Face-expression-recognition.

Using Deep Learning

En bild som visar tecknad serie, clipart, rita, illustration

Automatiskt genererad beskrivning

Lotta Larsson

NBI/Handelsakademin

Kunskapskontroll 2-deep learning

2024-08-29

# Abstract

This report presents the development of a Streamlit application designed to recognize facial expressions from a live webcam feed or from a photo. Since this is a deep learning project that utilizes a dataset containing images, the model used for this project is a Convolutional Neural Network (CNN) to achieve this task. The goal was to reach 75% accuracy, but the highest accuracy achieved was 69%. The model was trained using both GPU and CPU, which presented challenges throughout the process.

**Skapas automatiskt i Word genom att gå till Referenser > Innehållsförteckning.**

Innehållsförteckning

[1 Inledning 1](#_Toc167972420)

[1.1 Syfte och frågeställningar 1](#_Toc167972421)

[2 Teori 2](#_Toc167972422)

[2.1 Ansiktsuttryckigenkänning 2](#_Toc167972423)

[2.1.1 Djupinlärning och optimering 2](#_Toc167972424)

Bilder………………………………………………………………………………………………………………………………………2, 3

3 Metod…………………………………………………………………………………………………………………………………………4

3.1 Data som använts och kombinering……………………………………………………………………………………4

3.1.1 Förberedelse av data……………………………………………………………………………………………………..4

3.1.2 Modellarkitektur……………………………………………………………………………………………………………5

3.1.3 Träningsprocess och utvärdering…………………………………………………………………………………..5

4 Resultat och diskussion…………………………………………………………………………………………………………..6, 7

5 Slutsatser……………………………………………………………………………………………………………………………………8

6 Teoretiska frågor……………………………………………………………………………………………………………………9,10

7 Sjärvutvärdering…………………………………………………………………………………………………………..11

**Källförteckning…………………………………………………………………………………………………………………………….12**

# Inledning

Artificiell intelligens (AI) är en teknologi som påverkar mycket i vår vardag, och en särskild del av detta är datorseende och djupinlärning. Datorseende handlar om att ge datorn förmågan att tolka och förstå visuell data, såsom bilder och videor. Detta möjliggör att känna igen objekt, ansikten, bildklassificering, segmentering och rörelsedetektering. Dessa teknologier används inom en rad områden, såsom automatisk styrning av robotar eller fordon, samt medicinsk bildanalys för att upptäcka cancerceller och tumörer.

Djupinlärning är en avancerad form av maskininlärning och har revolutionerat datorseendet, särskilt genom användningen av Convolutional Neural Networks (CNN), som är att föredra för att bearbeta visuell data. Möjligheten att tolka visuell data har ökat markant. Användningen av GPU har ytterligare accelererat processen, vilket möjliggör bearbetning av stora mängder data och träning av komplexa modeller på kortare tid. Dock innebar det vissa utmaningar i detta projekt eftersom CPU användes för det slutgiltiga testet, vilket begränsade hastigheten på träningen, men i vissa fall ökade accuracy med någon procent.

## Syfte och frågeställning

Syftet med denna rapport är att utveckla och utvärdera en CNN modell för att känna igen ansiktsuttryck från både livevideo och foto. Detta genom en streamlit applikation. För att uppfylla syftet så kommer följande frågeställningar att besvaras.

1. Hur väl presterar den utvecklade CNN modellen för att känna igen ansiktsuttryck i termer av noggrannhet?
2. Vilka utmaningar uppstår när man tränar och implementerar modellen i både GPU och CPU? Och hur kan dessa övervinnas?
3. Hur effektivt integreras ansiktsuttrycksigenkänning i en Streamlit-applikation som använder både livevideo, och uppladdat foto?

# Teori

## Ansiktsuttryckigenkänning

Detta är en teknik inom datorseende som automatiskt identifierar och analyserar mänskliga ansiktsuttryck från digitala bilder eller live videor. Målet är att klassificera ansiktsuttryck baserat på visuella mönster som ansiktet har(rörelser).  
Några vanliga metoder för att använda detta är Convolutional Neural Network(CNN), Transfer Learning, Haar-cascades, Recurrent Neural Networks(RNNs). Eftersom det är CNN vi använder i denna rapport så är det där fokus ligger.Denna modell är uppbyggd i lager av konvolutionella filter som automatiskt lär sig läsa relevanta punkter från bilden såsom kanter, texturer, former etc som är typiska för olika uttryck. (Bild 1)

En bild som visar diagram, text, linje, Graf

Automatiskt genererad beskrivning

Bild 1 – CNN

### Djupinlärning och optimering

De koncept som har använts i denna modell inkluderar CNN och data augmentation, vilket är en teknik som används för att öka mångfalden i datasetet, särskilt om datan är ojämn i sina klasserna (bild 2). Data augmentation innebär att bilderna roteras, zoomas in, speglas och vinklas för att skapa variation.

Hyperparameteroptimering som har använts är” Adam”. Detta för att hitta balansen i Learning Rate snabbare. Antal lager och neuroner i nätverket. Modellen får inte bli för komplex då risken för ”overfitting” föreligger, eller för enkel, då den inte kan fånga upp viktiga mönster i datan.

Batchstorleken(bild 3)är en annan viktig parameter. En större batchstorlek ger en stabil uppskattning, men kan ta tid. En mindre kan ge mer variation men kan ge mer ”brus” i datan(bild 4).

Antal epoker(Epochs) är hur många gånger träningsdatan körs genom modellen. För många epochs leder till överträning(overfitting), och för få leder till underträning(Bild 4). Med early\_stopping så avslutas träningen automatiskt när valideringsdatan slutar förbättras(Bild 5).

En bild som visar text, skärmbild, Teckensnitt, programvara

Automatiskt genererad beskrivning  
Bild 2 – data augmentation

En bild som visar text, skärmbild, Teckensnitt

Automatiskt genererad beskrivning

Bild 3 – Adam, learning\_rate

En bild som visar text, skärmbild, Teckensnitt, nummer

Automatiskt genererad beskrivning

Bild 4 – batch\_size, epochs

En bild som visar text, skärmbild, Teckensnitt, linje

Automatiskt genererad beskrivning

Bild 5 – early stopping

# Metod

## Data som använts, och kombination

Två dataset har använts eftersom datan var ojämn. Geonom att kombinera dessa dataset ökade modellens generaliseringsförmåga. Båda dataset är hämtade från Kaggle. Dataset 1(Face expression recognition) innehåller 27560 bilder, och dataset 2(fer 2013) innehåller 7178 bilder. Båda dataset har 7 klasser och är i svartvitt. (bild 6.)

Dataseten kombinerades genom att bilderna laddades in och omformades till en gemensam storlek 48x48 pixlar för att säkerställa konsistens vid träningen. Därefter kombinerades de två dataseten till en gemensam tränings och testmatris.

En bild som visar text, Människoansikte, skärmbild, person

Automatiskt genererad beskrivningBild 6 exempel på uttryck

### Förberedelser av data

Bilderna omformas så att de blir normaliserade till ett intervall mellan 0 och 1 genom att dividera med 255. Då konvergerar modellen snabbare under träningen.

De ursprungliga klassettiketterna omvandlas sen till en binär vector(1 eller 0) med one-hot-encoding, vilket är standard i kategorisk klassificering.

Data augmentation används för att får en större variation i träningsdatan. Detta simulerar olika möjliga variationer i ansiktsuttryck för att göra modellen mer stabil.

### Modellarkitektur

### Modellen som använts är en CNN-modell(Convolutional Neural Network)

### Input Layer - där modellen tar bilder i storleken 48 x 48 pixlar som ingång, där bilderna har 1 kanal(gråskala)

* Convolutional Layers – två lager med 64 neuroner, därefter två lager med 128 neuroner, alla lagren har ReLU aktiveringsfunktion och 3x3 filter storlek. Maxpooling -lagret används efter varje par av konvolutionella lager för att minska dimensionerna.
* Dropout – detta används för att förhindra överanpassning.
* Fully Connected Layers – två dolda lager med 256 respektive 128 neuroner , och ReLU aktivering.
* Outputlayer – är ett fullt anslutet lager med 7 neuroner(en för varje känsloklass) med softmax aktivering för att generera sannolikhetsfördelning över känslorna.
* Initialt valda hyperparametrar: Learnig rate, batch size och Adam.

### Träningsprocess och utvärdering

* Träningen genomfördes till en början i Kaggle med Tensorflow/keras. Kaggle valdes för möjligheten att använda GPU som minskar träningstiden avsevärt. En inlärningshastighet(learning rate) på 0.001 användes, och en batch\_size på 64.
* Early stopping används för att förhindra överanpassning. Träningen stoppas om valideringsförlusten inte förbättras under ett visst antal epoker( 3 i denna modellen)
* Modellen utvärderades på testdatan med en accuracy på 69%.
* Confusion matrix användes för att analysera prestandan på datan i varje enskild klass.

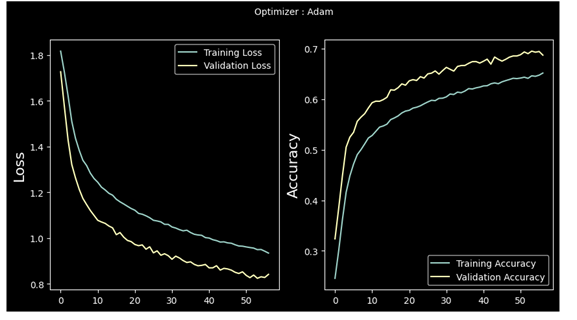
# Resultat och diskussion

Under träningen av modellen observerades en stadig minskning av både tränings och valideringsförlusten, vilket framgår av grafen i figur 1. Efter ca 50 epoker stabiliserade sig valideringsförlusten på ca 0.8, medan träningsförlusten fortsatte att minska något. Detta visar att modellen lärde sig effektivt på träningsdatan samtidigt som den generaliserade väl till valideringsdatan utan att överanpassa sig. När det gäller noggrannhet så ökade både tränings och valideringsnoggrannhet stadigt under träningens gång. Vid slutet av träningen så nådde modellen en noggrannhet på ca 69%, vilket ligger nära den maximala noggrannheten på 70%. Detta tyder på att modellen har en god balans mellan komplexitet och generaliseringsförmåga.

Figur 2 visar en confusion matrix för testdatan, varje rad representerar de faktiska klasserna och varje kolumn de förutsagda klasserna. Modellen var särskilt bra på att känna igen ”happy” med 3252 korrekta klassificeringar. Dock är det svårt med klassificeringen för ”fear” och ”disgust” där det förväxlades med ”sad” och ”angry”. Detta indikerar att modellen är svagare i dessa klasser vilket kan bero på att vissa ansiktsuttryck kan vara lika varandra trots att det är olika klasser. Men också att dataseten är ojämna. Det finns fler bilder i vissa klasser än andra. Ojämnheten har försökts balanseras med hjälp av data augmentation, men även andra sätt övervägs för framtida utveckling.

Figur 3 visar en ansoktuttryckavläsning av ett foto i streamlit applikationen.

Sammanfattningsvis visar resultaten att den tränade modellen uppnådde en stabil valideringsnoggrannhet på 69% och en valideringsförlust på 0.8 efter 50 epoker. Modellen presterade bäst på positiva uttryck, och hade det svårare med negativa uttryck. Detta indikerar på att det finns utrymme för förbättringar. Speciellt genom att fokusera på att balansera träningsdata, men också att finjustera modellen.



Figur 1

En bild som visar text, skärmbild, skärm, programvara

Automatiskt genererad beskrivning

Figur 2.

En bild som visar Människoansikte, skärmbild, kostym, person

Automatiskt genererad beskrivning

Figur 3

# Slutsatser

I rapporten har vi undersökt implementeringen och utvecklingen av en CNN-modell för att känna igen ansiktsuttryck i en streamlit applikation. Här är resultatet av de frågor som ställdes:

1. Hur väl presterar den utvecklade CNN-modellen för att känna igen ansiktsuttryck i noggrannhet?  
   Modellen uppnådde en maximal valideringsnoggrannhet på 67%. Detta indikerar på att modellen är kababel att identifiera ansiktsuttryck i de flesta fall, men det finns utrymme för förbättringar. Särskilt när det gäller de negativa uttrycken såsom ”fear” och ”disgust”. Fokus kommer att ligga på att utöka datasetet och finjustera parametrarna.

2 Vilka utmaningar uppstår när man tränar och implementerar modellen i både GPU och CPU?   
 Hur kan dessa övervinnas?  
 Träningen på GPU är avgörande för att få en effektiv träning av modellen. Det minskar   
 träningstiden avsevärt. När modellen sedan implementerades på en CPU så uppstod en   
 väsentlig prestandaförlust. Dock så höjdes accuracy med ett par % i vissa fall. Men i det  
 stora hela så är det nödvändigt med GPU när man tränar på stor mängd data. För att över-  
 vinna problemen med CPU så finns det olika molntjänster som man kan dra nytta av.

3 Hur effektivt integreras uttrycksigenkänning i en Streamlit-applikation som använder både  
 livevideo och uppladdade foto?  
 Uttrycksigenkänning integrerades med framgång i Streamlit-applikationen. Applikationen   
 presterade tillfredsställande för både livevideo och foto. Det kvarstår dock utmaningar med   
 att optimera analysen när den körs på CPU. Med ytterligare förbättringar så kommer   
 applikationen bli mer användbar i praktiken.

Sammanfattningsvis så har detta arbete lett till en fungerande CNN-modell integrerad i en användarvänlig applikation. Modellen presterar tillfredsställande med utvecklingsmöjlighet.   
Den grundläggande implementationen visar på potentialen att använda den för djupinlärning i realtid, och för vidare förbättring.

# Teoretiska frågor

1.  
Hur är Tensorflow och Keras relaterat?

Tensorflow är ett fristående bibliotek för maskin inlärning och AI. Det kan användas till många olika saker , men är framförallt användbart till djup inlärning.   
 Keras är designat som ett API för att bygga och träna neurala nätverk. I början var det ett fristående bibliotek , men är numera integrerat i tensorflow biblioteket. Detta gör att det är betydligt enklare att använda Tensorflows funktioner

2.  
Vad är en parameter? Vad är en hyperparameter?

Parameter är en variabel som används från datan under träningsprocessen. Parametern använder de underliggande relationerna i datan, och används för prediktion på ny data.   
Hyperparameter är en variabel som ställs in innan träningen för att göra modellen så bra som möjligt, för att förbättra modellens prestanda.   
Hyperparameter ställs in av användaren eller genom hyperparameter optimerare.

3.  
När man skall göra modellval och modellutvärdering så kan man använda ett tränings, validerings och test data. Förklara hur de olika delarna kan användas.

Träning: Den faktiska data som vi använder att träna på. Den innehåller vikter, biases.  
 Modellen ser och lär från denna data.(fit-processen). Här justeraras vikter och bias.

Validering: Valideringen används för att utvärdera en modell. Detta görs flera ggr för att finjustera   
 hyperparametrarna så att modellen presterar så bra som möjligt. Modellen lär sig aldrig   
 från denna data. Validerings data påverkar modellen indirekt för att hjälpa till att hitta de   
 bästa hyperparametrarna.

Test: Detta används bara när modellen är färdigtränad. Detta är den färdiga modellen så att säga.   
 Som används på ny data som inte använts tidigare.

4.  
Din kollega frågar dig vilken typ av neuralt nätverk som är populärt för bildanalys,  
 vad svarar du?

Convolutional Neural Network(CNN) eftersom detta är det mest använda för bildanalys.  
(Fungerar även på data med tal eller audio)

5.  
Förklara översiktligt hur ett ”Convolutional Neural Network” fungerar.   
Det är uppbyggt av olika lager(convolutional, pooling, fully-connected).Conv-lager är det första lagret, som kan följas av flera conv-lager eller pol-lager. Ful-con-lagret är det sista lagret. Varje lager ökar i komplexitet för att identifiera större delar av bilderna. De första lagren identifierar tex linjer och färger, nästa lager tar med ex skuggor osv……

Lagren är uppbyggt av noder där alla noder är kopplade till varandra, och har vikter och ”threshold”som bestämmer om det ska användas eller inte. Dvs om utfallet i en individuell nod är över det specifika ”threshold” värdet så skickas datan till nästa lager. Om det är under värdet så går det inte vidare till nästa.

6.  
Din vän har ett album med 100 olika bilder som innehåller t.ex. tennisbollar och zebror. Hur hade han/hon kunnat klassificera de bilderna trots att han/hon inte har någon mer data att träna en  
modell på?

Genom att använda ”transfer learning”. Det innebär att man använder en förtränad modell som har tränats på ett stort dataset(detta för att spara tid eftersom den förtränade modellen vet mycket om just bilder). Då gör man i regel så att man fryser de första lagren i modellen eftersom de innehåller de generella funktionerna för de flesta bild-klass uppgifterna. När man fryser så innebär det att vikterna inte ändras under träningen. Sen lägger man till nya lager som är anpassade till just detta datasetet. Efter detta tränas de nya anpassade lagren på datan(100 bilderna). Efter utvärderingen kan man finjustera om det behövs för att få bästa ”Accuracy”.

7.  
Vad gör nedanstående kod?   
”model.save(’model\_file.h5)”  
”my\_model = load\_model(’model\_file.h5)”

Den översta raden sparar den modell jag byggt i en specifik h5 fil som gör att jag kan återanvända just den modell jag skapat. Med arkitektur, vikter och allt annat.  
Den undre raden laddar ner den sparade modellen in i programmet igfen så jag kan fortsätta använda den.

8.  
GPU och CPU.  
GPU: ”Grafich Processing Unit” är en processor som är optimerad för att hantera grafikrelaterade   
 uppgifter. Den är mest effektiv när det gäller tolkning och visning av bilder, videor och 3D   
 grafik. GPU har en massiv parallell arkitektur med tusentals kärnor, och kan köra flera trådar   
 samtidigt.  
CPU: ”Central Processing Unit” är en processor för allmänna beräkningsuppgifter, och har   
 vanligtvis ett fåtal kraftfulla kärnor för sekventiell bearbetning. CPU är utmärkt för upp-  
 gifter som kräver hög enkeltrådad prestanda.

# Självutvärdering

1. Utmaningen har varit att träna modellen på ett ställe och att använda den på data på ett annat. Men det har ändå flutit på bra tycker jag i det hänseendet. Det var tufft i början med hyperparametrar som inte fungerade som jag tänkt. Men för min del har det varit ”less is more” Jag valde en enklare modell eftersom den första jag tränade som vi fick i videon inte funkade för mig alls. Den var väldigt instabil hur jag än gjorde , och fick dåliga resultat. Den största utmaningen har nog varit att skapa en förståelse för allt man gjort. Från att tycka det var rena ”grekiskan” till att känna att man faktiskt förstår hur det fungerar.
2. Vilket betyg anser ni att ni skall ha och varför.   
   Det är klart att jag skulle vilja ha ett VG, men det är ett svårt område så jag håller tummarna.
3. Något ni vill lyfta fram till Antonio?  
   Ett stort tack för en skitrolig kurs! Och som vanligt ett otroligt tydligt upplägg som är lätt att följa.
4. Om ni hade gjort om kursen, hade ni gjort något annorlunda?   
   Nä….faktiskt inte. Jag hade gjort på samma sätt.

# Källförteckning

Chat Gpt. (2024). Open AI.

Educative.io.(n.d).*Real-time emotion detection in deep learning.* Hämtad från   
<https://www.educative.io/answers/real-time-emotion-detection-in-deep-learning>

European Data Protection Supervisor.(2021). *TechDispatch: Facial Emotion Recognition.* Hämtad från  
<https://www.edps.europa.eu/system/files/2021-05/21-05-26_techdispatch-facial-emotion-recognition_ref_en.pdf>

Kaggle.(n.d). Hämtad från <https://www.kaggle.com/>

Streamlit.(n.d). Hämtad från <https://streamlit.io/>

TensorFlow.(n.d). *Data Augmentation.* Hämtad från  
https://www.tensorflow.org/tutorials/images/data\_augmentation

Wikipedia.(n.d). *Datorseende.* Hämtad från https://sv.wikipedia.org/wiki/Datorseende