Practicum FGREP

Peter Kleiweg

Januari 1996

1 Bestanden

Tabel 1 geeft een overzicht van de bestanden die bij dit practicum worden gebruikt.[1] Omdat er onder Unix al een programma is met de naam `fgrep` is hier de naam `miik` gebruikt. Onder Unix geldt: `miik` moet executie-permissie hebben. De programma's uit het practicum Topografische Kaarten worden gebruikt voor het analyseren van resultaten.

2 Het programma `miik`

Het programma `miik` is een non-recurrente implementatie van Miikkulainen's FGREP algoritme. Het gebruikt een feedforward netwerk met één laag verborgen units. De activatiefunctie voor de verborgen en de uitvoer-units is:

$$f(x) = \frac{1}{1 + e^{-x}}$$

Er wordt geen momentum gebruikt.

Geef het commando `miik` zonder parameters om een overzicht te krijgen van de geldige parameters. Het bestand `sentence.pat` is een voorbeeld van de syntax in een bestand met patronen.

Een patroon dat bij de training een hoogste fout geeft die kleiner is dan de maximaal toegestane fout wordt niet gebruikt bij de backpropagation.

1. Het bestand `sentence.pat` is geschreven door Erik F. Tjong Kim Sang.

Unix	DOS
`Makefile`	`MAKEFILE`
`miik`	`MIIK.EXE`
`miik.c`	`MIIK.C`
`sentence.pat`	`SENTENCE.PAT`
`miik_dat.pl`	`MIIK_DAT.PL`

Tabel 1: Bestanden voor practicum FGREP

Voor elk epoch wordt de volgorde van de patronen voor de training willekeurig veranderd.

Aanpassing van de patronen gebeurt indirect via de denkbeeldige gewichten van denkbeeldige verbindingen naar de invoerlaag. Op deze manier blijft de aanpassing van de patronen zo dicht mogelijk bij de regels voor de aanpassing van de gewone gewichten in standaard backpropagation. De eenvoudiger methode die door Miikkulainen wordt gebruikt leidt er toe dat de waarden van de vectoren heel snel naar 0 en 1 gaan, waarna ze moeilijk veranderen. Hierdoor ontstaan minder goede representaties.

Er worden door het programma twee patronen gedefinieerd: `!NILL` en `!ANY` (met uitroepteken!). De waarden van deze vectoren worden constant gehouden op allemaal 0.05 voor `!NILL` en 0.5 voor `!ANY`.

Het programma laat een aantal waarden op het scherm zien, die hieronder worden toegelicht:

Totaal De som van alle fouten.

Hoogste De hoogste fout.

>0.15 Het aantal fouten groter dan 0.15. Deze waarde verandert met de waarde voor de parameter `-q`.

Dev.W1 Standaarddeviatie van de gewichten tussen invoerlaag en verborgen laag.

Dev.W2 Standaarddeviatie van de gewichten tussen verborgen laag en uitvoerlaag.

Gem.W1 Gemiddelde waarde van de gewichten tussen invoerlaag en verborgen laag.

Gem.W2 Gemiddelde waarde van de gewichten tussen verborgen laag en uitvoerlaag.

Dev.Vec Standaarddeviatie van de waarden in de vectoren.

Gem.Vec Gemiddelde waarde van de waarden in de vectoren.

De waarden van **Totaal**, **Hoogste** en **>0.15** zijn benaderingen, totdat **Hoogste** onder de maximale fout komt, of het maximale aantal beurten is bereikt. Op dat moment worden de waarden precies bepaald. Een verhoging van de deviaties tijdens het leren wijst er meestal op dat het programma leert.

3 Opgaven

Train het programma met de zinnen uit het bestand `sentence.pat`. Bepaal zelf de waarden voor de diverse parameters zoals bijvoorbeeld het aantal verborgen units, het aantal units per vector, en de leerconstante. Je kunt het leerproces in delen uitvoeren door de parameter `-m` te gebruiken, en na elk deel verder gaan met de oude resultaten (gebruik `-w` en `-l`), maar met een andere leerconstante. Bekijk de resultaten (die bewaard zijn in een bestand met de extensie `.vec`) met het merge clustering algoritme. Gebruik eventueel ook een Kohonen-kaart. Woorden die op gelijke wijze worden gebruikt moeten dicht bij elkaar komen te staan. Is dit gelukt? Herhaal het experiment een aantal maal met andere parameters tot een goed resultaat wordt bereikt.

Train het programma met een grotere set patronen. Gebruik meer en langere zinnen, meer verschillende woorden en meer *case-role* posities. Analyseer het resultaat.

Het bestand `miik_dat.pl` is een prolog-programma dat alle 1475 zinnen genereert die Miikkulainen en Dyer hebben gebruikt [MD91, Appendix A, p. 395–396]. Ze gebruikten vectoren van lengte 12, 25 verborgen units, $\eta = 0.1$ voor de eerste 200 epochs, daarna $\eta = 0.05$ voor de volgende 300 epochs, en tenslotte nog 100 epochs met $\eta = 0.025$. Ze gebruikten 1439 zinnen voor de trainingsfase, en de overige 36 voor de testfase. Meer details vind je op blz. 352 van hun artikel. Start prolog, consult het bestand, geef het commando `info`, en volg de aanwijzingen. Je kunt een (kleine) selectie van de zinnen gebruiken voor je experiment, of je kunt het aantal woorden in `miik_dat.pl` beperken.

Referenties

[MD91] Risto Miikkulainen and Michael G. Dyer. Natural language processing with modular PDP networks and distributed lexicon. *Cognitive Science*, 15:343–399, 1991.