Examensarbete inom Medieteknik
Grundnivå, 15 HP

Structural and Statistical Artefacts in AI-images for Deepfake Detection

KTH, Kungliga Tekniska Högskolan

Författare:
Per Sjöstedt och Albin Speijer

Datum:
25 juni 2026

Handledare:
Jarmo Laaksolahti
Examinator:
Olof Bälter

Sammanfattning

Den snabba utvecklingen av artificiell intelligens har lett till skapandet av mycket realistiska syntetiska bilder, vilket gör det allt svårare att skilja mellan verkligt och AI-genererat innehåll. Denna studie undersöker om strukturella och statistiska artefakter i bilder kan användas för att klassificera AI-genererade bilder med hjälp av maskininlärningsmetoder. Vi utvärderar tre metoder för extraktion av variabler: diskret cosinustransform (DCT), fourierbaserad frekvensanalys samt gradientbaserad lokal riktningsanalys. Dessa variabler används för att träna och jämföra traditionella maskininlärningsmodeller, specifikt Random Forest och eXtreme Gradient Boosting (XGBoost), samt en djupinlärningsmetod baserad på ett konvolutionellt neuralt nätverk av typen ResNet50V2. Samtliga experiment genomförs på en balanserad delmängd av OpenFake-datasetet bestående av 100 000 bilder från flera generativa modeller samt verkliga fotografier. Resultaten visar att dessa tre variabelbaserade metoder kan särskilja verkliga och syntetiska bilder betydligt bättre än slumpen. Bland de enskilda variabelseten uppnår gradientbaserade variabelberäkningar högst prestanda, medan fourierbaserade variabler presterar svagast. En kombination av dessa tre variabeldomäner leder till förbättrad klassificeringsförmåga, där XGBoost uppnår upp till 90% noggrannhet och ett ROC-AUC-värde på 0,96. Samtidigt når det konvolutionella neurala nätverket 79% noggrannhet, vilket är lägre än de bästa resultaten från handkonstruerade variabler. Sammanfattningsvis tyder resultaten på att AI-genererade bilder innehåller detekterbara statistiska och strukturella artefakter som kan utnyttjas för klassificering. Studien belyser vikten av maskininlärningsmetoder för att uppnå robust detektion av syntetiskt bildinnehåll.

Abstract

The rapid development of artificial intelligence has led to the creation of highly realistic synthetic images, making it increasingly difficult to distinguish between real and AI-generated content. This study investigates whether structural and statistical artefacts in images can be used to classify AI-generated images using machine learning methods. We evaluate three feature extraction methods: Discrete Cosine Transform (DCT), fourier-based frequency analysis, and gradient-based local directional analysis. These features are used to train and compare traditional machine learning models, specifically Random Forest and eXtreme Gradient Boosting (XGBoost), as well as a deep learning approach based on a ResNet50V2 convolutional neural network. All experiments are conducted on a balanced subset of the OpenFake dataset, consisting of 100,000 images from multiple generative models as well as real photographs. The results show that these three feature-based methods can distinguish between real and synthetic images significantly better than random chance. Among the individual feature sets, gradient-based feature calculations achieve the highest performance, while fourier-based features perform the weakest. A combination of these three feature domains leads to improved classification performance, with XGBoost achieving up to 90% accuracy and a ROC-AUC value of 0.96. Meanwhile, the convolutional neural network reaches 79% accuracy, which is lower than the best results from engineered features. In conclusion, the findings suggest that AI-generated images contain detectable statistical and structural artefacts that can be leveraged for classification. The study highlights the importance of machine learning methods to achieve robust detection of synthetic image content.

Innehållsförteckning

1 Introduktion **5**

1.1 Bakgrund . . . 5
1.2 Syfte . . . 6
1.3 Problemformulering . . . 6
1.3.1 Hypoteser . . . 6

2 Teori **6**

2.1 Dataset och Filformat . . . 7
2.2 Variabler . . . 7
2.2.1 Diskret Cosinustransform (DCT) . . . 7
2.2.2 Fouriertransform och Frekvensanalys . . . 8
2.2.3 Gradient- och Lokal Riktningsanalys . . . 8
2.3 Traditionella Klassificeringsmetoder . . . 8
2.3.1 Random Forest . . . 8
2.3.2 eXtreme Gradient Boosting (XGBoost) . . . 9
2.4 Faltningsnätverk (CNN) . . . 9
2.4.1 Faltningsoperationen . . . 9
2.4.2 Residualt Lärande och ResNet50v2 . . . 9
2.4.3 Överföringsinlärning . . . 10
2.4.4 Regularisering och Optimering . . . 10
2.4.5 Lager och Sigmoid-aktivering . . . 10
2.4.6 Loss function - Binär Krossentropi . . . 11
2.5 Utvärderingsmått för Klassificering . . . 11
2.5.1 Konfusionsmatris . . . 11
2.5.2 Noggrannhet, Precision, Känslighet och F1-score . . . 11
2.5.3 ROC-AUC . . . 12

3 Metod **13**

3.1 Dataförbehandling . . . 13
3.2 Variabelextraktion . . . 13
3.2.1 Diskret Cosinustransform (DCT) . . . 13
3.2.2 Fouriertransform och Frekvensanalys . . . 14
3.2.3 Gradient- och Lokal Riktningsanalys (MGLH) . . . 14
3.3 Experimentell Strategi och variabelurval . . . 15
3.4 Modellering . . . 15
3.4.1 Random Forest (RF) . . . 15
3.4.2 eXtreme Gradient Boosting (XGBoost) . . . 15

3.4.3 Faltningsnätverk (CNN) . . . 16
3.5 Utvärdering av Modeller . . . 16

4 Resultat **16**
4.1 DCT . . . 16
4.1.1 Hyperparameteroptimering . . . 17
4.1.2 Klassificeringsprestanda . . . 17
4.1.3 Variabelvikt . . . 17
4.2 Fourier . . . 19
4.2.1 Hyperparameteroptimering . . . 19
4.2.2 Klassificeringsprestanda . . . 19
4.2.3 Variabelvikt . . . 19
4.3 Gradient . . . 20
4.3.1 Hyperparameteroptimering . . . 21
4.3.2 Klassificeringsprestanda . . . 21
4.3.3 Variabelvikt . . . 21
4.4 Sammanslagna variabler . . . 22
4.4.1 Hyperparameteroptimering . . . 23
4.4.2 Klassificeringsprestanda . . . 23
4.5 Sammanslagen Modell med variabel reduktion . . . 23
4.5.1 Hyperparameteroptimering . . . 23
4.5.2 Klassificeringsprestanda . . . 24
4.6 Convolutional Neural Network (CNN) . . . 24
4.6.1 Arkitektur och Hyperparametrar . . . 24
4.6.2 Klassificeringsprestanda . . . 25

5 Diskussion **25**
5.1 Resultat och analys . . . 25
5.2 Val av dataset och förbehandling . . . 27
5.3 Felkällor och förbättringar . . . 27

6 Slutsats **28**

7 AI-Användning **29**

A Dataset och Datadistribution **32**

1 Introduktion

Den snabba utvecklingen av generativ artificiell intelligens (AI) har resulterat i modeller som kan producera realistiska bilder av en kvalitet som tidigare ansetts omöjlig. Denna tekniska milstolpe har revolutionerat kreativa industrier, men har samtidigt introducerat stora samhällsutmaningar. AI-genererade bilder, ofta benämnda som deepfakes används i allt större utsträckning för att sprida desinformation, manipulera politiska opinioner och kränka den personliga integriteten [14]. Begreppet *deepfake* användes ursprungligen för att beskriva manipulerade videor där ansikten bytts ut med hjälp av djupinlärning, men syftar i detta arbete bredare på all högst realistisk, syntetisk media och AI-genererade bilder som skapats i syfte att efterlikna verkligheten. I takt med att dessa syntetiska bilder blivit mer realistiska har det blivit allt svårare för det mänskliga ögat att särskilja dessa från verkliga fotografier, därav har behovet av detektionssystem blivit av större vikt [14].

Då hotbilden och bedrägerier från AI-genererad media växer, krävs inte bara robusta detektionssystem, utan även en djupare förståelse för vilka klassificeringsalgoritmer som är bäst lämpade för uppgiften. Beroende på om bilder analyseras i sin helhet eller genom statistiska artefakter kan olika maskininlärningsmodeller uppvisa varierande grad av effektivitet och träffsäkerhet [3, 10, 12]. Denna studie tar denna problematik och syftar på att jämföra prestandan för analys av strukturer och artefakter med moderna modeller för detektion av AI-genererade bilder.

1.1 Bakgrund

Framväxten av generativa bildmodeller växte sig stort i samband med uppkomsten av generativa AI-modeller under 2020-talet. Bildmodeller som GANs (Generative Adverserial Networks) skapades av Ian Goodfellow och medarbetare 2014. Det stora genomslaget för generativ AI skedde däremot i samband med framväxten av LLM:en (Large Language Model) chatGPT av OpenAI som slog igenom 2022. GANs fanns alltså många år innan framväxten av LLM:er men det publika genomslaget för AI skedde kort efter LLM:ers genomslag [14].

Tidigare forskning visar på att det blivit allt svårare att särksilja deepfakes från riktiga bilder för det mänskliga ögat [14]. Samtidigt påstår forskningen att det finns andra metoder för att klassificera dessa bilder med hjälp av maskininlärning. På grund av metoderna som används vid bildgenerering av AI-modeller påstår tidigare forskning att det går att hitta olika sorters strukturella och statistiska skillnader i AI-genererade bilder jämfört med verkliga bilder [3, 10, 12]. På grund av de underliggande matematiska metoder som används vid bildgenerering av AI-modeller, uppstår olika sorters strukturella och statistiska skillnader i AI-genererade bilder [3, 10, 12]. Dessa ofta osynliga, digitala spår från genereringsprocessen benämns i detta sammanhang som *artefakter*.

Tre exempel på dessa skillnader från tidigare forskning är följande:

– Genom att göra diskret cosinustransform (DCT) [12] går det att hitta spår och artefakter av AI-genererade bilder som tydligt skiljer sig mot verkliga bilder givet att bilderna inte är hårt komprimerade.

– AI-modeller misslyckas ofta med att återskapa en naturlig balans av frekvenser i bilder [3]. Efter frekvensanalys uppvisar AI-genererade bilder ofta onaturliga mönster i de högre frekvenserna (det vill säga i bildens finaste detaljer) vilket avviker från den fördelning som förekommer naturligt i äkta fotografier.

– Analys av bilders lokala texturer och hur färger och ljus övergår i varandra, så kallade gradienter [10]. De mikroskopiska riktningarna och mönstren i dessa färgövergångar skiljer sig systematiskt mellan äkta fotografier och syntetiska bilder.

1.2 Syfte

Syftet med denna studie är att identifiera strukturella eller statistiska artefakter och egenskaper hos AI-genererade bilder samt om dessa kan användas för att identifiera AI-genererade bilder med hjälp av maskininlärning. Vi kommer att jämföra resultat från dessa metoder för att ta reda på vilka egenskaper som är mest avgörande för klassificeringen samt jämföra detta med klassificering på hela bilder.

1.3 Problemformulering

Detta examensarbete syftar till att besvara frågan: *Finns det dolda strukturer eller egenskaper som skiljer en AI-genererad bild från en verklig och kan det hjälpa oss att identifiera dessa med hjälp av maskininlärning?*

1.3.1 Hypoteser

Vi kommer att utgå från fyra huvudhypoteser:

- DCT-koefficienter uppvisar statistiskt skilda fördelningar mellan AI-genererade och verkliga bilder.
- Frekvensspektrum i fourierdomänen skiljer sig mellan AI-genererade och verkliga bilder.
- Gradient- och intensitetsövergångar i AI-bilder skiljer sig från verkliga bilder.
- En kombination av ovanstående egenskaper ger bättre klassificering än varje enskild metod.

2 Teori

I detta kapitel presenteras den teoretiska bakgrund och de koncept som ligger till grund för studiens genomförande. Inledningsvis beskrivs det valda datasetet samt filformatens tekniska relevans för maskininlärning. Därefter redogörs för de matematiska metoder som används för att extrahera variabler ur bilderna. Vidare presenteras de maskininlärningsalgoritmer och det faltningsnätverk (CNN) som tillämpas för klassificeringen, och avslutningsvis definieras de utvärderingsmått som används för att mäta modellernas prestanda.

2.1 Dataset och Filformat

Datasetet Openfake är ett omfattande dataset bestående av både AI-genererad och verklig media. Datasetet fokuserar primärt på deepfakes och är publikt tillgängligt just i syfte att underlätta träning och utvärdering av detektionsmodeller. För att säkerställa att tränade modeller uppnår en god generaliseringsförmåga innehåller datasetet bilder från flera olika generativa bildmodeller [11]. Innehållet är till stor del av politiskt känslig natur och bilderna är specifikt utvalda för att representera seriösa hot, såsom desinformation, och manipulation som riskerar att påverka opinionen och den allmänna tilltron till samhällets institutioner.

Filformatet JPEG är ett vanligt förekommande format för lagring av digital media, oftast i komprimerade format. JPEG är en förstörande komprimeringsalgoritm vilket innebär att information förloras i komprimeringen. Genom kvantisering så slänger algoritmen framförallt bort frekvenser som det mänskliga ögat inte kan se [16]. Det är därav passande för exempelvis maskininlärning då man gärna eftersträvar stora mängder data och måste kompromissa mellan datamängd och kvalitet.

2.2 Variabler

I detta avsnitt beskrivs de tre matematiska och statistiska metoder som studeras för att extrahera variabler från bilderna: diskret cosinustransform, fouriertransform och gradientanalys. Gemensamt för dessa metoder är att de syftar till att fånga upp de osynliga frekvensartefakter som generativa nätverk ofta lämnar efter sig.

2.2.1 Diskret Cosinustransform (DCT)

Det finns osynliga artefakter i bilder från generativa AI-bildmodeller som kan analyseras utifrån den statistiska fördelningen av DCT-koefficienter [12]. Denna metod inleds med att dela upp varje bild i icke-överlappande block om 8x8 pixlar. För varje sådant block beräknas de 64 DCT-koefficienterna. Den första koefficienten, DC, beskrivs som ett mått på den genomsnittliga ljusintensiteten för blocket och de resterande 63 koefficienterna, AC, beskrivs som variansen i bilden. Vidare kan AC-koefficienterna hos verkliga fotografier uppskattas tillhöra en Laplace-fördelning. För dessa AC-koefficienter skattas värdet $\beta_{AC} = \frac{\sigma}{\sqrt{2}}$, där σ är standardavvikelsen för en normalfördelning. Dessa 63 värden är de variabler som används för klassificering.

Bilder bör först konverteras till en färgrymd som separerar krominans (färg) och luminans (ljusintensitet) innan DCT-beräkning [16]. Eftersom DCT-koefficienterna inte uppvisar lika stor variation i krominanskanalerna jämfört med luminanskanalen, är det fördelaktigt att isolera den senare. Pixeldatan behöver därefter centreras kring noll innan transformen appliceras; för 8-bitars bilder (med pixelvärden mellan 0 och 255) görs detta genom att subtrahera 128 från varje pixel.

2.2.2 Fouriertransform och Frekvensanalys

Vissa metoder som används i generativa modeller för bilder lyckas inte återskapa frekvensfördelningar likt de för verkliga bilder [3]. Metoder som uppskalning (up-convolution) lyckas inte återskapa frekvensfördelningar, särskilt för högre frekvenser, vilket kan utnyttjas för att särskilja verkliga bilder från AI-genererade bilder med hjälp av dessa frekvensfördelningar.

Fouriertransform kan användas för deepfake detection. Den börjar med en tvådimensionell diskret fouriertransform för att få fram ett tvådimensionellt amplitudspektrum. Därefter utförs en azimuthal integration, förutsatt att bilderna är kvadratiska, där man integrerar för varje koncentrisk cirkel med centrum i mitten av bilden och en radie som varierar från noll till hörnen av bilden. Detta för att skapa en endimensionell vektor av variabler att använda för maskininlärning.

2.2.3 Gradient- och Lokal Riktningsanalys

Ytterliggare en metod att särksilja AI-genererade bilder mot verkliga är med hjälp av gradientberäkning [10]. Denna metod bygger på att först beskära bilderna till 256x256 pixlar och konvertera bilderna till färgrymden nyans, mättnad och intensitet (HSV) då denna färgrymd ger högst noggrannhet. Dessa tre lager (H, S och V) separerades. Därefter beräknas vertikal och horisontell gradient med hjälp av sobeloperatorn för varje lager. Vidare, för dessa två gradienter, skapas kompassmasker. Grundmasken definieras som derivatan av en förskjuten Gaussfunktion. Åtta masker skspas genom att rotera grundmasken i 45 grader (så att de pekar i de åtta kompassriktningarna). De skapar sedan en "Gradient-domain Local Directional Number Pattern"-kod (GLDN-kod) genom faltning av bilden för de olika kompassmaskerna för att sedan hitta det största (i) och minsta (j) värdet av faltningen. GLDN-koden beräknas genom:

$$GLDN(x, y) = 8i_{x,y} + j_{x,y} \tag{2.1}$$

GLDN-koden kan anta 64 olika värden eftersom i och j kan variera mellan 0-7 vilket blir 63 värden och värdet 0.

Slutligen delas bilden in i block, normaliserade histogram för fördelningen av GLDN-kod beräknas och konkateneras till en endimensionell vektor.

2.3 Traditionella Klassificeringsmetoder

2.3.1 Random Forest

Random Forest är en klassificeringsmetod som bygger på att kombinera ett stort antal oberoende beslutsträd. Den grundläggande principen är att varje enskilt träd i ensemblen (skogen) konstrueras utifrån en oberoende och identiskt fördelad slumpvariabel [1].

Denna inbyggda slumpmässighet introduceras primärt på två sätt: för det första genom att varje träd ofta tränas på ett slumpmässigt urval av träningsdatan, och för det andra genom en slumpmässigt

urval av variabler. Vid varje nod i trädbyggnadsprocessen utvärderas endast en slumpmässigt vald delmängd av alla variabler för att hitta den optimala delningen.

Efter att en stor mängd oberoende träd har genererats, hanteras en klassificering för osedd data genom att varje träd får lägga en röst på vilken klass datan tillhör. Modellens slutgiltiga svar bestäms därefter genom ett majoritetsbeslut [1].

2.3.2 eXtreme Gradient Boosting (XGBoost)

XGBoost bygger på principen om stegvis förstärkning. Istället för att träna oberoende träd parallellt, tränas träden efter varandra. Varje nytt träd i följden är designat för att rätta till de fel (residualer) som det föregående trädet gjorde. XGBoost använder en matematisk optimeringsprocess, kallad gradientförstärkning, för att minimera en förlustfunktion i varje steg. Metoden är extremt effektiv på att maximera prediktionsnoggrannheten och finna mönster i strukturerad data [2]. Metoden är dessutom specifikt designad för att hantera högdimensionell och gles data. Genom inbyggda tekniker som delurval av variabler och en optimerad algoritm för träddelningar, kan XGBoost bearbeta tusentals variabler effektivt utan att drabbas av överanpassning [2].

2.4 Faltningsnätverk (CNN)

Faltningsnätverk, mer kända som convolutional neural networks (CNN), har blivit en standard inom maskininlärning för bilder. Till skillnad från traditionella neurala nätverk som tar varje pixel och ser det som en oberoende variabel är CNN utformade för att utnyttja den spatiala korrelationen i bilder [8]. Genom att använda delade vikter kan nätverket identifiera lokala mönster oavsett vart i bilden de förekommer.

2.4.1 Faltningsoperationen

Kärnan i en CNN är faltningslagret. Här dras matriser (så kallade kernels) över bilden för att skapa kartor. De tidigaste lagren lär sig enkla strukturer såsom kanter och hörn. Samtidigt som de djupare lagren kombinerar dessa för att första komplexa strukturer. Denna struktur gör det möjligt för nätverket att lära sig strukturer i datan utan behovet av manuell variabelberäkning. [8].

2.4.2 Residualt Lärande och ResNet50v2

Ett historiskt problem vid träning av djupa neurala nätverk är problemet med försvinnande gradienter, vilket innebär att inlärningen blir svagare ju fler lager den passerar, vilket till slut hindrar modellen från ytterligare förbättring. ResNet (Residual Network) introducerade konceptet med residuala kopplingar, eller "skip connections", som hindrar gradienten från att försvagas mellan lagren [5]. I ekvationen nedan representerar $\mathbf{x}$ indatan till varje enskilt lager. $\mathcal{F}(\mathbf{x})$ är residualen och $\mathcal{H}(\mathbf{x})$ representerar det perfekta slutresultatet enligt vår beräkning. Indatan till nästa lager blir därav $\mathcal{F}(\mathbf{x}) + \mathbf{x}$.

$$\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x} \tag{2.2}$$

ResNet50V2 är en vidareutveckling där man använder "pre-activation". Detta innebär att batch-normalisering och aktiveringsfunktioner (ReLU) appliceras före faltningslagret, vilket görs för att uppnå en stabilare träning och bättre generaliseringsförmåga jämfört med den ursprungliga arkitekturen [6]. ResNet50v2 är en modell som förtränats på datasetet ImageNet som fungerar som basvikter för modellen.

2.4.3 Överföringsinlärning

Genom att tillämpa överförningsinlärning kan en förtränad ResNet50V2-modell användas för att snabba på träningsprocessen. Istället för att uppdatera nätverkets alla parametrar "fryses" basmodellens faltningslager, vilket innebär att endast ett fåtal nya klassificeringslager tränas för den specifika uppgiften. Detta tillvägagångssätt bevarar de generella variabler som modellen lärt sig från ImageNet, samtidigt som behovet av arbetsminne (RAM) minskar drastiskt. Genom att begränsa antalet träningsbara parametrar underlättas träningen avsevärt, vilket möjliggör en snabbare konvergens och minskar risken för överanpassning. [17].

2.4.4 Regularisering och Optimering

För att säkerställa att modellen inte bara memorerar träningsdatan (överanpassning) används flera tekniker:

- Global Average Pooling (GAP): Istället för att platta ut datan, vilket skapar ett stort antal parametrar, beräknar GAP medelvärdet för varje variabelavbildning. Detta minskar risken för överanpassning drastiskt [9].
- Dropout: Under träning deaktiveras slumpmässigt valda neuroner. Detta tvingar nätverket att lära sig centrala representationer istället för att förlita sig på några få specifika noder. För de flesta fall och användningsområden kan en ett valideringsset användas för att bestämma sannolikheten p för att en nod ska deativeras men en bestämd sannolikhet på p = 0.5 ska resultera i nära optimala resultat [15].
- Adam-optimeraren: För uppdatering av nätverkets vikter används Adam, som kombinerar fördelarna med anpassade inlärningshastigheter för att snabbare hitta globala mönster [7].

2.4.5 Lager och Sigmoid-aktivering

Efter att alla variabler har extraherats skickas de vidare till ett fullt anslutet lager, där nätverket väger samman all information. Eftersom uppgiften är binär klassificering (verklig eller AI genererad) används en sigmoid-funktion i det sista lagret. Sigmoid-funktionen omvandlar nätverkets utdata till

ett värde mellan 0 och 1, vilket tolkas som en sannolikhet[4]. I ekvationen nedan motsvarar x utdatan från föregående lager och $\sigma(x)$ modellens beräknade sannolikhet.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

2.4.6 Loss function - Binär Krossentropi

För att nätverket ska kunna lära sig behövs ett mått på hur väl gissningarna stämmer överens med sanna labels. För binära problem är binär korsentropi den matematiskt rekommenderade förlustfunktionen. Funktionen definieras som:

$$L = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \tag{2.4}$$

där N är det totala antalet bilder i en batch, y_i är den sanna etiketten för bild i (exempelvis 1 för AI-genererad och 0 för äkta), och $\hat{y}_i$ är nätverkets beräknade sannolikhet från sigmoid-funktionen. Den straffar nätverket hårt om det är väldigt säkert på fel svar och är gjord för att fungera väl med sigmoid-aktivering för att undvika problem med avtagande gradienter [4].

2.5 Utvärderingsmått för Klassificering

2.5.1 Konfusionsmatris

För att utvärdera effektiviteten för klassificeringen så behöver vi mått på hur väl olika klassificerings metoder presterar. För att förstå följande mått behöver följande notationer förtydligas.

Notation	Beskrivning
TP (True Positive)	Sanna positiva: Korrekt klassificerade positiva fall.
FP (False Positive)	Falska positiva: Negativa fall felaktigt dömda som positiva.
FN (False Negative)	Falska negativa: Positiva fall felaktigt dömda som negativa.
TN (True Negative)	Sanna negativa: Korrekt klassificerade negativa fall.

Tabell 1: Konfusionsmatris notationer förtydligad

Konfusionsmatris är en matris med dessa värden som kan användas för att utvärdera modellens resultat [13]

2.5.2 Noggrannhet, Precision, Känslighet och F1-score

För att utvärdera modellernas prestanda används ett antal etablerade mätvärden som härleds ur konfusionsmatrisen [13].

Noggrannhet (Accuracy) Noggrannhet definieras som förhållandet mellan antalet korrekta klassificeringar och det totala antalet utförda klassificeringar (N). Detta mått ger en bild av modellens totala träffsäkerhet.

$$\text{Noggrannhet} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{TP + TN}{N} \tag{2.5}$$

Precision Precision definieras som förhållandet mellan sanna positiva utfall och samtliga positiva prediktioner.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.6}$$

Känslighet (Recall) Känslighet definieras som förhållandet mellan sanna positiva utfall och det totala antalet faktiska positiva fall. Måttet beskriver modellens förmåga att identifiera samtliga relevanta instanser och kallas även för känslighet.

$$\text{Känslighet} = \frac{TP}{TP + FN} \tag{2.7}$$

F1-score F1-score definieras som medelvärdet av precision och känslighet. Detta mått används för att utvärdera modellens förmåga att både undvika falska alarm och att hitta alla sanna fall i ett enda värde.

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Känslighet}}{\text{Precision} + \text{Känslighet}} \tag{2.8}$$

2.5.3 ROC-AUC

Den mottagaroperativa karakteristikkurvan (receiver operating characteristic, ROC) används för att utvärdera en binär klassificeringsmodell. Kurvan skapas genom att plotta förhållandet mellan sanna positiva (känslighet) och andelen falska positiva (false positive rate, FPR) vid olika tröskelvärden [13].

Känslighet (true positive rate, TPR) har tidigare definierats och mäter andelen faktiska positiva fall som korrekt identifieras.

$$TPR = \frac{TP}{TP + FN} \tag{2.9}$$

Andelen falska positiva (false positive rate, FPR) mäter andelen faktiska negativa fall som felaktigt klassificeras som positiva. Detta motsvarar förhållandet mellan falska positiva (FP) och de totala faktiska negativa fallen (FP+TN) [13].

$$FPR = \frac{FP}{FP + TN} \tag{2.10}$$

För att sammanfatta ROC-kurvans information i ett enda mätvärde beräknas arean under kurvan (area under curve, AUC). Detta värde representerar sannolikheten att modellen rankar en slumpmässigt vald positiv instans högre än en slumpmässigt vald negativ instans. Ett AUC-värde på 1,0 indikerar en perfekt separation, medan 0,5 motsvarar slumpmässiga gissningar. Fördelen med AUC i detta arbete är att det ger ett mått på modellens fundamentala diskrimineringsförmåga som är oberoende av den specifika beslutsgränsen [13].

3 Metod

Arbetet utgick från en kvantitativ och experimentell ansats. En pipeline implementerades för extraktion, behandling och utvärdering av handkonstruerade variabler samt automatiserad inlärning direkt på bilder via ett djupt neuralt nätverk. Experimenten genomfördes på det publika datasetet OpenFake [11]. Ett slumpmässigt urval om totalt 100 000 bilder, exakt fördelat mellan verkliga och syntetiska, valdes ut.

Datasetet delades in i tre isolerade set:

- 64% Träning: För anpassning av klassificeringsmodellerna.
- 16% Validering: För hyperparameteroptimering.
- 20% Testning: För opartisk utvärdering av modellernas generaliseringsförmåga.

där train/test split är 80/20 och träningsdatan i sin tur delades upp 80/20 igen till train/validation. Därav är träningsdatan 64% (80% av 80%) och validering 16% (20% av 80%).

3.1 Dataförbehandling

Samtliga bilder beskars till en fast upplösning om 448x448 pixlar genom en beskärning runt mitten av bilderna. Bilder med en ursprunglig upplösning mindre än detta format exkluderades. Bilderna skalades inte om innan beskärningen. Efter beskärningen sparades samtliga bilder i JPEG-format med kvalitetsnivå 95. Beroende på den efterföljande extraktionen av variabler konverterades bilderna därefter till färgrymden YCbCr eller HSV.

3.2 Variabelextraktion

Tre matematiska representationer extraherades från bilderna i datasetet.

3.2.1 Diskret Cosinustransform (DCT)

Bilderna konverterades till färgrymden YCbCr, luminanskanalen (Y) isolerades med hjälp av OpenCV och Y-lagret centrerades kring noll genom att subtrahera 128. Bilden delades in i icke-överlappande block om 8x8 pixlar. För varje block beräknades 64 DCT-koefficienter med Scipys DCT-funktion för två dimensioner. Den första koefficienten (DC-koefficienten) exkluderades. Därefter beräknades β_{AC} för alla AC-koefficienter genom att använda NumPys funktion för att beräkna standardavvikelse för en normalfördelning samt dividera med $\sqrt{2}$. När dessa 63 β_{AC} värden beräknats så sparades dessa för alla bilder i ett dataset.

3.2.2 Fouriertransform och Frekvensanalys

Bilderna konverterades till färgrymden YCbCr och luminanskanalen (Y) isolerades, detta eftersom vi undersöker strukturella artefakter och inte färgavvikelser. Vidare utfördes tvådimensionell snabb fourier transform (FFT) med hjälp av NumPy. Frekvensspektrat centrerades med inbyggda funktioner från NumPy. Endast magnituden behölls genom att beräkna absolutbeloppen av frekvenserna. Slutligen beräknades logaritmiska värden av magnituderna för att minska dynamiskt omfång, detta gjordes enligt [3] genom att dividera med den första frekvenskomponenten. Eftersom vi inte vet på förhand vilka frekvenser som är av störst storleksordning så logaritmerade vi istället.

För att få en endimensionell vektor så utfördes azimuthal integration. Detta utfördes genom att räkna ut alla avstånd mellan centrum av bilden och ett hörn av bilden där värdena avrundades för att få heltal för index av pixlar. För varje avstånd beräknades summan av alla pixelvärden för en cirkel med denna radie och dividerades med antalet pixlar i cirkeln. Detta ger en vektor med medelvärden för dessa cirklar, avståndet från centrum till ett hörn avrundas till 317 och centrum är även en punkt vilket ger oss 318 variabler.

3.2.3 Gradient- och Lokal Riktningsanalys (MGLH)

Inledningsvis beskärdes bilderna till en storlek av 256x256 pixlar, konverterades till HSV-färgrymden och separerades till tre dessa tre HSV-lager.

För varje lager beräknas vertikal och horisontell gradient med hjälp av sobeloperatorn där vi använder OpenCV i koden. För dessa gradientmatriser skapas åtta kompassmasker enligt metoden av Liu et al. för två Gaussiska skalor, $\sigma = \{1, 2\}$ för att fånga strukurer av olika storleksordning. Grundmasken som roteras för att få åtta kompassmasker beräknas genom:

$$M_\sigma(x, y) = G'_\sigma(x + k, y) * G_\sigma(x, y) \tag{3.1}$$

där σ är den Gaussiska bredden, x och y är positioner. $G_\sigma(x, y)$ beskrivs av:

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} exp(-\frac{x^2 + y^2}{2\sigma^2}) \tag{3.2}$$

* betecknar faltningsoperatorn, G'_σ betecknar derivatan av G_σ och k är förskjutningen i x-led. Värdet på förskjutningen, k, valdes som en fjärdedel av storleken på kompassmasken enligt tidigare forskning.

När dessa kompassmasker skapades användes storleken 5 då det är den minsta storleken som får plats med den största Gaussiska skalan $\sigma = 2$ åt respektive håll.

Varje gradientmatris faltades med de åtta kompassmaskerna och GLDN-koden räknades ut för varje punkt genom att hitta största och minsta värde samt använda dem i formel (2.1).

För att behålla information om vart i bilden information finns delades denna GLDN-kodade matris upp i fyra icke-överlappande block. För varje block skapas normaliserade histogram för GLDN-koden

och dessa fyra histogram konkateneras. Dessa histogram benämns av Liu et al. som Multi GLDN Histogram (MGLH).

Slutligen har vi tre lager i färgrymden, två gradientriktningar, två Gaussiska skalor (σ), fyra block per bild och 64 histogram-bins av möjliga värden för GLDN-kod. Detta ger oss $3 \cdot 2 \cdot 2 \cdot 4 \cdot 64 = 3072$ stycken MGLH variabler.

3.3 Experimentell Strategi och variabelurval

Separata modeller tränades inledningsvis på respektive variabler (DCT, fourier, MGLH). Denna data klassificerades sedan mot Random Forest repsketive XGBoosts algoritmer.

Slutligen skedde en sammanslagning av variabel datan för att testa modellen på samtliga variabler. Detta utfördes först på alla set (DCT + fourier + gradient = 3453).

Slutligen utnyttjades respektive klassificeringsalgoritm för att rangordna variablerna utifrån variabelvikt, och de variabler med högst variation bevarades för den slutgiltiga modellen. Alla DCT-variabler bevarades, de fouriervariabler som motsvarade 75% av variabelvikt bevarades och de gradient-variabler som motsvarade 50% av variabelvikt bevarades. Detta skedde för de enskilda variabel-seten innan sammanslagning. Dessa utvalda variabler sattes sedan ihop till ett komplett variabel-set för alla tre kategorier av variabler för Random Forest.

3.4 Modellering

Tre distinkta modelleringsstrategier implementeras.

3.4.1 Random Forest (RF)

En Random Forest-klassificerare implementerades. Optimalt antal träd fastställdes via en Grid Search med 5-faldig korsvalidering på valideringssetet. Antalet träd testades för 50, 100, 200, 300, 400 och 500 träd.

Separata modeller tränades på de individuella variabelmängderna (DCT, fourier eller gradient). Därefter tränades separata modeller för ett sammanslaget dataset (DCT, fourier och gradient) och för det sammanslagna datasetet med variabelreduktion enligt 3.3.

3.4.2 eXtreme Gradient Boosting (XGBoost)

En XGBoost-klassificerare implementerades även. En liknande Grid Search användes som för random forest användes men i detta fall för att fastställa inlärningshastighet, maximalt djup och antal träd. Optimeringen av dessa parametrar gjordes för inlärningshastigheter på 0.01, 0.1 och 0.2, djup på 3 och 6 samt för 100, 200 och 300 träd.

Separata modeller tränades på de individuella variabelmängderna (DCT, fourier och gradient). Därefter tränades separata modeller för ett sammanslaget dataset (DCT, fourier och gradient) och för det sammanslagna datasetet med variabel reduktion enligt 3.3.

3.4.3 Faltningsnätverk (CNN)

Ett faltningsnätverk (CNN) baserat på ResNet50V2-arkitekturen implementerades via metoden Transfer Learning. Nätverkets förtränade basvikter frystes och ett nytt fullt klassificeringslager tränades direkt på bildpixlarna. Modellen tränades, validerades och testades utifrån den tidigare definierade datauppdelningen.

Nätverkets förtränade ImageNet-vikter frystes och ett nytt klassificeringshuvud lades till bestående av global average pooling, dropout (0.5) och ett dense-lager med sigmoid-aktivering. Bilderna skalades förbehandlades enligt ResNet50V2:s standard. Modellen tränades med Adam-optimeraren (learning rate 0.0001) och binary cross-entropy som förlustfunktion. Early stopping med patience 3 användes för att undvika överanpassning, där de vikter som gav lägst valideringsförlust återställdes. Basmodellens vikter hölls frysta snarare än att fine-tunas (se avsnitt 2.4).

För denna träning användes följande hyperparametrar: indatastorlek 448 × 448 pixlar, batchstorlek 32, en maxgräns på 30 epoker med early-stopping efter tre epoker. En shuffle-buffer på 2000 bilder användes under träning för att variera ordningen mellan epoker.

3.5 Utvärdering av Modeller

Samtliga modeller utvärderades slutgiltigt på den isolerade testdatan. Resultaten analyserades primärt utifrån modellernas förmåga att utföra binär klassificering (verklig / AI-genererad) baserat på följande mätetal:

- Noggranhet: Andelen korrekta klassificeringar över hela testsetet.
- Precision och Känslighet: För utvärdering av distributionen mellan falska positiva och falska negativa utfall.
- Konfusionsmatris: Som visuell representation av fördelningen av sanna och falska förutsägelser.
- ROC-AUC: Utvärderar modellens förmåga att separera klasserna oberoende av specifik prediktionströskel.
- Variabelvikt: Visuell representation av hur stor påverkan olika variabler modellen hade vid klassificering. Denna visualisering utfördes för de tre kategorierna DCT, fourier och gradient individuellt för både Random Forest och XGBoost men inte för de sammansatta variabelmängderna.

4 Resultat

4.1 DCT

För att utvärdera modellernas förmåga att klassificera bilder utifrån frekvensdomänen via diskret cosinustransform (DCT), testades och optimerades både Random Forest och XGBoost, se tabell 2.

Tabell 2: Sammanställning av utvärderingsmått.

Indata	**Modell**	**Noggrannhet**	**ROC-AUC**
DCT	XGB	0.83	0.91
DCT	RF	0.80	0.89
Fourier	XGB	0.74	0.82
Fourier	RF	0.75	0.83
Gradient	XGB	0.86	0.94
Gradient	RF	0.80	0.88
Sammanslagen	XGB	0.90	0.96
Sammanslagen	RF	0.81	0.90
Sammanslagen (Variabel reduktion)	XGB	0.90	0.96
Sammanslagen (Variabel reduktion)	RF	0.82	0.90
Hel bild	CNN	0.79	0.87

4.1.1 Hyperparameteroptimering

Genom Grid Search fastställdes de optimala konfigurationerna för DCT-datasetet

Tabell 3: Sammanställning av optimala hyperparametrar och resultat vid korsvalidering för DCT-datasetet.

Hyperparameter / Mått	**Random Forest**	**XGBoost**
Inlärningshastighet (*learning_rate*)	—	0,2
Maximalt djup (*max_depth*)	—	6
Antal träd (*n_estimators*)	400	300
Korsvalideringsnoggrannhet	**79,06 %**	**82,27 %**

4.1.2 Klassificeringsprestanda

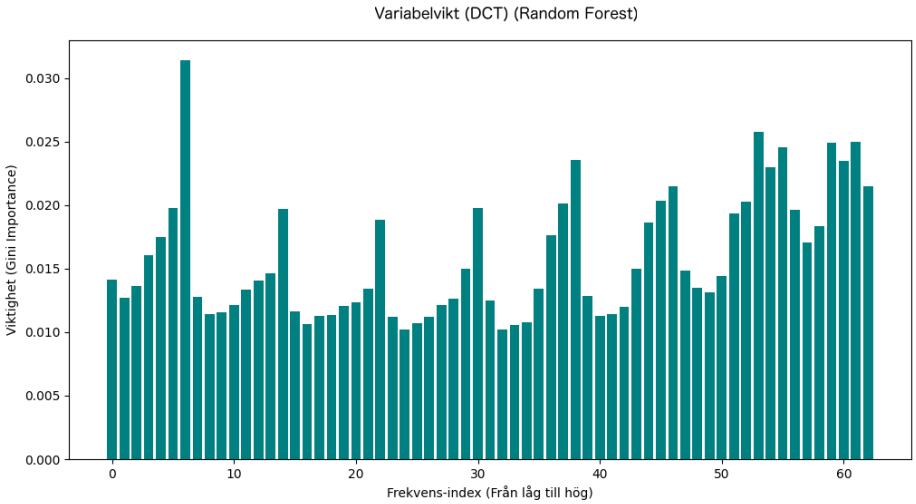
Här utvärderades modellerna på det osedda testsetet (20 000 bilder). En detaljerad sammanställning av utvärderingsmåtten för båda modellerna presenteras sida vid sida i Tabell 4.

4.1.3 Variabelvikt

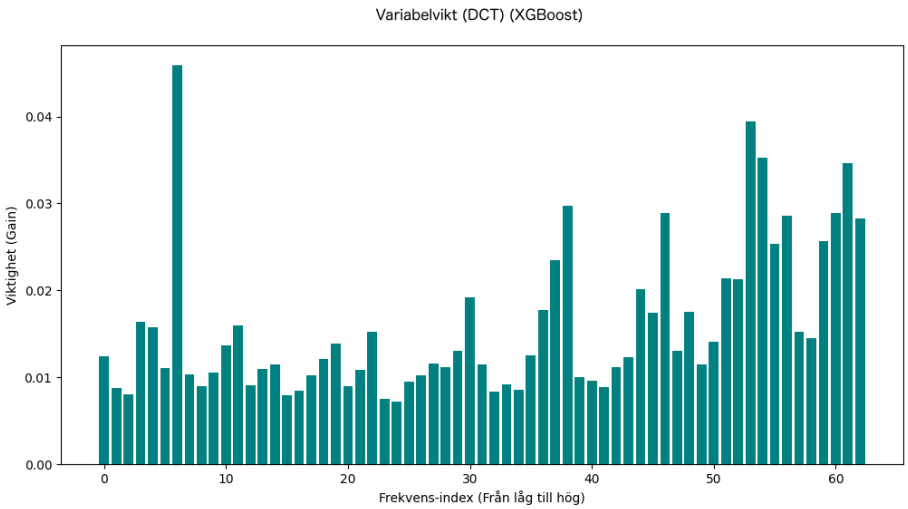
För att visualisera vilka specifika frekvenskomponenter som var mest avgörande för modellernas beslut, extraherades deras respektive "variabelvikt". Dessa presenteras för jämförelse i Figur 1 och 2.

Tabell 4: Jämförelse av klassificeringsrapporter för Random Forest och XGBoost baserat på DCT-särdrag. Testdatan innehöll 10000 bilder från varje klass (foton och syntetiska bilder).

	Random Forest			**XGBoost**		
Klass	**Precision**	**Känslighet**	**F1-score**	**Precision**	**Känslighet**	**F1-score**
Real	0,80	0,80	0,80	0,81	0,86	0,84
Fake	0,80	0,80	0,80	0,85	0,80	0,83
Noggrannhet	**0,80 (80 %)**			**0,83 (83 %)**		
ROC-AUC	**0,8852**			**0,9097**		



Figur 1: Variabelvikt för de blockbaserade frekvensegenskaperna (DCT β_{AC}-koefficienter), utvärderade vid träning av en Random Forest-klassificerare.



Figur 2: Variabelvikt för de blockbaserade frekvensegenskaperna (DCT β_{AC}-koefficienter), utvärderade vid träning av en XGBoost-klassificerare.

4.2 Fourier

För att utvärdera modellernas förmåga att klassificera bilder utifrån frekvensdomänen via fouriertransform testades och optimerades både Random Forest och XGBoost, se tabell 2.

4.2.1 Hyperparameteroptimering

Genom Grid Search fastställdes de optimala konfigurationerna för fourier-datasetet.

Tabell 5: Sammanställning av optimala hyperparametrar och resultat vid korsvalidering för fourier-datasetet.

Hyperparameter / Mått	**Random Forest**	**XGBoost**
Inlärningshastighet (*learning_rate*)	—	0,1
Maximalt djup (*max_depth*)	—	6
Antal träd (*n_estimators*)	700	300
Korsvalideringsnoggrannhet	**74,26 %**	**74,05 %**

4.2.2 Klassificeringsprestanda

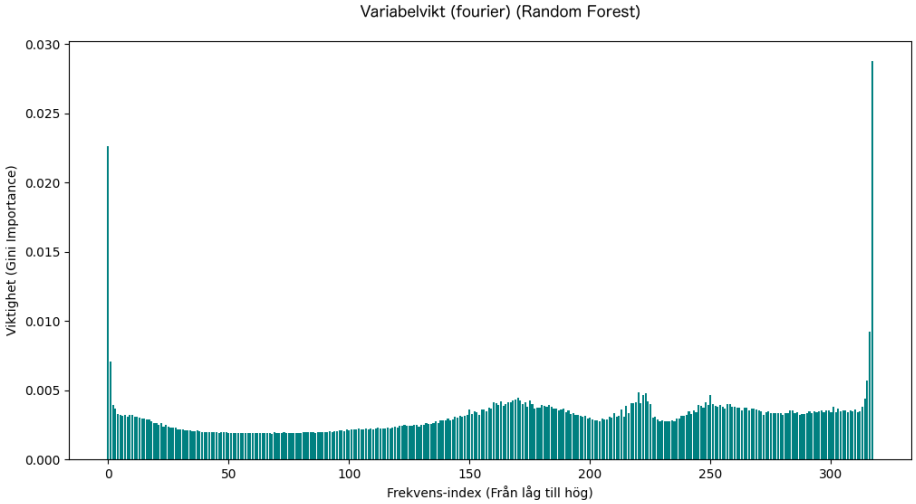
Här utvärderades modellerna på det osedda testsetet (20 000 bilder). En detaljerad sammanställning av utvärderingsmåtten för båda modellerna presenteras sida vid sida i Tabell 6.

Tabell 6: Jämförelse av klassificeringsrapporter för Random Forest och XGBoost baserat på fouriersärdrag. Testdatan innehöll 10000 bilder från varje klass (foton och syntetiska bilder).

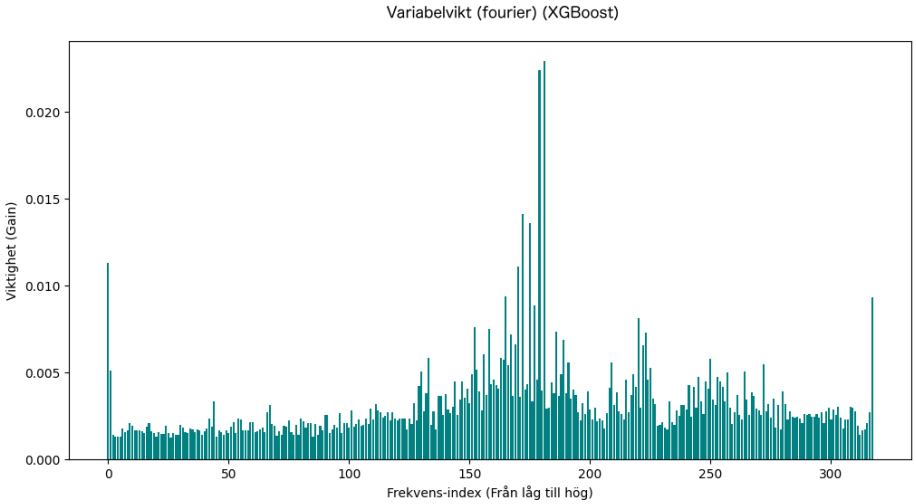
	Random Forest			**XGBoost**		
Klass	**Precision**	**Känslighet**	**F1-score**	**Precision**	**Känslighet**	**F1-score**
Real	0,77	0,72	0,75	0,76	0,72	0,74
Fake	0,74	0,78	0,76	0,73	0,77	0,75
Noggrannhet	**0,75 (75 %)**			**0,74 (74 %)**		
ROC-AUC	**0,8286**			**0,8224**		

4.2.3 Variabelvikt

För att visualisera vilka specifika frekvenskomponenter som var mest avgörande för modellernas beslut, extraherades deras respektive "variabelvikt". Dessa presenteras för jämförelse i Figur 3 och 4.



Figur 3: Variabelvikt för de frekvensbaserade egenskaperna extraherade via fouriertransform (1D radiell profil), utvärderade vid träning av en Random Forest-klassificerare.



Figur 4: Variabelvikt för de frekvensbaserade egenskaperna extraherade via fouriertransform (1D radiell profil), utvärderade vid träning av en XGBoost-klassificerare.

4.3 Gradient

För att utvärdera modellernas förmåga att klassificera bilder utifrån gradient variabler, testades och optimerades både Random Forest och XGBoost, se tabell 2.

4.3.1 Hyperparameteroptimering

Genom Grid Search fastställdes de optimala konfigurationerna för gradient-datasetet.

Tabell 7: Sammanställning av optimala hyperparametrar och resultat vid korsvalidering för Gradient-datasetet.

Hyperparameter / Mått	**Random Forest**	**XGBoost**
Inlärningshastighet (*learning_rate*)	—	0,2
Maximalt djup (*max_depth*)	—	6
Antal träd (*n_estimators*)	500	300
Korsvalideringsnoggrannhet	**79,76 %**	**85,85 %**

4.3.2 Klassificeringsprestanda

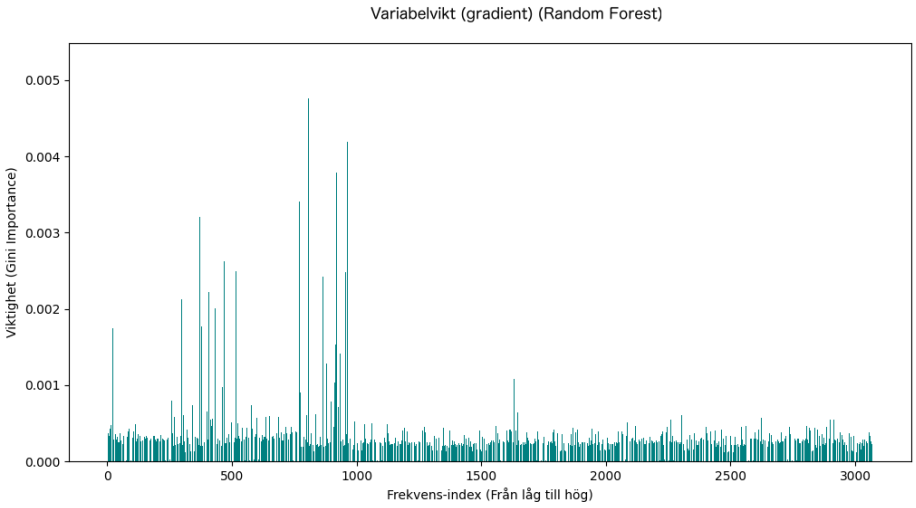
Här utvärderades modellerna på det osedda testsetet (20 000 bilder). En detaljerad sammanställning av utvärderingsmåtten för båda modellerna presenteras sida vid sida i Tabell 8.

Tabell 8: Jämförelse av klassificeringsrapporter för Random Forest och XGBoost baserat på gradient-särdrag. Testdatan innehöll 10000 bilder från varje klass (foton och syntetiska bilder).

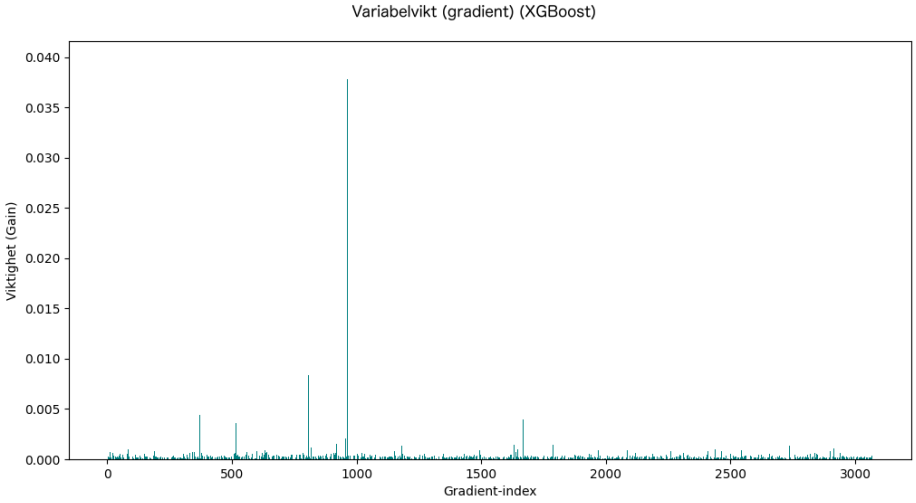
	Random Forest			**XGBoost**		
Klass	**Precision**	**Känslighet**	**F1-score**	**Precision**	**Känslighet**	**F1-score**
Real	0,82	0,78	0,80	0,87	0,86	0,86
Fake	0,79	0,83	0,81	0,86	0,87	0,86
Noggrannhet		**0,80 (80 %)**			**0,86 (86 %)**	
ROC-AUC		**0,8826**			**0,9388**	

4.3.3 Variabelvikt

För att visualisera vilka specifika gradient-komponenter som var mest avgörande för modellernas beslut, extraherades deras respektive "variabelvikt". Dessa presenteras för jämförelse i Figur 5 och 6.



Figur 5: Variabelvikt för de gradientbaserade egenskaperna (MGLH), utvärderade vid träning av en Random Forest-klassificerare.



Figur 6: Variabelvikt för de gradientbaserade egenskaperna (MGLH), utvärderade vid träning av en XGBoost-klassificerare.

4.4 Sammanslagna variabler

För att utvärdera om en kombination av de olika domänerna kunde förbättra prediktionsförmågan, skapades ett sammanslaget dataset med variabler från DCT, fourier och gradienter. Se avsnitt 3.3.

4.4.1 Hyperparameteroptimering

Genom en systematisk Grid Search fastställdes de optimala konfigurationerna för det stora sammanslagna datasetet.

Tabell 9: Optimala hyperparametrar och resultat vid korsvalidering för det fullständiga sammanslagna datasetet.

Hyperparameter	**Random Forest**	**XGBoost**
Inlärningshastighet	—	0,2
Maximalt djup	—	6
Antal träd	500	300
Korsvalideringsnoggrannhet	**80,74 %**	**89,06 %**

4.4.2 Klassificeringsprestanda

Här utvärderades modellerna på det osedda testsetet (20 000 bilder). En detaljerad sammanställning av utvärderingsmåtten för båda modellerna presenteras sida vid sida i Tabell 10.

Tabell 10: Jämförelse av klassificeringsrapporter för Random Forest och XGBoost baserat på det fullständiga sammanslagna datasetet. Testdatan innehöll 10000 bilder från varje klass (foton och syntetiska bilder).

	Random Forest			**XGBoost**		
Klass	**Precision**	**Känslighet**	**F1-score**	**Precision**	**Känslighet**	**F1-score**
Real	0,83	0,79	0,81	0,90	0,90	0,90
Fake	0,80	0,84	0,82	0,90	0,90	0,90
Noggrannhet	**0,81 (81 %)**			**0,90 (90 %)**		
ROC-AUC	**0,8951**			**0,9638**		

4.5 Sammanslagen Modell med variabel reduktion

Utvärdering av sammanslagen modell med variabel reduktion enligt avsnitt 3.3. 100% DCT, 75% fourier 50% gradient (procent av variabelvikt).

4.5.1 Hyperparameteroptimering

Genom Grid Search på det reducerade datasetet fastställdes de optimala hyperparametrarna.

Tabell 11: Optimala hyperparametrar och resultat vid korsvalidering för det reducerade sammanslagna datasetet.

Hyperparameter / Mått	**Random Forest**	**XGBoost**
Inlärningshastighet (*learning_rate*)	—	0,2
Maximalt djup (*max_depth*)	—	6
Antal träd (*n_estimators*)	400	300
Korsvalideringsnoggrannhet	**81,45 %**	**89,19 %**

4.5.2 Klassificeringsprestanda

Vid utvärdering på det osedda testsetet (20 000 bilder) kan vi redovisa jämförelse av de detaljerade mätvärdena presenteras i Tabell 12.

Tabell 12: Jämförelse av klassificeringsrapporter för Random Forest och XGBoost baserat på det reducerade sammanslagna datasetet. Testdatan innehöll 10000 bilder från varje klass (foton och syntetiska bilder).

	Random Forest			**XGBoost**		
Klass	**Precision**	**Känslighet**	**F1-score**	**Precision**	**Känslighet**	**F1-score**
Real	0,84	0,80	0,82	0,90	0,90	0,90
Fake	0,81	0,85	0,83	0,90	0,90	0,90
Noggrannhet		**0,82 (82 %)**			**0,90 (90 %)**	
ROC-AUC		**0,9016**			**0,9647**	

4.6 Convolutional Neural Network (CNN)

CNN tar direkt på den spatiala bilddatan (pixlar i RGB-format). Modellen byggdes på den förtränade arkitekturen ResNet50V2 med tillämpning av Transfer Learning, där basens faltningslager frystes och ett nytt klassificeringshuvud adderades. Se avsnitt 3.4.3.

4.6.1 Arkitektur och Hyperparametrar

Se avsnitt 3.4.3 för motivering av parametrar. De centrala hyperparametrarna och arkitekturvalen sammanfattas i Tabell 13.

Tabell 13: Sammanställning av arkitekturval och hyperparametrar för CNN-modellen (ResNet50V2).

Parameter / Arkitektur	**Värde**
Basmodell	ResNet50V2 (frysta vikter från ImageNet)
Optimerare	Adam (0,0001)
Regularisering	Dropout (0,5)
Batchstorlek	32
Bildstorlek	448 x 448 x 3
Testförlust	**0,4690**

4.6.2 Klassificeringsprestanda

Vid utvärdering på det osedda testsetet (20 000 bilder) uppvisade CNN-modellen följande resultat, tabell: 14.

Tabell 14: Klassificeringsrapport för CNN (ResNet50V2) på spatial bilddata. Testdatan innehöll 10000 bilder från varje klass (foton och syntetiska bilder).

Klass	**Precision**	**Känslighet**	**F1-score**
Real	0,79	0,79	0,79
Fake	0,79	0,79	0,79
Noggrannhet		**0,79 (79 %)**	
ROC-AUC		**0,8713**	

5 Diskussion

5.1 Resultat och analys

Faltningsnätverket (CNN) levererade resultat som kan ses som en baslinje i arbetet. Samtliga mätvärden var jämna, vilket indikerar en balanserad klassificering mellan klasserna utan tydlig partiskhet i prediktionerna.

Enskild variabeldata: Samtliga av våra hypoteser visade sig stämma. Där kan man generellt se bättre prestanda för XGBoost än Random Forest. XGBoost presterade bättre i samtliga fall utöver den enskilda variabeldatan för fouriertransform. Sämsta enskild noggrannhet var fourier med XGBoost men även denna uppnådde 74% noggrannhet. Detta tyder på att även fourierdomänen innehåller diskriminerande information gällande bildens ursprung. Gradient var den enskilt bäst presterande modellen då XGBoost klassificerade. Detta resultat nådde 86% noggrannhet. Detta är även den variabeldata som innehåller flest variabler. XGBoost är, genom sin underliggande förstärkningsarkitektur, särskilt

väl lämpad för att hantera högdimensionell data. Detta förklarar sannolikt varför den gynnades av det stora antalet gradientvariabler.

En intressant observation vid analysen av de enskilda domänerna är den tydliga obalansen i fouriermodellerna. Både Random Forest och XGBoost uppvisar här en högre känslighet för AI-genererade bilder och en lägre för äkta bilder, vilket innebär att de har en tendens att överdetektera syntetiskt material. Detta tyder på att fourierdatan innehåller mycket brus som klassificeraren felaktigt tolkar som syntetiskt material.

Ett bredare mönster som kan urskiljas över flera tester är att Random Forest generellt uppvisar en hög precision men en lägre känslighet för äkta bilder (och vice versa för syntetiska). Modellen tenderar med andra ord att oftare gissa på AI. XGBoost hanterar i de flesta fall denna problematik avsevärt bättre och ger mer balanserade prediktioner. Denna skillnad kan tillskrivas algoritmernas natur: gradient boosting (XGBoost) korrigerar systematiska fel och partiskhet i varje nytt träd, vilket gör den mer motståndskraftig mot den typ av förvrängning som kan uppstå i Random Forests beräkningar.

Det finns dock undantag, intressant nog uppvisar XGBoost, när den tränades på DCT-variabler, det motsatta mönstret, en hög känslighet för äkta bilder men en lägre för AI-genererade. Modellen var därmed mer konservativ och hade en tendens att missa faktiska deepfakes. Samtidigt som Random Forest var helt jämn i sina prediktioner.

Det kan konstateras att denna typ av modell-partiskhet helt eliminerades i det sammanslagna datan samt i CNN-modellen. Genom att kombinera variabler från flera domäner, alternativt förlita sig på CNN-nätverkets helhetsanalys skapas en motståndskraft som neutraliserar de enskilda frekvensernas och algoritmernas individuella partiskhet, vilket resulterar i helt balanserade och tillförlitliga prediktioner vilket kan ses i resultatet för ROC-AUC på 0.96 för den sammanslagna datan med XGBoost respektive 0.87 för CNN.

En noggrannhet på 90% indikerar att modellen korrekt klassificerar majoriteten av bilderna i testdatan. Dock innebär detta samtidigt att cirka 1 av 10 bilder klassificeras fel, vilket i verkliga tillämpningar kan vara problematiskt. Även om modellen uppvisar hög och balanserad precision och känslighet, samt ett ROC-AUC på 0.96, innebär felklassificeringar fortfarande en potentiell risk i praktiska tillämpningar där konsekvenserna av enskilda fel kan vara betydande.

Trots begränsningarna visar resultaten att metoden har potential att användas som ett stödverktyg i praktiska system. Snarare än att fungera som en definitiv klassificerare kan modellen användas för att flagga misstänkt innehåll baserat på modellens säkerhet. Det är viktigt att förtydliga att modellens prediktion i detta sammanhang inte är en absolut sanning, utan snarare ett mått på algoritmens statistiska konfidens i sin egen prediktion. Detta kan exempelvis tillämpas i sociala medier eller nyhetsflöden. Ett sådant system skulle kunna bidra till att öka medvetenheten kring AI-genererat innehåll, utan att kräva perfekt precision.

5.2 Val av dataset och förbehandling

Datasetet i denna studie består av olika typer av diffusionsmodeller. Av 100 000 bilder består samtliga av diffusionsmodeller som ett urval från Openfake. När det kommer till valet av dataset är en central fråga just att finna ett tillräcklig stort dataset på så moderna genereringsmodeller som möjligt. Openfake-datasetet är fyllt med innehåll som kan ses som politiskt och samhällskritiskt, vilket är just den problematik studien avser förbättra. I tabell 15 ser vi 27 olika modeller som vårt dataset innehåller. Detta innebär att våra modeller inte lärt sig att finna mönster hos en specifik genereringsmodell utan att den i viss utsträckning lyckas generalisera bland flera.

Eftersom vi har en exakt 50/50 fördelning mellan AI/verklig så blir fördelningen helt jämn och risken för partiskhet minskar. En uniform förbehandling med samma filformat, storlek och kvalitet minskar risken att modellen lär sig klassificera baserat på filformatet, och säkerställer därmed i högre grad att klassificeringen baseras på verklig kontra AI-genererad bild. Detta tyder på att modellerna inte enbart förlitar sig på kompressionsartefakter, utan även fångar mer strukturella skillnader i bilddatan.

Samtidigt kan det faktum att modellerna uppnår hög noggrannhet trots att datan är något komprimerad ses som ett starkt resultat. Tidigare forskning har visat att kompression tenderar att minska eller maskera de frekvensartefakter som många detektionsmetoder bygger på. Att våra modeller fortfarande uppvisar god klassificeringsförmåga tyder därför på att de identifierade mönstren är relativt robusta, vilket sannolikt även påverkas av att vi behöll den relativt höga JPEG-kvaliteten på 95.

5.3 Felkällor och förbättringar

För CNN-klassificeringen kan valda parametrar påverka modellens prestanda. Basvikterna är tränade på datasetet ImageNet, se avsnitt 2.4, och är därav inte optimerade för detta uppdrag på AI/verklig klassificering. Att låsa upp basvikterna bör därför öka prestandan. Begränsning för denna studie var tid och resurser. Baserat på tillgänglig hårdvara hölls därför basvikterna frysta. Vid finjustering av basvikterna ökar sannolikt prestandan, vilket innebär att CNN-modellens resultat i denna studie kan underskatta dess faktiska potential.

En central avvägning i denna studie rör valet av en uniform förbehandling av bilddatan. Genom att standardisera samtliga bilder till samma upplösning, filformat och komprimeringsnivå minimeras risken att modellerna lär sig ytliga skillnader mellan klasserna. Samtidigt introducerar detta en begränsning i form av minskad extern validitet. I verkliga tillämpningar förekommer bilder med stor variation i kvalitet, upplösning och komprimering, vilket innebär att den homogena datadistributionen i denna studie inte fullt ut speglar verkligheten. Särskilt frekvensbaserade metoder, såsom DCT och fourieranalys, är kända för att vara känsliga för förändringar i komprimering, vilket kan påverka modellernas prestanda i verkliga scenarion. Detta blir därför en central avvägning där båda fallen har sina för- och nackdelar.

Intressant för framtida studier är att analysera och träna modeller per genereringsmodell. Hur prestandan påverkas av att istället för att generalisera över flera generatorer, träna specifikt per modell, är en öppen fråga. Det är oklart om detta skulle förbättra eller försämra prestandan.

Det är också viktigt att ta hänsyn till när studien genomförs. Klassificeringsmodeller kräver kontinuerligt uppdaterad data från de senaste genereringsmodellerna, vilket är tidskrävande att samla in. Samtidigt fortsätter utvecklingen snabbt, och AI-modeller befinner sig i en tillväxtfas utan tydliga historiska motsvarigheter. Detta innebär att datan som används riskerar att snabbt bli föråldrad, vilket skapar en kontinuerlig kapplöpning mellan utvecklingen av genereringsmodeller och detektionsmetoder.

Vidare var valet av hur mycket variabelvikt som behölls vid variabel reduktion av det sammanslagna datasetet utfört baserat på godtycklig analys av variabelvikt för Random Forest av de enskilda dataseten (DCT, fourier eller gradient). Eftersom DCT innehöll 63 variabler och grafen i figur 1 är relativt jämnt fördelad behölls alla variabler. Eftersom fourier innehöll 318 variabler och grafen i figur 3 har få och starka toppar så valdes 75% som andel variabelvikt att behållas. Eftersom Gradient innehöll 3072 variabler och grafen i figur 5 har relativt få och starka toppar så valdes 50% som andel variabelvikt att behållas. Detta är siffror som valdes genom analys av våra resultat, för att se om det förbättrade de sammanslagna modellerna, och andra siffror skulle kunna väljas samt skulle sannolikt ge andra resultat.

Slutligen skiljde sig metoden för fouriertransform och frekvensanalys i avsnitt 3.2.2 något från den som användes av [3]. I metoden så beräknades logaritmerade värden av magnituderna för att minska dynamiskt omfång medan [3] normaliserade magnituderna genom att dividera med den första koefficienten. Då detta skiljde sig från deras metod kan det ha påverkat resultaten och kan vara en anledning till att modellerna som tränades på fouriervariabler gav sämst resultat.

6 Slutsats

För att svara på det huvudsakliga syftet på detta arbete så tyder våra resultat på att det kan finnas strukturella och statistiska artefakter hos AI-genererade bilder samt att dessa kunde användas för att klassificera AI-genererade bilder och verkliga med hjälp av maskininlärning. Samtliga modeller presterade bättre än slumpmässiga gissningar och kunde korrekt klassificera en majoritet av bilderna i vårt dataset.

Samtliga modeller tränade på antingen DCT-, fourier- eller gradientvariabler fick mellan 74-86% noggrannhet vilket visar på att alla tre beräkningsmetoder verkar kunna skilja på AI-genererade bilder och verkliga. För att svara på den fjärde hypotesen så hade den sammaslagna modellen högre noggrannhet än samtliga enskilda modeller.

Sammanfattningsvis så visar resultaten att metoden visar potential för klassificering av deepfakes men att den inte är perfekt. För detta dataset lyckas metoden även prestera bra trots många olika genereringsmodeller vilket visar på en bra generaliseringsförmåga. Vidare utveckling skulle eventuellt behövas för att täcka fler modeller, olika kompression och filformat på bilder samt för att prestera lika bra för nyare genereringsmodeller eftersom de ständigt utvecklas.

7 AI-Användning

I detta arbete har AI använts i vissa delar för att underlätta arbetsflödet och som hjälp för att fortsätta när vi har fastnat. Inledningsvis har AI använts för att konkretisera och problematisera vår grundidé för att göra arbetet mer omfattande. Vidare har AI även använts för att hitta en större bredd av tidigare forskning inom området. Huvudsakligen har AI använts som hjälpmedel för skrivandet av koden till arbetet. Detta har varit lite hjälp att skriva kod i språk eller bibliotek vi inte har så mycket erfarenhet av samt lösa syntaxfel och andra buggar. Slutligen har AI använts som språklig hjälp vid rapportskrivning. Alla resultat och slutsatser i detta arbete är våra egna och alltså inte skapade med AI.

Referenser

[1] Breiman, L. Random forests. *Machine learning 45*, 1 (2001), 5–32. https://link.springer.com/article/10.1023/A:1010933404324#citeas.

[2] Chen, T., and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016). arXiv: 1603.02754 [cs.LG]. Tillgänglig: https://arxiv.org/abs/1603.02754.

[3] Durall, R., Keuper, M., and Keuper, J. Watch your up-convolution: CNN based generative deep neural networks are failing to reproduce spectral distributions, 2020. arXiv:2003.01826 [cs.CV]. Tillgänglig: https://arxiv.org/abs/2003.01826.

[4] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. Tillgänglig: http://www.deeplearningbook.org.

[5] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 770–778. Tillgänglig: https://ieeexplore.ieee.org/document/7780459.

[6] He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2016), pp. 630–645. arXiv: 1603.05027 [cs.CV]. Tillgänglig: https://arxiv.org/abs/1603.05027.

[7] Kingma, D. P., and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)* (San Diego, CA, USA, 2015). arXiv: 1412.6980 [cs.LG]. Tillgänglig: https://arxiv.org/abs/1412.6980.

[8] LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature 521* (2015), 436–444. Tillgänglig: https://doi.org/10.1038/nature14539.

[9] Lin, M., Chen, Q., and Yan, S. Network in network. In *Proceedings of the International Conference on Learning Representations (ICLR)* (2014). arXiv: 1312.4400 [cs.CV]. Tillgänglig: https://arxiv.org/abs/1312.4400.

[10] Liu, Y., Wan, Z., Yin, X., Yue, G., Tan, A., and Zheng, Z. Detection of gan generated image using color gradient representation. *J. Vis. Comun. Image Represent. 95*, C (Sept. 2023). Tillgänglig: https://doi.org/10.1016/j.jvcir.2023.103876.

[11] Livernoche, V., Arodi, A., Musulan, A., Yang, Z., Salvail, A., Caron, G. M., Godbout, J.-F., and Rabbany, R. Openfake: An open dataset and platform toward real-world deepfake detection, 2025. arXiv: 2509.09495 [cs.CV]. Tillgänglig: https://arxiv.org/abs/2509.09495.

[12] Pontorno, O., Guarnera, L., and Battiato, S. On the exploitation of DCT-traces in the generative-AI domain, 2024. arXiv: 2402.02209 [cs.CV]. Tillgänglig: https://arxiv.org/abs/2402.02209.

[13] Powers, D. M. W. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061* (2020). arXiv: 2010.16061 [cs:LG]. Tillgänglig: https://arxiv.org/abs/2010.16061.

[14] Singh, L. H., Charanarur, P., and Chaudhary, N. K. Advancements in detecting deepfakes: Ai algorithms and future prospects a review. *Discovery Internet of Things 5*, 53 (2025). Tillgänglig: https://doi.org/10.1007/s43926-025-00154-0.

[15] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res. 15*, 1 (Jan. 2014), 1929–1958. Tillgänglig: https://dl.acm.org/doi/10.5555/2627435.2670313.

[16] Wallace, G. K. The jpeg still picture compression standard. *Commun. ACM 34*, 4 (Apr. 1991), 30–44.

[17] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems* (2014), vol. 27. arXiv:1411.1792 [cs.LG]. Tillgänglig: https://arxiv.org/abs/1411.1792.

A Dataset och Datadistribution

Den äkta klassen bestod av 50 000 fotografier (50 %). Den syntetiska klassen (50 000 bilder) utgjordes av material från 27 olika generativa AI-modeller, inklusive flera versioner av Stable Diffusion, Flux, Midjourney och DALL-E. Denna mångfald säkerställer att klassificerarna tvingas identifiera universella maskinella artefakter snarare än att memorera signaturer från en enskild algoritm. Den detaljerade fördelningen av datasetet presenteras i Tabell 15.

Tabell 15: Detaljerad fördelning av äkta och AI-genererade bilder i datasetet.

Källa / Bildgenerator	**Antal bilder**	**Andel (%)**
Äkta fotografier (Real)	**50 000**	**50,0 %**
sd-3.5	7 612	7,6 %
flux.1-dev	5 974	6,0 %
sd-2.1	5 222	5,2 %
sdxl-epic-realism	3 217	3,2 %
midjourney-6	2 784	2,8 %
gpt-image-1	2 263	2,3 %
sdxl	2 188	2,2 %
flux.1-schnell	1 998	2,0 %
sd-1.5-dreamshaper	1 969	2,0 %
sdxl-touchofrealism	1 825	1,8 %
dalle-3	1 799	1,8 %
flux-1.1-pro	1 704	1,7 %
sdxl-realvis-v5	1 561	1,6 %
hidream-i1-full	1 500	1,5 %
ideogram-3.0	1 494	1,5 %
sdxl-juggernaut	1 275	1,3 %
sd-1.5-epicdream	1 077	1,1 %
sd-1.5	1 031	1,0 %
flux-mvc5000	885	0,9 %
mystic	821	0,8 %
imagen-4.0	548	0,5 %
grok-2-image-1212	485	0,5 %
flux-amateursnapshotphotos	209	0,2 %
imagen-3.0-002	189	0,2 %
chroma	178	0,2 %
midjourney-7	135	0,1 %
flux-realism	57	0,1 %
Totalt	**100 000**	**100,0 %**

TRITA-EECS-EX-2026:398
Stockholm, Sverige 2026

www.kth.se