Masterpresentasjon

Hei, jeg heter Jiader Chou og jeg skal presentere terreng klassifikasjon ved bruk av 3D optical tactile sensor

**Overview**

Først en oversikt over hva jeg skal gå igjennom. Jeg skal si litt om bakgrunnen/motivasjonen for denne oppgaven. Deretter skal jeg si litt om målene for denne oppgavene. Jeg skal vise implementasjonen som oppsett av experimenter, forklare kjapt algoritmene og typer features og si litt hvordan jeg har evaluert sensoren. Også skal jeg vise litt av resultatene som jeg har fått under eksperimentene og hvilke forbedringer jeg har gjort underveis. Til slutt skal jeg komme med en konklusjon samt en videre arbeid.

**Background**

Først litt om bakgrunnen og motivasjonen for denne oppgaven. Hvorfor terreng klassifikasjon? Ta et eksempel på vi mennesker har en egenskap til å gå på forskjellige måter på forskjellige terrenger. Som for eksempel: På en isete eller glatt vei vil man på en måte gå forsiktig for å unngå å falle og skade seg, mens på en tørt vei vi har muligheten til å løpe uten problemer. Denne egenskapen å bytte forskjellige gå mønster er mulig å anvende på roboter, men den må den klare å identifisere hva slags terreng den er eller går på.

For å kunne skille mellom terrenger må man ha sensor for å hente informasjon fra terrenget. Det er brukt en del sensorer og tidligere forskning har vist at tactile sensor er en av dem som gir mest informasjon. Data som måles er ved direkte kontakt mellom sensor og terrenget. Det finnes mange typer tactile sensor, og mye av tidligere forskning rapporterer ikke hva slags type er brukt. Typen av tactile sensor kan være avgjørende på oppnå en god resultat på terreng klassifikasjon. I denne oppgaven brukte jeg en 3D optical tactile sensor som ser sånn ut. Den bruker lys intensiten til å måle kraften. Denne typen sensor har også fordelen med å måle i tre forskjellige retninger, x,y, og z. Fordelen med denne sensoren er høy sensitivitet, liten og lett, og lav deteksjon tid.

**Goal of the thesis**

Målet for denne oppgaven er å sjekke hvordan det er å bruke denne typen sensor på å identifisere terrenger. Og finne det beste features og klassifikator for denne sensoren.

**Implementasjon**

De neste slidene skal jeg vise dere litt av implementasjonen og verktøy som har blitt brukt min. Først skal jeg vise dere roboten, deretter hvilke terrenger som er brukt, hvilke og hvordan jeg har hentet sensor dataene, hvilke features jeg har brukt og hvordan jeg evaluerte sensorene.

**Robot**

Her er roboten som jeg brukte. Roboten er utviklet på ifi, of kalles for «dyret» og jeg har montert sensorene som på føttene som det kan ses.

**Enviroment setup**

Jeg har til sammen 4 forskjellige terrenger: gulvet, teppe, myk matte og hard matte. Litt av grunnene for å velge disse er at gulvet, teppe og hard matte har ganske like egenskaper. Hardheten er ganske lik og alle har omtrent samme friksjon. Grunnen er at hvis den klarer å skille ut terrenger med ganske like egenskaper, da bør den også klare skille mellom mer ulike terrenger. Og det er også derfor jeg har tatt med myk matte som utskiller fra de andre med mye friksjon og er myk.

**Segmentation of sensor data**

Litt om hvordan jeg har hentet data. Her er det en eksempel på sensor data på en av terrengene. De dataene som jeg er interessant er de med mye endringen, for da er det kontakt mellom sensor og terrenget. Der hvor det er nesten konstant er foten i luften. Disse karakteristikkene brukes for å få segmentere data. Jeg detekter når sensor er i luften og begynner å lagre data når den er i bakken dvs. Når det er mye endringer. Disse dataene blir lagret til en fil som blir trening og test data.

**Feature sets**

Her er en oversikt over forskjellige feature sett som jeg har laget. Den første er bare rå data som er hele sekvensen som vist tidligere. Sett to er statistiske feature i tids domene, som max, min..etc. Set tre og fire er feature i frekvens domene og ligner mye på set 1 og 2. Set 5 er en kombinasjon av set 2 og set 4.

**Classifiers**

Her er liste over klassifikatorene som jeg skal bruke, nerual network, naive bayes, SVM, decision tree og KNN. I begynnelsen har jeg forløpig bare brukt default verdier siden de gir ofte greie resultater og tune parameter er tidkrevende.

**Test machine learning model**

Her er prossesen på hvordan jeg tester de nåværende data jeg har. Starter med å lese en fil, deretter ekstrakterer en av feature settene som vist tidligere. Dataene blir splittet til en test data og en trening set. Skaleringen er stiplet fordi den kan utelates, slik at jeg tester de forskjellige både med og uten skaling for å se om det er lurt å ta med. Deretter trenes klassifikatoren tester klassifikator med test settet. Lagrer det som en del av hele resultatet. Hvis det er mer data så splitter den ny data og samme prosess. Hvis det ikke er mer så gis en samlet resultat.

**Evaluation**

I denne oppgaven er det til sammen 6 eksperimenter, men den kan deles i tre deler.

Den første delen er å finne det beste feature set og klassifikator. Så jeg vil først teste implementasjonen som jeg viste. Deretter skal jeg fobedre noen av modellene ved å integrere nye funksjoner, teste på nye data og tune parameter som jeg kommer tilbake senere. Den andre delen bruker jeg modellen dvs klassifikator og feature set som ga høyest nøyaktighet til å teste med real-time implemtasjon ved å la den går på to forskjellige terrenger. Den siste del var å sjekke om det er mulig å bruke treningsdataene fra venstre front fot til å predikere data fra høyre front fot.

**Result**

Her er det resultatene på implementasjoen min. Grønn er over 90 % gul er fra og med 80% til og med 89% og rød er under 80%. Som det kan ses så ligger mye av resultatene på mellom 80% til 89%. De beste er fra feature set 1 med over 90 %. Skalering ga stort utsalg spesielt på SVM som gikk fra 52 % til 93%.

**Integrating feature selection**

I neste experiment integrerte jeg feature selction for å sjekke om er noen forbedring. Jeg har tester to forskjellige metoder, filter og wrapper. Jeg skal ikke gå i detalj på hver av dem men kort fortalt, filter bruke statestikk til å plukke relevante features, mens wrapper bruker en klassifikator til å plukke features.

**Results**

Her er resultatene. Grønn er forbedring, mens gul er samme resultat og rød blitt dårligere. Som det kan ses så har ikke er ikke mange som har blitt forbedred seg så mye, bortsett fra SVM med feature set 3 som har gått fra 93% til 97%. Noen av grunnene for at filter har gitt dårlige resultater kan være at den har fjernet for mye av featurene. De som jeg har markert med fet skrift skal jeg brukes videre experimenter som er vist her.

**Test on new data**

Disse her er de som er har gitt gode resultater. Disse skal jeg bruke til å teste på nye data set. For jeg kan jo hatt flaks med forrige data set med at de har vært gode, og hvor mange ganger jeg kjører den vil jeg forsåvidt få ganske samme skår. Derfor har jeg lagret helt nye test data som jeg skal bruke på å teste hver av modellene.

Man kan lure på hvorfor jeg har også tatt med den siste, er fordi de over bruker enten hele sekvensen i tids domene eller frekvens domene. Derfor har jeg tatt med den som bruker statistikk features i begge domene for å sjekke hvor god den er.

**Results**

Her er det en boxplot diagram av resultatene jeg har fått. Som dere kan se så har alle minket nøyaktigheten. SVM skårer fortsatt veldig bra, mens neural network som hadde den dårligste skår ligger nå på en annen plass. De andre har synket drastisk ned, og kan tyde på at bruk av rådata ikke har generalisert dataene så bra og gitt derfor en dårligere resultat. I videre eksperiment brukte jeg de to som ga høyest resultatene som er SVM og neural network til å finne parameterne.

**Hyperparameter tuning**

I denne eksperimentet plukket jeg ut de to beste fra forrige ekseperiment til å finne de beste paramterne som kan være med å gjøre forbedre resultatene. Jeg brukte grid search som søker igjennom alle parameterne verdiene som jeg har satt og gir ut den de beste verdiene, de beste verdiene blir videre testet for å se om det har gitt noen forbedring. Her er parameterne som jeg skal teste for SVM. Skal ikke gå i detalj på dem, men kjernen er den som beregner likheten mellom to datapunkter. Og det er forskjellige måter å bergene den på. Parameter C sier hvor mye SVM skal tillatte å ha feil klassifisering. Stor C gir bedre skille mellom datanene, men er kanskje ikke i stand til å generalisere det. For neural network tester jeg alle mulig kombinasjoner av antall neuroner i første og andre av hidden layer og finne den beste. I tillegg skal jeg representere output med binære tall. Forløpig har jeg hatt output som strings, det vil si at den alltid vil plukke den som har høyest skår. Med å representere binært så behandler man litt mer klassik, for eksempel at en neuron burde avfyres når den kommer til en terskel. Jeg har enten representert med to eller fire binære tall. Med to så vil man alltid få et svar uavhengig om andre gir ut feil resultat, mens med 4 binære tall kan føre til ukjente terrenger hvis mer enn to neuron avfyres.

**Results**

Her er litt av resultatene etter å har funnet de parameterne og testet på nytt. Som det kan ses så var det ikke mye forbedringer. Når det er sagt så har jeg ikke testet veldig mange parameter, spesielt for neural network. De har en del andre parameter som også burde blitt testet, og som kan påvirke resultatet. Men man kan jo også si at default verdier har jo gitt ganske gode resultater. Og den beste er fortsatt SVM med feature set tre. Etter denne eksperimentet så kommer jeg til å bruke denne modellen i resterende eksperimenter.

**Classification in real-time**

I denne eksperimentet har jeg latt roboten å gå på to forskjellige terrenger det vil si at jeg sjekker hvordan den presterer ved overgang mellom to forksjellige terrenger. For å kunne teste dette her har jeg implementert real-time.

**Real-time implementation**

Her er prosessen av real time implemtasjonen min. Mye er samme som før. Den leser fil, ekstrakterer features plukker feature, skalerer den trener opp modellen, også kommer den nye prosessen som er å lage en ROS service. ROS service er på en måte en server som kan tar imot data fra en klient. Slik at den vil vente på om det kommer noe sensor data. Sensor data er segementerte data som jeg viste i tidligere slide, og som også er klienten. Den sender en data sekvens og når den får data behandler den data med extraction osv. Og til slutt predikerer den hva slags terrenge det er. Etter prediksjon venter den på ny data sekvens.

**Floor to carpet**

Viser bare et av resultatene. Grafen er sensor data og den gråe streken er overgangen fra den ene terrenget til den andre. Under er tabell på en skår på de forskjellige terrenge, og hva maskinen har predikert. Grønt betyr at den har predikert riktig. Ved feil prediksjon så er rødt hva den predikerte, mens gul er den akutelle terrenget. Noe som kan være verdt å legge merke til er at steget før overgangen er spesielt lavt og det ser ut som den er relativ rask til å detekere nye terrenget. Det kan se ut som den er i stand til å skille mellom gulvet og teppe.

**Summary**

For å oppsummere litt av de eksperimentene så har den en nøyaktighet av 85.3%. Noe av grunnen er at en del kjøringer gikk ikke roboten rett frem under eksperimentene. Det som ble observert er at det er vanlig at lav sannsynlighet/ modellen er usikker på terrenget ved overgangen. Noen av grunnene kan være at treningsdataene har bare latt roboten gå rett frem noe som kan ha påvirket og. Når det gjelder hvor for den detekter det nye terrenget. Når terrengene er ganske like som jeg viste gulvet og teppe var litt treg med å detekere det nye terrenget, men hvis terrengene har større forskjeller slik som myk matte og de andre så er den raskere å detekere det nye.

**Investigate the other sensor**

Forløpig har jeg bare brukt sensor data fra den ene foten på roboten som er venstre front fot. I denne eksperimentet var tanken å teste hvordan det er å bruke sensor data fra den ene foten til å predikere den andre foten. Hvis det er mulig så betyr at man at jeg kunne bruke samme treningsdataene fra høyre front fot.

**Results**

Her er resultatet, men skal ikke si så mye om den. Man kan jo legge merke til at den sliter med harde matten. Når det er sagt så har den harde matten vært den verste å identifisere, den har vært den vanskelige å predikere riktig. Resultaten viser jo at det er mulig å bruke samme treningsdata, men hvis man vil ha bedre kvalitet på treningsdataene så er det best å bruke dataene hver for seg.

**Conclusion**

Konklusjonen av denne oppgaven er at resultatene viser at sensor som jeg brukte er egnet for terreng klassifikasjon. Fikk jo en høy skår med over 90% som kan sies til å være veldig høy. Det beste var en kombinasjon av bruke komplette frekvens domene med SVM.

**Compared to earlier work**

Bare for å vise liten sammenligning mellom tidligere arbeid så er min ganske høyt oppe i listen. Det må sies at jeg bare har brukt den ene sensor og har oppnådd bedre resultat enn de som har brukt flere. Den som er mest sammenlignbar er Degrave. Fordi den brukte lignende robot og sensor som meg, men dessverre er ikke typen av tactile sensor rapportert slik at jeg ikke for sammenlignet om det er faktisk sensor som hadde noe å si eller om det er klassifikator som ga bedre resultat.

**Future work**

Til slutt kort liste over hva som kan jobbes videre med denne oppgaven. Egenskaper som terrenget i seg selv som hardhet og friksjon er jo ikke direkte analysert. Det kan være interessant å klare å trekke ut disse egenskapene utifra sensor datane. Forløpig har jeg bare tested på flate overflater og kan være greit å inkludere andre mer ujevne terrenger/utendørs terrenger for å teste implementasjonen. Jeg for det meste testet ved å bruke den ene sensoren, men så vidt vært borti hvodan det er å bruke treningsdata på den ene foten på den andre, og det kan være greit å bruke alle sensorene til å få bedre resultat. Når det er sagt er det også mulighet for å legge til flere sensor til å forbedre resultatet. Unsupervised learning er interressant å få til, dvs la roboten selv utforske og finne ut hva slags terreng det er. Den siste er å gjøre den i stand til å bytte gå mønster på forskellige terrenger. For det er jo noe av grunnene for at roboten skal klare identifisere forskjellige terrenger.

Takk for meg!