Øving 4 – Artificial Neural Networks

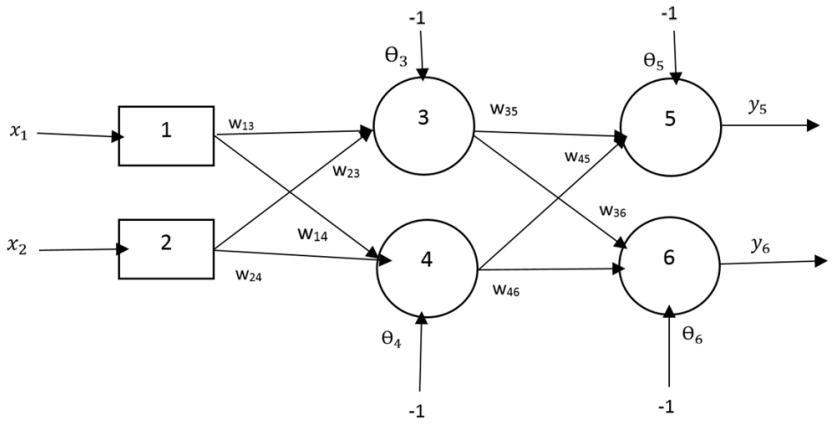
Frist: Fredag 27. Oktober

Legg besvarelse på spørsmål og tilhørende kode i en zip-fil og last opp zip-filen.

a) Betrakt læringsregelen til et perceptron, se foiler fra forelesning eller side 172 i utdelt kapittel på ANN (Artificial Neural Networks). Kod algoritmen i et valgfritt programmeringsspråk.

b) Kjør læringsalgoritmen i a) med input fra AND og OR og legg ved utskrift på hvordan vektene justerer seg. Du kan legge inn dataene direkte i koden. Blir vektene de samme når du forandrer startverdier på vektene w_i og terskelen Θ?

c) I denne oppgaven skal vi regne på én iterasjon med **backpropagation** på et nevralt nett som vist i figuren nedenfor.



La input, ønsket output ($y_{d,n}$), vekter/thresholds, læringshastighet være gitt ved:

$x_1 = 0, x_2 = 1,\ y_{d,5} = 0, y_{d,6} = 1, \alpha = 0.1, w_{13} = 0.5, w_{14} = 0.0,\ w_{23} = 0.0, w_{24} = 0.9, w_{35} = 0.4, w_{36} = 1.0, w_{45} = -1.2, w_{46} = 1.1,\ \Theta_3 = 0.8, \Theta_4 = -0.1, \Theta_5 = 0.3, \Theta_6 = 0.5$

Hva vil verdiene til vektene/thresholds være etter én ny iterasjon?

Variasjoner man kan øve på: Fjern nevron 6 og gjenta øvelsen.

d) I den neste delen av øvingen skal vi utforske et **feed-forward-nettverk**. Det finnes mange biblioteker som kan hjelpe med maskinlæring gjennom nevrale nettverk. Vi legger opp til å benytte Python og biblioteket "PyBrain", men dersom du finner noe annet du vil bruke står du fritt til det. PyBrain gir oss snarveier til å komme raskt i gang og lar oss samtidig finjustere alle deler av nettverkene, treningsalgoritmene og datasettene vi setter opp.

Pybrain bruker endel underliggende teknologi som må være installert. Dersom du ikke har disse installert fra før eller du støter på problemer under installasjonen kan du installere Anaconda-distribusjonen av Python. Den har alt vi trenger.

Pybrain er et nokså stort bibliotek og man kan gjøre mye forskjellig på et detaljert nivå. Vårt mål er bare å stikke tåa i vannet og derfor skal vi bare benytte oss av enkel funksjonalitet for å utforske nevrale nett. Etter installasjonen kan du ta en titt under "Quickstart" for å se hvordan du kan bruke noen snarveier i biblioteket for å raskt komme i gang. Merk at metoden "trainUntilConvergence()" ikke vil fungere bra på datasettet i XOR-eksempelet. Grunnen er at metoden, uten eksplisitte parametere, deler opp datasettet i 75% til trening og 25% validering (test). Ettersom datasettet i XOR-eksempelet er så lite betyr det at når 25% blir fjernet vil den resterende biten ikke dekke løsningsområdet tilstrekkelig. Dermed blir nettverket ikke tilstrekkelig trent og forsøk på klassifisering av ny input gir dårlige resultater.

Vi skal lage et nevralt feed-forward nettverk hvor **input er lik output** (kalles en **auto-enkoder**). Vi vil undersøke **hvor stort** nettverket må være for å gjenskape oppførselen vi ønsker.

1. Lag først et datasett som mapper input->output for auto-enkoderen:
   1->1, 2 >2, 3->3, .., 8->8
2. Bygg et nettverk med 1 input-node, 8 hidden-noder og 1 output-node.
   Bruk _"TanhLayer" som aktiveringsfunskjon "hiddenclass".
3. Lag en backpropagation-trainer basert på datasettet og nettverket (du skal her ikke kode backpropagation fra grunnen av, men bruke biblioteket).
4. Tren til konvergens. Her er passende parametere: trainUntilConvergence(verbose = False, validationProportion = 0.15, maxEpochs = 1000, continueEpochs = 10).
5. Aktiver nettet på forskjellige heltall slik som input-dataene i 1) og vurder resultatet.
6. Resultatet blir antageligvis bra med 8 tilgjengelige hidden-noder og dette er kanskje ikke så overraskende? Vi kan for eksempel forestille oss at for å løse problemet så lærer nettverket at den kan aktivere en distinkt node for hver input. Gjenta derfor eksperiementet med færre og færre **hidden**-noder.

Besvar nå følgende tre spørsmål:

1. Hva er det laveste antall hidden-noder som gir et godt resultat?
2. Hva er det det nevrale nettet har gjenskapt gjennom hidden-laget for å produsere et bra resultat med dette minste antallet noder i hidden-laget?
3. Aktiver nettverket med tall du ikke har trent med, som desimaltall, negative tall og tall over 8. Kommenter hvor godt nettverket håndterer disse nye input-dataene.

Lever inn koden du har skrevet og besvarelsen på de tre spørsmålene 1-3 i d) ovenfor. Du kan levere koden som tekst sammen med svarene.