Intelligent djurlätes identifiering

Namn	Jens Holm
Utbildning	Pythonutvecklare inriktning AI
Datum	30 maj 2025

Sammanfattning

I detta examensarbete har en webbaserad applikation utvecklats för att identifiera om ett ljudklipp kommer från en katt eller en hund. Projektet genomfördes med hjälp av Python, Flask och flera maskininlärningsmodeller som tränades på mel-spektogram av djurläten. Initialt användes ett mindre dataset, men bytet till ett större och mer balanserat förbättrade modellernas träffsäkerhet markant. Resultatet blev en fungerande prototyp med god precision (96,5-99%). Arbetet visar tydligt hur även enklare AI-verktyg kan användas effektivt i ljudklassificering. Fokuset under projektet låg på att kombinera maskininlärningsmetoder med ett användarvänligt gränssnitt.

Innehåll

1. Projektbeskrivning

Detta projekt markerar avslutet för min utbildning. Jag har valt att fördjupa mig inom artificiell intelligens i koppling till djurkommunikation eftersom jag under denna utbildning har upptäckt ett stort intresse för ämnet.

1.1 Bakgrund

Ämnet är mycket relevant och runtom i världen så investeras stora mängder pengar i denna typ av forskning. Projektet ska skildra en applikation av mindre skala, liknande ett konceptbevis.

Projektet genomförs fristående och är därmed inte knutet till något företag. Valet att utforska artificiell intelligens inom djurkommunikationsklassificering kom naturligt med insikten om området som utvecklas snabbt och visar stor potential.

Jag har tidigare under utbildningen skrivit en rapport om forskningens framsteg inom djurkommunikation tack vare artificiell intelligens, och ville själv applicera de koncept jag läst om.

1.2 Syfte

Syftet är att redogöra enklare maskinlärningsmodellers förmåga att skilja på djurkommunikation. Detta som en bit på vägen till att människor kan förstå sina djur bättre. Samt att redogöra hur användbara enklare AI verktyg är i dessa typer av klassificeringar.

1.3 Mål

Målet för slutprodukten är en flaskapplikation där användaren kan ladda upp en ljudfil från en katt eller en hund. Filen ska sedan konverteras till siffervärden och sedan evalueras av maskininlärningsmodeller. Användaren ska sedan få en prediktion på vilket djur ljudfilen härstammar från genom applikationen.

1.4 Avgränsningar och fokus

Fokuset under arbetet kommer rikta sig mot en enklare applikation med hänsyn till tidsspannet. Djuren som kommer undersökas avgränsas till hundar och katter. Även antalet prediktiva modeller begränsas för att minimera projektstorleken.

1.5 Tidsplan/aktiviteter

Arbetstiden delas upp i 4 sprintar som är på 1 vecka vardera.

Sprint 1:
Bygga flaskapplikationen, och börja förbereda data för träning.

Sprint 2:
Träna och utveckla modellerna, koppla dessa till flaskapplikationen.

Sprint 3:
Färdigställa kodprojektet, skriva rapporten.

Sprint 4:

Färdigställa rapporten tillsammans med presentationen.

2. Teknisk bakgrund och val

I detta kapitel redovisas den tekniska bakgrund och de val som format projektet.

2.1 Vad är ljud?

Ljud är mekaniska vågor, vilket kan beskrivas som en varierande täthet som färdas genom gas, vätska eller ett fast ämne. För att det ska klassas som ljud behöver vi även kunna höra det, och då behöver ljudets frekvenser falla inom intervallet för hörsel och med tillräcklig styrka.[1]

2.2 Val

Ljudklassificering med hjälp av maskininlärning är ett brett ämne, vilket tillför flera vägval. Nedan redovisas de besluts som gjorts tillsammans med korta motiveringar.

2.2.1 Dataformat:

Ljudfilerna är i waveform (.wav) format. Detta är noggrant utvalt för syftet av arbetet, eftersom de är okomprimerade. Komprimerad ljuddata, exempelvis MPEG-1 Audio Layer 3 (.mp3) tillför detaljförluster som negativt påverkar de senare analyser av datan.[2]

2.2.2 Webramverk:

Applikationen är byggd genom Flask, istället för exempelvis Django. Båda är ramverk som fungerar, men Flask är bättre anpassat till syftet av flera anledningar. Några exempel är

[1] URL1
[2] URL2.

dess lätta vikt, flexibilitet, och storleken på detta projekt, där Django har mer onödig resurskostnad avsedd för större projekt.[3]

2.2.3 Modeller:

Att välja vilken typ av modeller som används för ljudprediktionen i applikationen är ett viktigt val, som påverkar många faktorer, därbland resurskostnad, träffsäkerhet, svårighetsgrad att förstå och implementera och möjlighet att justera efter specifika dataset. Nedan listas de modeller som valdes med en kort beskrivning.

Random forest (RF). En modell med många beslutsträd som är lätt att implementera och billig att träna.[4]

Support vector machine (SVM): En robust statistisk klassificerare .[5]

eXtreme gradient boosting (XGboost). Bygger vidare på beslutträd genom att iterativt förbättra klassificeringen.[6]

Multilayer Perceptron (MLP). Artificiellt neuralt nätverk som endast går från indata till utdata utan feedbackloopar. Lättimplementerat eftersom det finns tillgängligt i sci-kit learn, som samtliga modeller.[7]

2.3 Datahämtning

Först användes ett mindre dataset för testning, som bestod av 164 katt filer (1323sek) och 113 hund filer (598 sek). Vissa ljudfiler var under 1 sekund i längd och vissa över 17 sekunder, de varierade mycket mellan varje fil.

[3] URL 3.
[4] URL 4.
[5] URL 5.
[6] URL 6.
[7] URL 7.

Den stora variationen i längd mellan ljudfilerna gjorde det svårt att skapa en enhetlig indata för modellen, vilket i sin tur påverkade både träningstiden och modellens prestanda negativt. Dessutom fanns det en tydlig obalans i både antalet filer och den totala längden mellan de två klasserna, vilket riskerade att leda till en partisk modell. För att få mer tillförlitliga resultat och förenkla förbehandlingen valdes det att leta efter ett annat dataset med mer jämnt fördelade och konsekvent långa ljudklipp.[8]

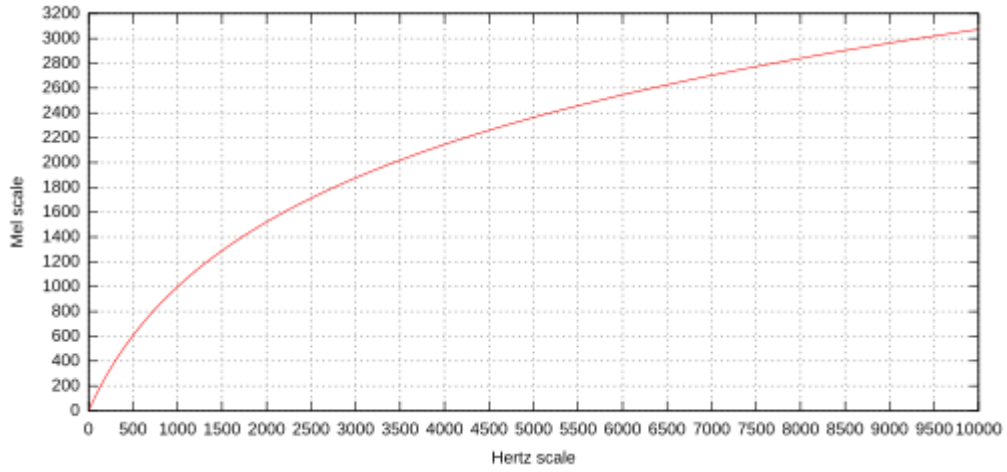
Senare hittades ett annat dataset med mindre variation i längder och fler datapunkter.[9]Detta ökade min test accuracy från 78% till 95% med samma Random forest modell, ett utmärkt exempel av vikten med mängd och kvalité i data för resultatet.

2.4 Databearbetning

Ursprungligtvis extraherades vanliga spektogrammet från ljudfilerna i berarbetningen av datan, men vid djupare undersökning så byttes metod till att extrahera mel-spektogrammet, vilket följer mel-skalan.

Vad är då Mel-skalan? Det är en perceptuell skala av tonhöjder som gör så att en lika stor skillnad långt ner på skalan låter lika långt ifrån varandra som vid slutet av skalan. Vilket det utan mel-skalning inte gör.

[8] URL 8.
[9] URL 9.



En plot av mel skalan mot hertz skalan.[10]

Efter att mel-spektrogrammen extraherats från varje ljudfil, genomfördes flera steg för att förbereda datan för maskininlärning.

Först behövde alla mel-spektrogram ha samma djup, eftersom ljudfilerna varierade i längd. Därför förlängdes eller kortades spektrogrammen längs tidsaxeln till ett gemensamt antal tidssteg, vilket möjliggjorde att all data kunde representeras i samma matrisstorlek. Detta är viktigt för att kunna använda traditionella maskininlärningsmodeller som kräver att alla indata har samma format.

När alla mel-spektrogram var lika långa, plattades de till en endimensionell vektor. Detta görs för att kunna mata in dem i de enklare scikit-learn-modellerna.

Nästa steg var att normalisera datan. Detta innebär att varje punkt i vektorn skalades så att den fick medelvärde 0 och standardavvikelse 1. Normalisering är ett kritiskt steg för många algoritmer, särskilt för att förbättra prestanda och konvergens i metoder som Principal Component Analysis (PCA).[11]

[10] URL 10.
[11] URL 11.

För att minska datans dimension, samtidigt som så mycket information som möjligt bevarades, applicerades Principal Component Analysis (PCA). Målet var att behålla 99 % av datans variation. Denna reduktion av dimensioner gör datan mer hanterbar och reducerar risk för överanpassning (overfitting) i senare klassificeringsmodeller.

Detta resulterade i att över ett tusen vektorer sparades i en csv (comma separated values) fil, där varje rad representerade en ljudfil med 127 datapunkter och en label som var katt eller hund.

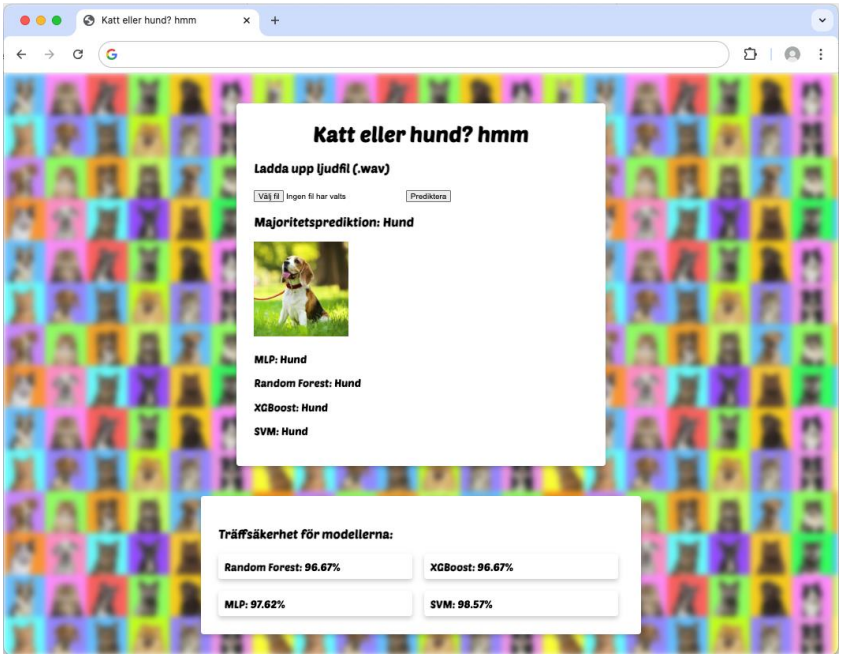
3. Resultat

Resultatet av arbetet är en webbaserad enkelsidad applikation där en användare kan ladda upp en ljudfil och sedan får en prediktion gällande om man hör en hund eller katt i uppladdade filen. Denna prediktion görs av en sammanställning av 4 maskininlärningsmodellers prediktioner. Användaren visas en bild av modellernas prediktion tillsammans med modellernas träffsäkerhet.

3.1 Gränssnitt

Gränssnittet är delen av applikationen som kopplar ihop användaren med funktionaliteten. Detta innebär att det är en viktig del. Om användaren inte förstår gränssnittet så är all funktionalitet värdelös.

Gränssnittet är simpelt upplagt för användarvänlighet. Hela applikationen består av en sida, där media och knappar döljs eller visas beroende på vad som anses nödvändigt för det steg i prediktionsprocessen användaren är på. Detta medför en enkel och trevlig upplevelse att använda applikationen som inte kräver mycket funderande för att kunna utvinna resultat.



Bilden visar applikationen efter en ljudfil har laddats upp och en prediktion har utförts.

3.2 Logik

Vid uppstart av applikationen laddas färdigtränade modeller och data-bearbetnings verktyg in. Finns ej dessa färdiga får man först köra utility filer där modeller tränas och sparas i pickle (.pkl) filer.

När en fil sedan laddas upp till hemsidan sparas denna på klient-sidan för att kunna lyssnas på innan man gör prediktionen. När prediktionsknappen blir tryckt skickas filen via ett POST anrop till servern för att bearbetas och sedan predikteras. Den uppladdade filen bearbetas med samma verktyg som träningsdatan modellerna har tränat på. Modellerna predikterar sedan den bearbetade filen.

När prediktionerna är klara hämtas en bild av det djur som majoritetsprediktionen har valt. Bilden hämtas ur ett bibliotek med 40 bilder av vardera djur, och slumpas vid varje

hämtning i rätt kategori för att ge användaren en mer spännande upplevelse av varje prediktion.

Som tidigare nämnt, så är olika delar av applikationen är synliga vid olika steg. Vid uppstart visas bara en uppladdningsknapp och en prediktionsknapp. När en fil laddas upp visas en ljudspelare där man kan lyssna på filen. Vid en prediktion döljs ljudspelaren och istället visas modellernas prediktioner, samt en bild och även modellernas träffsäkerhet.

3.3 Modellernas resultat

Noggrannheten beräknades genom att använda en train-test-split på 80/20, där 80% av datan användes för träning och 20% för testning. Mätningen gjordes med accuracy_score från scikit-learn, vilket anger hur stor andel av testdata som klassificerats korrekt.

Modell	Noggrannhet (%)
Random Forest	96.67
Support Vector Machine	98.57
eXtreme Gradient Boosting	96.67
Multilayer Perceptron	97.62

4. Avslutning och slutsatser

Projektet har gett mig en djupare förståelse för hur maskininlärning kan tillämpas på ljudklassificering, och specifikt hur man kan särskilja djurläten med hjälp av relativt enkla modeller och verktyg.

Jag har fått en större förståelse för hur man förbereder ljuddata, och vilka steg som krävs för att göra datan användbar, exempelvis extraktion av mel-spektogram, men även normalisering och dimensionsreduktiuon (pca). Något jag under arbetet fick bekräftat var vikten av datakvalitet. Vilket verkligen utmärkte sig när jag med samma modell gick från en test accuracy på 78% till 95%.

Applikationen som utvecklades är relativt enkel men fyller sitt syfte som en tydlig demonstration vad man kan åstadkomma med enkla modeller. Den lyckas prediktera om ett ljudklipp kommer från en katt eller hund med hög precision, och det enkla gränssnittet gör den rolig och användarvänlig även för de utan teknisk bakgrund.

Sammanfattningsvis är jag nöjd med att jag har lyckats följa min tidsplan och färdigställt mitt projekt inom tidsramen. Detta arbete visar att det är fullt möjligt att skapa en fungerande ljudklassificeringsapplikation med enklare verktyg och metoder.

Om jag skulle göra om projektet eller ha mer tid att utveckla det så skulle jag börja med att behandla datan annorlunda och testa nya modeller. Nu gjorde jag datan endimensionell för att fungera med mina lättare modeller, men det finns modeller som klarar av tvådimensionell data och därmed kan hitta fler mönster och prestera ännu bättre.

5. Referenslista

Här redovisas de källor som har angetts i fotnoterna.

Internet

URL 1: https://sv.wikipedia.org/wiki/Ljud
[hämtad 150525]

URL 2: https://medium.com/@karthikmandapaka/handling-audio-data-for-machine-learning-7ba225f183cb
[hämtad 170525]

URL 3: https://medium.com/@lauren-fox/why-should-you-use-flask-framework-for-web-development-f5a7233e17a6
[hämtad 170525]

URL 4: https://en.wikipedia.org/wiki/Random_forest
[hämtad 180525]

URL 5: https://sv.wikipedia.org/wiki/St%C3%B6dvektormaskin
[hämtad 180525]

URL 6: https://en.wikipedia.org/wiki/XGBoost
[hämtad 180525]

URL 7: https://en.wikipedia.org/wiki/Multilayer_perceptron
[hämtad 180525]

URL 8: https://www.kaggle.com/datasets/mmoreaux/audio-cats-and-dogs
[hämtad 190525]

URL 9: https://figshare.com/articles/dataset/Dogs_versus_Cats_Audio_Dataset/20219408?file=36137315
[hämtad 200525]

URL 10: https://en.wikipedia.org/wiki/Mel_scale#/media/File:Mel-Hz_plot.svg
[hämtad 200525]

URL 11: https://scikit-learn.org/stable/auto_examples/preprocessing/plot_scaling_importance.html
[hämtad 200525]