válec do sebe na obrázku vzájemně do promítají, což je přípustné pouze pro neurony klasifikující stejnou třídu vzorů. Pro neurony klasifikující různé třídy je průnik naopak nežádoucí, protože by byl jeden vstupní vektor rozpoznáván v různých kategoriích.

Uvedené příklady vycházejí z dvourozměrného vstupního prostoru (dvouúvstupové RBF neurony). V praxi jsme však často nuceni přejít zobecnění na mnohem více dimenzí, např. 64. Takové prostory však nelze názorně zobrazit.

Neurony ve výstupní vrstvě RBF sítě jsou perceptronovského typu a jsou definovány takto:

$$y = \sum_{i=1}^n w_i y_i$$

kde y je výstup RBF neuronu, w_i jsou váhy a y_i je výstup neuronové sítě. Úkolem výstupních neuronů je nasčítat příspěvky od RBF neuronů tak, aby požadovaná funkce byla aproximována co nej přesněji.

10.2 Učení neuronových sítí typu RBF

10.2.1 Učení neuronů RBF vrstvy

Pro správné pochopení učících algoritmů je třeba mít na paměti, že je se jedná o učení s učitelem. A to i přes to, že stanovení sítě RBF neuronů přímo nevyžaduje znalost požadovaných výstupních hodnot. Učení probíhá na základě trénovací množiny, která je totožná s trénovací množinou používanou pro perceptrony (např. pro algoritmus Back-propagation). Trénovací množinu tvoří páry **vstup-výstup** sítě. Pro úlohy typu aproximace jsou to páry **argument funkce-funkční hodnota**, pro klasifikační úlohy to jsou páry **vzor-kategorie**.

Učení RBF sítě je rozděleno na dvě fáze. V první fázi se určí prototyp $\bar{C} = \{c_1, \dots, c_n\}$ a σ pro každý RBF neuron. Tento proces probíhá bez znalosti funkčních hodnot nebo kategorií. Pracuje se jen se vstupními hodnotami. Pro tuto fázi se používají algoritmy podobné algoritmu pro shlukovou analýzu, nebo je možno využít algoritmy pro Kohonenovy sítě.

Pro maximální urychlení první fáze je také možno využít neadaptivních metod. Např. rovnoměrně nebo náhodně rozložení sítě RBF neuronů po vstupním prostoru. Nastavení můžeme rovněž provést tak, že pro každý RBF neuron náhodně vybereme jeden vzor ze vstupních dat a ten použijeme přímo jako prototyp. Tato metoda má na rozdíl od předchozích dvou výhodu v tom, že reflektuje rozložení dat po vstupním prostoru.

Druhá fáze učení má za úkol určit váhy výstupních neuronů. Vzhledem k charakteru výstupních neuronů může me použít metodu nejmenších čtverců nebo gradientní algoritmy.

Standardní K-means algoritmus

Při popisu tohoto učícího algoritmu vycházíme z množiny $\{X^{(i)} : i = 1, \dots, T\}$, kde $X^{(i)}$ je vstupní vektor a i určuje jeho pořadí v trénovací množině. T je celkový počet vektorů. Nejprve odhadneme kolik shluků může ve vstupních datech být. Toto musíme udělat prostě ad-hoc, na základě zkušenosti, nebo prostě tipovat s vědomím, že později můžeme odhad upravit. Řekněme, že odhadneme p shluků, pak dostáváme $\bar{C}_1, \dots, \bar{C}_p$ vektorů, které určují sítě RBF neuronů. Každý vektor má n složek, jejichž počet odpovídá počtu vstupů RBF neuronu.

Dále definujeme funkci příslušnosti m vzoru ke shluku:

$$m(X^{(i)}) = k \text{ jestliže } \bar{C}_k \text{ je nejbližší shluk ke vzoru } X^{(i)}$$

Kroky K-means algoritmu: