Projekt II – Neuralt nätverk i ett inbyggt system

4 poäng (G = 2p, VG = 4p)

Mål

- Kunna implementera neurala nätverk från grunden via mjukvara och använda för inbyggda system.
- Känna till och kunna beskriva begrepp såsom *feedforward, backpropagation* samt aktiveringsfunktioner.
- Erhålla fördjupad kunskap inom C++, Python eller Rust.

Uppgiftsbeskrivning

- Ni ska i grupper om 1 - 2 skapa ett inbyggt system innehållande ett neuralt nätverk konstruerat från grunden i C++, Python eller Rust. Detta nätverk ska kunna prediktera en utsignal via insignaler från fyra tryckknappar, där predikterad utsignal används för att styra en lysdiod. För det inbyggda systemet ska en Raspberry Pi användas.

- Om ett ojämnt antal tryckknappar är nedtryckta ska lysdioden tändas, annars ska den vara släckt. Därmed ska det neurala nätverket tränas till att prediktera en hög utsignal om en eller tre knappar är nedtryckta, annars en låg utsignal. Samtidigt som lysdioden tänds eller släcks ska det neurala nätverkets predikterade utsignal skrivas ut i Linuxterminalen, tillsammans med information gällande om lysdioden tänds eller släcks.

- Nätverket ska implementeras från grunden, alltså utan användning av externa bibliotek. För det neurala nätverket ska användaren kunna välja lärhastighet, antalet epoker samt antalet noder i in- och utgångslagret. Användaren ska också kunna lägga till valfritt antal dolda lager med valfritt antal noder i respektive dolda lager. Träningsdata ska kunna läsas in via en textfil alternativt genom att passera träningsdatan via arrayer.

- Se till träning sker direkt när systemet startar och att nedtryckning av någon eller några av tryckknapparna inte medför någon prediktion förrän träningen är slutförd.

- Använd gärna följande referensmodell skriven i C för tips på implementering av det neurala nätverket: https://github.com/hp22-ela21/Neural-Network-C.git

- Vänligen kopiera inte något av referensmodellen, varken programkoden eller kommentarerna, och följ kutym för det programspråk ni använder (använd klasser i stället för struktar med funktionspekare, använd vektorer/listor i stället för att skapa egna sådana med mera).

Examination

- Koden ska lämnas in via ett repo på *GitHub.* Koden ska dokumenteras i vanlig ordning (flerradiga kommentarsblock på engelska eller svenska). Använt programspråk (C++, Python eller Rust) påverkar inte poängen, dock prestandan!

- Validering av systemet kan demonstreras antingen på en lektion eller via en kort film som lämnas in på Classroom. Vid validering ska träning av modellen samt prediktion i enlighet med uppgiftsbeskrivningen (hög utsignal vid udda antal nedtryckta tryckknappar) demonstreras.

- **För godkänt (G = 2p)** ska det neurala nätverket fungera såsom avsett, koden ska vara hyfsat genomarbetat och lättförståelig med hyfsad dokumentation. Det ska finnas möjlighet att lägga till valfritt antal dolda lager vid start, men krav på omallokering (tillsättning av extra lager i efterhand utan minnesläckor med mera).
  Ni har genomfört projektet med en relativt hög grad av handledning/hjälp.

- **För väl godkänt (VG = 4p)** ska det neurala nätverket vara välfungerande, koden ska vara väl genomarbetat och uttänkt med lättförståelig kod som inte kräver extrem mängd kommentarer för att förstå vad som sker. Dokumentationen ska vara genomarbetat med beskrivning av syftet med funktionen (inte exakt vad som sker inuti, det ska er kod vara tillräckligt tydlig för) samt kortfattat beskrivning av funktionsparametrar. Medlemmar i klasser/struktar ska beskrivas kortfattat. Det ska finnas möjlighet att lägga till lager eller ändra storlekar på ett givet lager när som helst, alltså inte bara vid initiering. Programmet ska fortfarande fungera fläckfritt. Ni har genomfört projektet till stor del självständigt.