



Examensarbete inom datateknik och ekonomi

Grundnivå, 15hp

Deep Learning-baserat system för upptäckt av falska röster

Maryam Bahno

Deep Learning-baserat system för upptäckt av falska röster

Deep Learning-Based System for Voice Spoofing Detection

Maryam Bahno

Examensarbete inom Datateknik och
ekonomi. Grundnivå, 15 hp.

Handledare på KTH: Mikael
Grönkvist

Examinator: Anders Lindström KTH

TRITA-CBH-GRU-2026:140.

Skolan för kemi, bioteknologi och
hälsa

141 52 Huddinge, Sverige

Sammanfattning

Röstbaserad autentisering används i allt större utsträckning inom finansiella system och andra säkerhetskritiska miljöer. Utvecklingen har lett till att systemen blivit mer sårbara för spoofing-attacker, särskilt sådana som bygger på syntetiskt eller manipulerat tal. Samtidigt som teknik för syntetiskt tal utvecklas, utgör attacker ett hot mot tillförlitligheten på automatiska talverifieringssystem (ASV).

Mot denna bakgrund undersöker denna studie hur effektivt olika djupinlärningsbaserade modeller kan användas för att upptäcka spoofing-attacker. Studien genomför en jämförande analys av fyra olika arkitekturer med hjälp av ASVspoofer 2019 Logical Access-datasetet. De modeller som ingår i utvärderingen är ett VGG-inspirerat konvolutionellt neuralt nätverk (CNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) samt bidirektionell GRU (BiGRU). Modellerna valdes utifrån dektekteringsprestanda med hänsyn till beräkningskrav och praktisk användbarhet.

För att bedöma modellernas prestanda används flera etablerade utvärderingsmått, däribland Equal Error Rate (EER), minimum normaliserad tandem Detection Cost Function (t-DCF), noggrannhet, precision, recall, F1-poäng och AUC. Confusion matrix analyseras också för att ge en mer detaljerad bild av modellernas felbeteende.

Resultaten visar att de rekurrenta arkitekturerna presterar bättre än den konvolutionella baslinjemodellen. BiGRU uppvisar bäst resultat, med de lägsta värdena för EER och t-DCF, samtidigt som den uppvisar hög noggrannhet och F1-poäng. Detta indikerar att modellen på ett effektivt sätt kan skilja mellan äkta och manipulerade röstprover och samtidigt upprätthålla en god balans mellan säkerhet och användbarhet. Även den CNN-baserade modellen uppvisar konkurrenskraftiga resultat, medan LSTM-modellen visar tecken på begränsad generaliseringsförmåga trots hög träningsnoggrannhet, vilket tyder på överanpassning.

Sammanfattningsvis visar studien att GRU-baserade arkitekturer, och i synnerhet BiGRU, är lämpad för robust och kostnadseffektiv spoofing-detektion i röstbaserade autentiseringssystem. De erbjuder balans mellan dektekteringsprestandan i relation till beräkningskraven, vilket gör dem effektiva för praktisk implementering.

Nyckelord

ASVspoofer 2019, Anti-Spoofing, BiGRU, Convolutional Neural Networks (CNN), Deep Learning, Equal Error Rate (EER), GRU, Model Evaluation, Recurrent Neural Networks, Speaker Verification, Spoof Detection t-DCF, Voice Biometrics

Abstract

Voice-based authentication is increasingly used in financial systems and other security-critical environments. However, this development has made such systems more vulnerable to spoofing attacks, particularly those based on synthetic or manipulated speech. As speech synthesis technologies continue to advance, these attacks pose a significant threat to the reliability of automatic speaker verification (ASV) systems.

Against this background, this study investigates how effectively different deep learning–based models can be used to detect spoofing attacks. A comparative analysis is conducted using the ASVspoof 2019-LA-dataset. The evaluated models include a VGG-inspired Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Bidirectional GRU (BiGRU). The models were selected not only based on detection performance, but also with consideration of computational requirements and practical applicability.

To assess model performance, several established evaluation metrics are used, including Equal Error Rate (EER), minimum normalized tandem Detection Cost Function (t-DCF), accuracy, precision, recall, F1-score, and Area Under the Curve (AUC). Confusion matrix analysis is also performed to provide a more detailed understanding of the models' error behavior.

Results show that recurrent architectures perform better than the convolutional baseline model. Particularly the BiGRU, achieved the strongest performance across evaluation metrics. These findings indicate that the model can effectively distinguish between genuine and spoofed speech samples while maintaining a good balance between security and usability. CNN-based models also demonstrate competitive performance, whereas the LSTM model shows signs of limited generalization despite high training accuracy, suggesting overfitting.

In conclusion, the study demonstrates that GRU-based architectures, and BiGRU in particular, are well suited for robust and cost-effective spoofing detection in voice-based authentication systems, offering a favorable trade-off between high detection performance and relatively low computational requirements.

Keywords

ASVspoof 2019, Anti-Spoofing, BiGRU, Convolutional Neural Networks (CNN), Deep Learning, Equal Error Rate (EER), GRU, Model Evaluation, Recurrent Neural Networks, Speaker Verification, Spoof Detection, t-DCF, Voice Biometrics

Förord

Denna rapport är resultatet av ett examensarbete inom datateknik och ekonomi med fokus på detektion av röstspoofing.

Jag vill rikta ett varmt tack till min handledare Mikael Grönkvist för värdefull vägledning och stöd under arbetets gång.

Ordlista över förkortningar

- **ASVspoof** – Automatic Speaker Verification Spoofing (utmaning/dataset)
- **BiGRU** – Bidirectional Gated Recurrent Unit
- **CNN** – Convolutional Neural Network
- **EER** – Equal Error Rate
- **GRU** – Gated Recurrent Unit
- **LSTM** – Long Short-Term Memory
- **min t-DCF** – Minimum Tandem Detection Cost Function
- **RNN** – Recurrent Neural Network
- **VGGNet** – Visual Geometry Group Network
- **TTS** – Text-To-Speech
- **VC** – Voice Conversion
- **GAN** – Generative Adversarial Network
- **MGD** – Modified Group Delay

1. Inledning	1
1.1 Bakgrund.....	1
1.2 Problemformulering.....	1
1.3 Syfte och målsättning.....	2
1.4 Avgränsningar.....	2
1.5 Författarens bidrag till examensarbetet.....	2
2. Bakgrund och teori	3
2.1 Vad i ljudet kan avslöja röstspoofing.....	3
2.2 Utveckling av röstspoofing-tekniker.....	4
2.2.1 Typer av röstspoofing-attacker.....	4
2.2.2 Utmaningar inom spoofing-detektion.....	5
2.3 Traditionella metoder för att upptäcka röstspoofing.....	6
2.4 Deep learning-baserad spoofingdetektion.....	7
2.4.1 Convolutional Neural Networks (CNN).....	7
2.4.2 Recurrent Neural Networks (RNN) och Long Short-term Memory (LSTM).....	8
2.4.3 Transformer-baserade-modeller.....	8
2.4.4 End-to-end-inläring och självövervakande modeller.....	8
2.5 Dataset och utvärderingsmått.....	9
2.5.1 Dataset.....	9
2.5.1.1 Standarddataset som används som jämförelse.....	9
2.5.1.2 Dataset för syntetisk tal (TTS och GAN).....	10
2.5.1.3 Dataset som liknar verkliga miljöer.....	11
2.5.1.4 Dataset för olika språk.....	11
2.5.2 Utvärderingsmått.....	12
2.5.2.1 Equal Error Rate (EER).....	12
2.5.2.2 Tandem Detection Cost Function (t-DCF).....	13
2.5.2.3 Klassificeringsmått.....	13
2.6 Begränsningar hos befintliga metoder och framtida riktningar.....	14
2.6.1 Identifierade begränsningar.....	14
2.6.2 Framtida forskningsinriktningar.....	14
3. Metod	17
3.1 Val av dataset och modeller.....	17
3.2 Förbehandling av data.....	17
3.2.1 Datainsamling och tolkning av protokoll.....	18
3.2.2 Sortering och balansering av klasser.....	18
3.2.3 Extraktion av egenskaper med MFCC.....	18
3.3 Djup-inlärningsmodeller.....	19
3.3.1 Konvolutionella neurala nätverk (CNN).....	19
3.3.1.1 VGG-inspirerad CNN.....	20
3.3.2 Rekurrenta neurala nätverk (RNN).....	21
3.3.2.1 LSTM.....	21
3.3.2.2 GRU.....	23
3.3.2.3 BiGRU.....	24
3.4 Implementationsmiljö.....	25

3.5 Modellens träningsflöde	25
3.5.1 Förbereda och läsa in data	26
3.5.2 Uppdelning av dataset	26
3.5.3 Modellens kompilering	27
3.5.4 Förstärkning av data	27
3.5.5 Egenbyggda callbacks för utvärdering anpassad till området	28
3.5.6 Early Stopping och träning av modellen	30
3.5.7 Slutlig utvärdering och tröskelvärdesbestämning	30
3.5.8 Visualisering och analys.....	31
3.5.9 Export av modellen	32
3.5.10 Rapportering	32
4. Resultat.....	33
4.1 Sammanfattande testresultat	33
4.2 VGG-inspirerad CNN	34
4.3 LSTM.....	36
4.4 GRU	37
4.5 BiGRU	39
4.6 Fördelning av klassificeringsfel	40
5. Diskussion	43
5.1 Jämförelse mellan EER och min-tDCF.....	43
5.2 Analys av klassificeringsmått.....	43
5.3 Insikter om confusion matrix	44
5.4 AUC-, DET- och poängfördelningar	44
5.5 Sammanfattning av iakttagelserna.....	44
5.6 Modellernas beräkningskrav och praktisk användbarhet.....	45
5.7 Relatering till tidigare forskning	46
5.8 Begränsningar och val av dataset.....	46
5.9 Påverkan på samhällsnivå	47
5.9.1 Kostnadsbesparingar och riskreducering	47
5.9.2 Marknadsspecifika tillämpningar samt fördelar.....	47
5.9.3 Bredare ekonomiska och samhälleliga konsekvenser	48
5.9.4 Sammanfattning av påverkan.....	49
6. Slutsats och fortsatt arbete	51
6.1 Slutsats.....	51
6.2 Framtida arbete	51
Referenser	53
Bilagor.....	57
Bilaga A -Kodexempel 3.5.....	57
Bilaga B -Kodexempel 3.6	58
Bilaga C -Repository	59

1. Inledning

Detta kapitel beskriver hur djupinlärning, som är en gren av AI, används inom röstbaserad autentisering och detektion av röstspoofing. Sedan beskrivs problem med röstautentisering och de säkerhetsrisker som är kopplade till utvecklingen av syntetiskt och manipulerat tal. Till sist presenteras arbetets syfte, målsättningar och avgränsningar.

1.1 Bakgrund

Djupinlärning är en gren av AI som använder neurala nätverk för att förstå och analysera data. Djupinlärning har möjliggjort för datorer att självständigt identifiera relevanta mönster i data utan forskares manuella val [1]. Innan djupinlärning var experter tvungna att identifiera de viktiga egenskaperna i deras material, såsom bilddata eller vilka ljudsignaler som används för att identifiera och analysera en röst. Detta tog lång tid och man kunde då missa viktiga detaljer [2]. Med djupa inlärningsmodeller kan datorer automatiskt hitta och analysera dessa egenskaper, vilket gör tekniken både snabbare och mer exakt.

I dag används röstbaserad autentisering i flera digitala tjänster, exempelvis för identifiering och inloggning. Tekniken upplevs som smidig och användarvänlig. Samtidigt har utvecklingen av syntetiskt och manipulerat tal gjort röstsysten mer sårbar. Med moderna metoder, såsom text-to-speech (TTS) och voice conversion (VC), kan röster manipuleras så de låter realistiska. Detta innebär att samma tekniska framsteg som förbättrar röstsysten också kan utnyttjas för att försöka lura dem. Utvecklingen har gjort röstspoofing till ett växande problem, särskilt i system där rösten används som säkerhetsfaktor.

Inom den ekonomiska sektorn har djupinlärning blivit ett verktyg för att upptäcka bedrägerier, bedöma risker och verifiera röster genom att analysera stora mängder data för att hitta avvikelser och mönster [3]. Djupinlärning används därför både för att skapa och att upptäcka manipulerade röster. Syntetiskt tal som skapas med djupinlärning innehåller ofta små skillnader i ljudets frekvens och tidsmönster, som äldre metoder har svårt att upptäcka [4][5].

1.2 Problemformulering

Trots tekniska framsteg inom röstbaserade autentiseringssystem är fortfarande känsliga för manipulation. Med hjälp av tekniker som text-to-speech (TTS) och voice conversion (VC) kan angripare skapa röster som felaktigt identifieras som individer [4]. Dessa röster kan användas för att lura system.

Attackerna innebär konsekvenser i miljöer där rösten används för att ge tillgång till tjänster eller konton. Det kan handla om identitetsstöld, ekonomiska bedrägerier eller spridning av falsk information [4][7]. Även avancerade system har visat sig kunna påverkas av spoofing-attacker [6]. Detta kan i sin tur leda till ekonomiska förluster och minskat förtroende för röstbaserade tjänster [3]. Det finns därför ett tydligt behov av metoder som på ett tillförlitligt sätt kan skilja mellan äkta och manipulerat tal.

1.3 Syfte och målsättning

Trots framsteg inom röstspoofing-detektering så finns det fortfarande flera utmaningar, såsom begränsad tillgång till varierade dataset, svårigheter med att utvärdera modeller tvärvetenskapligt, samt snabb utveckling av spoofing-attacker. Målet med detta arbete är att undersöka hur väl olika djupinlärningsmodeller kan skilja mellan äkta och manipulerat tal baserat på extraherade MFCC-ljudegenskaper. Modellernas prestanda kommer att utvärderas med hjälp av standardiserade dataset och utvärderingsmått.

De primära målen med detta examensarbete inkluderar:

- **Robust detektering:** undersöka och utvärdera förmågan hos flera djupinlärningsbaserade arkitekturer om de på ett trovärdigt sätt kan skilja mellan äkta och manipulerat ljud och om de kan anpassa sig till olika brusnivåer, ljudkodning och variationer i ljudöverföring.
- **Djupinlärningsintegration:** använda olika typer av moderna neurala nätverk för att automatiskt analysera MFCC-egenskaper extraherade från ljudsignaler.
- **Omfattande utvärdering:** samla och förbereda ett standardiserat dataset för att implementera och testa modellerna samt analysera deras prestanda med etablerade utvärderingsmått.
- **Jämförande analys:** utvärdera och jämföra prestandan hos de modifierade djupinlärningsmodellerna för detektion av röstspoofing, i syfte att identifiera deras styrkor och svagheter med avseende på prestanda, robusthet och beräkningskrav.

1.4 Avgränsningar

Denna studie baseras på ett standardiserat dataset för träning och test av algoritmer som identifierar manipulerade röster. Fokus ligger på att jämföra olika djupinlärningsmodellernas förmåga att skilja mellan äkta och manipulerat tal.

Arbetet omfattar inte utveckling av nya spoofing-attacker eller skapandet av nya dataset. Studien avgränsas även till relativt kompakta och mindre resurskrävande modeller. Större och mer beräkningsintensiva modeller har inte inkluderats, även om sådana modeller i vissa fall kan uppnå högre prestanda. Den teoretiska bakgrunden presenteras i kapitel 2 och den praktiska implementationen beskrivs i kapitel 3.

1.5 Författarens bidrag till examensarbetet

Detta examensarbete inom datateknik vid Kungliga Tekniska Högskolan har genomförts av Maryam Bahno. Samtliga delar av arbetet, inklusive litteraturstudie, metodutveckling, implementation, experiment, analys samt rapportskrivning, har utförts av författaren.

2. Bakgrund och teori

Detta kapitel behandlar tidigare forskning om röstspoofing, hur spoofing-attacker kan motverkas och vilka djupinlärningsmetoder som används för detektion. Det beskriver även vanliga dataset och tar upp utvärderingsmetoder samt begränsningar i dagens lösningar.

Eftersom röstbaserad autentisering används inom banker och kundtjänster är det viktigt att skydda systemen. Genom att analysera tidigare forskning och tekniska framsteg framkommer ett behov av mer robusta lösningar för att skydda röstbaserade autentiseringssystem mot spoofingattacker. Kapitlet avslutas med en översikt av framsteg och utmaningar, vilket visar varför denna studie är viktig.

2.1 Vad i ljudet kan avslöja röstspoofing

Röstspoofing är ett system för att lura röstsystem genom att använda en falsk eller manipulerad röst som låter som efterliknar en annan persons röst [8][9]. Det kan till exempel vara en inspelning som spelas upp, eller en röst som skapats med hjälp av AI. Eftersom dagens teknik kan skapa realistiska röster har det blivit svårare att identifiera skillnaden mellan äkta och falskt tal.

För att förstå hur röstspoofing upptäcks behöver skillnader mellan ett naturligt mänskligt tal från ett syntetiskt eller manipulerat tal identifieras. När en människa talar uppstår hela tiden små variationer i rösten. Tonhöjd, tempo och energi förändras ständigt, ofta omedvetet. Dessa variationer beror på biologiska faktorer som andning, muskelrörelser och hur vi formar orden med mun och stämband. Resultatet blir ett levande och komplext ljudmönster som aldrig är exakt identiskt från gång till gång. Syntetiskt tal, som skapas med text-to-speech (TTS) eller voice conversion (VC), försöker efterlikna detta mänskliga beteende med hjälp av neurala nätverk och statistiska modeller. Moderna TTS- och röstkonverteringssystem kan generera mycket naturtroget tal, vilket gör det svårt för människor att höra skillnaden mellan äkta och syntetiskt tal [8]. Genereringen bygger på matematiska approximationer, och därför uppstår ofta små, systematiska avvikelser jämfört med naturligt tal. Dessa är sällan hörbara, men kan bli tydliga vid teknisk analys.

Analyserar man ljudet i ett spektrogram för att identifiera talets spektrala struktur ser man hur energin fördelas över frekvenser och tid. Då framträder ofta komplexa och oregelbundna mönster i naturligt tal. Syntetiskt tal kan däremot uppvisa jämnare strukturer eller sakna vissa finare detaljer, särskilt i högre frekvensområden. Flera studier inom anti-spoofing visar att sådana spektrala artefakter är vanliga i TTS- och VC-system [8][10].

För att beskriva dessa skillnader har traditionella metoder använt handkonstruerade egenskaper som Mel-Frequency Cepstral Coefficients (MFCC) och Constant-Q Cepstral Coefficients (CQCC) [11][12]. Dessa funktioner sammanfattar talets frekvensinnehåll och har länge varit centrala inom taligenkänning och spoofing-detektion. De kan fånga vissa avvikelser effektivt, men har samtidigt begränsningar när attackerna blir mer avancerade.

Tidsmässig dynamik är en relevant faktor vid identifiering av röstspoofing. Tal är en sekvens som förändras över tid. I naturligt tal varierar rytm, betoning och tempo på ett naturligt och kontinuerligt sätt. I syntetiskt tal kan övergångar mellan ljud ibland bli för jämna eller sakna den naturliga mikrovariation som uppstår vid mänsklig artikulation. Modeller som Long Short-Term Memory (LSTM) är särskilt utvecklade för att analysera sådana tidsberoenden [13] och har därför använts inom spoofing-detektion för att identifiera subtila tidsmässiga oregelbundenheter [8].

En tredje dimension är fasinformationen i signalen. Fas beskriver hur olika frekvenskomponenter är förskjutna i tid i förhållande till varandra. I naturligt tal uppstår komplexa fasrelationer, medan syntetiskt tal ibland kan uppvisa mer förenklade eller artificiella fasstrukturer. Tidigare forskning har visat att fasbaserade egenskaper, såsom Modified Group Delay (MGD), kan användas för att identifiera syntetiskt tal [10]. Metoderna har brister men, visar att viktig information finns även bortom den traditionella amplitudanalysen.

Självövervakade modeller, såsom wav2vec 2.0 [14], har skapat möjligheter att arbeta direkt med representationer som lärs från rå ljuddata. Dessa modeller kan fånga mer komplexa statistiska mönster än traditionella egenskaper och därmed upptäcka strukturer som annars skulle vara svåra att identifiera.

Det som gör spoofing-detektion utmanande är att skillnaderna ofta är subtila. För människor låter moderna deepfake-röster ofta helt naturliga. Däremot kan matematiska representationer av signalen avslöja små oregelbundenheter i frekvens, tidsdynamik och fasstruktur. Traditionella och djupinlärningsbaserade metoder försöker utnyttja dessa akustiska indikationer för att skilja mellan äkta och manipulerat tal.

Sammanfattningsvis kan röstspoofing ge upphov till systematiska men svårupptäckta avvikelser i spektrum, tidsmönster och fasinformation. Avvikelseerna är sällan hörbara för människor, men kan analyseras genom signalbehandling och maskininlärning. Dessa akustiska skillnader utgör grunden för moderna system för spoofing-detektion.

2.2 Utveckling av röstspoofing-tekniker

2.2.1 Typer av röstspoofing-attacker

En av de vanligaste typerna av röstspoofing är så kallade replay-attacker. Dessa innebär att en angripare spelar upp en inspelning av en verklig persons röst för att lura ett automatiskt röstverifieringssystem (ASV). I ASV-spoof 2019 utmaningen genomförde Yamagishi et al. en detaljerad studie med dessa testade replay-attacker genom att undersöka olika inspelningssituationer för att se hur dessa påverkade attacker i olika miljöer [9].

Forskarna använde standardiserade testmetoder och visade att även de mest avancerade ASV-systemen kan få sämre resultat, speciellt när det inspelade ljudet spelades upp i en miljö med ett annat ljudförhållande än det som användes vid träning. Skillnader i tränings-

och testförhållandena kan påverka systemets prestanda och leda till ökade fel, vilket belyser utmaningen att utveckla intelligenta system som fungerar i varierande miljöer [9].

En annan typ av röst-spoofing-attacker är text-till-tal (TTS), som bygger på en syntetisk röst som skapas från text. De skiljer sig från replay-attacker genom att när en inspelning återanvänds så skapar TTS-systemet helt nya ljudklipp med hjälp av avancerade modeller.

För att bekämpa TTS-attacker utvecklade Tak et al. [15] en avancerad metod som kallas ett spektro-temporalt grafuppmarksamhetsnätverk för att upptäcka dessa attacker. Metoden fokuserar på att analysera små avvikelser i ljudets struktur, såsom hur ljudet ser ut över tid och vilka frekvenser det innehåller. Dessa avvikelser finns ofta i syntetiska röster men de är så subtila att vi människor inte kan uppfatta dem med hörseln. I deras experiment, genom att analysera dessa detaljer, visade modellen en förbättring i detektionsnoggrannhet med en betydande minskning av EER, vilket betyder att systemet gjorde färre misstag än traditionella funktionsbaserade tekniker, som ofta missar viktiga ljudartefakter.

Dessutom finns det en annan typ av attacker som kallas Voice Conversion (VC) som ökar avanceringen av röstspoofing. Den ändrar en persons röst för att låta som någon annan. Li et al. presenterade en genomgång av olika VC-tekniker och beskrev att även om det omvandlade talet kunde låta mycket naturligt, så fanns det små spektrala oregelbundenheter i ljudets frekvensmönster [8]. De betonade också att med en detektionsmetod som fokuserar på dessa skillnader lyckades de minska EER märkbart i testerna. Detta visar hur spektralanalys kan vara effektiv vid identifiering av manipulerade röster.

Den mest avancerade typen av spoofingattacker kallas deepfake-röster. Dessa attacker skapar syntetiska röster med hjälp av modern AI såsom generative adversarial networks (GANs) och transformermodeller [16][17]. Dessa röster låter nästan helt verkliga och är svåra att skilja från riktiga. Lee et al. undersökte kvalitén på detektionssystem för att upptäcka dessa attacker och visade att traditionella detekteringsmetoder hade stora svårigheter [6]. Deras experiment visade en ökning av EER när systemen testades mot dessa deepfake-röster. Metoder behöver alltså avanceras i relation till utvecklingarna av röstspoofingsystem.

Eftersom röstautentisering har stor betydelse inom områden som bank, finans och telekommunikation, investerar många branscher nu i ny teknik för att kunna upptäcka spoofing-attacker och på så sätt minimera ekonomiska förluster.

2.2.2 Utmaningar inom spoofing-detektion

Något som lyfts i forskningen är att många detekteringsystem fungerar bra när de testas mot attacker som de har redan sett förut. Men så fort de möter nya eller ovanliga typer av attacker så blir det svårare, vilket tydligt minskar träffsäkerheten i systemet. Li et al. visade i en omfattande studie att EER ökade när systemet testades på sådana nya typer av attacker [8]. Detta visar att det fortfarande är svårt att bygga system som klarar av alla typer av hot. Forskningen pekar också på att modeller som enbart tränats på ett fåtal typer av spoofing-

attacker har svårt att anpassa sig till nya. Det gör dem mindre användbara i verkliga situationer, där man inte alltid vet exakt vilken typ av spoofing som kan förekomma. Därför behövs det mer flexibla modeller som klarar av att upptäcka en större variation av attacker.

En av utmaningarna är att det finns begränsningar kopplade till datatillgång. Li et al. pekar på att det saknas tillräckligt många varierade och högkvalitativa dataset med inspelat tal som genererats genom spoofing, vilket gör det svårt att träna modeller på ett effektivt sätt [8]. Bristen gör att det också blir svårare att både träna och utvärdera detekteringssystem på ett tillförlitligt sätt, vilket bromsar utvecklingen av robusta lösningar för verkliga användningsområden.

Li et al. diskuterar att modeller som bara tränats på ett begränsat dataset ofta överanpassas, vilket leder till en tydlig försämring av prestandan vid test på nya dataset [8]. Problemet är att många dataset brister i omfattningen för att kunna appliceras i verkligheten, vilket gör det svårt för modellerna att lära sig generella och hållbara mönster som fungerar mot olika typer av spoofing-attacker. Därför är det en stor utmaning att utveckla system som är både tillförlitliga och mångsidiga.

Utöver bristen på data påverkas systemens prestanda även av olika externa faktorer, till exempel bakgrundsljud, komprimeringsförluster och skillnader i hur ljud spelas in. Yamagishi et al. visade att variationer i inspelningsmiljö kan påverka detekteringsprestandan avsevärt [9]. Detta gör det svårt att bygga system som fungerar stabilt i alla miljöer såsom i en bank, i en mobilapp eller via ett telefonsamtal.

Samtidigt är det viktigt att systemen är robusta mot variationer, där beräkningskrav också spelar roll för hur användbara de är i praktiken. Moderna transformer-baserade modeller, som wav2vec 2.0, har visat imponerande resultat i tester. Men som Baevski et al. påpekar kräver dessa modeller mycket processorkraft och energi [14].

Deras arbete visade att även om dessa modeller har lågt EER i kontrollerade tester är den höga latenstiden och energiförbrukningen ett problem för system som behöver vara snabba och fungera på olika typer av system såsom mobil eller inbyggda enheter.

Alla dessa tekniska utmaningar motiverar inte bara utvecklingen av bättre spoofing-detektion, utan kan också få ekonomiska konsekvenser. Om attackerna inte hanteras effektivt kan det leda till stora ekonomiska förluster och minskat förtroende för röstbaserade ekonomiska tjänster.

2.3 Traditionella metoder för att upptäcka röstspoofing

Innan deep learning slog igenom använde man mest traditionella metoder för detektering av röstspoofing. Lösningarna byggde på manuellt framtagna ljudegenskaper och använde statistiska modeller för att analysera dem.

Kinnunen och Li gjorde en viktig genomgång av tekniker för textoberoende röstigenkänning, där de fokuserade på hur spektrala egenskaper som MFCCs (Mel-Frequency Cepstral

Coefficients), LFCCs (Linear Frequency Cepstral Coefficients) och CQCCs (Constant Q Cepstral Coefficients) användes som grundläggande verktyg för att identifiera manipulerat tal [11]. Deras arbete satte en baslinje för prestanda inom området och visade att dessa funktioner fungerade bra på att fånga vissa delar av röstens, men hade svårt att hantera förändringar över tid, vilket är avgörande för att identifiera manipulerat tal.

För att lösa begränsningarna med spektrala egenskaper började forskare titta på fasbaserade funktioner. Wu et al. testade en metod som heter modified group delay (MGD) för att hitta små avvikelser i fasinformation i ljudsignaler. Deras tester visade positiva resultat men metoden var inte tillräckligt stark på att upptäcka mer avancerade syntetiska röster, särskilt de röster som lyckades efterlikna naturliga fasmönster [10].

Baserat på dessa tekniker för att hämta funktioner, använde man olika statistiska modeller för att avgöra om rösterna var äkta eller förfälskade. Bishop och Nasrabadi lade grunden för hur man tillämpar statistiska modeller som Gaussian Mixture Models (GMMs), Support Vector Machines (SVMs) och Hidden Markov Models (HMMs), och beskrev dessa metoder som kraftfulla verktyg inom mönsterigenkänning [18]. Samtidigt visade det sig senare att dessa modeller gav höga falsklarm då de hade svårt att bekämpa nya eller komplexa attacker [10].

Trots dessa svagheter spelade de traditionella metoderna en viktig roll i början. De hjälpte till att bygga upp säkerheten i de tidiga systemen för röstautentisering, alltså inom områden som ekonomi och bank där det är extra viktigt att skydda mot bedrägerier och bygga förtroende hos användare.

Sammanfattningsvis bygger traditionella metoder på handkonstruerade egenskaper och statistiska klassificerare. När spoofing-attacker blev mer avancerade ökade behovet av metoder som kan lära sig mer komplexa mönster direkt från data, vilket har gjort deep learning till ett naturligt nästa steg.

2.4 Deep learning-baserad spoofingdetektion

2.4.1 Convolutional Neural Networks (CNN)

CNN-modeller används för att analysera spektrogram som är visuella bilder av ljudets förändring över tid och frekvens. Simonyan och Zisserman visade att djupa CNN-nätverk extraherar detaljerade mönster i bilder och den idén har visat sig fungera mycket bra även för ljud [19]. Jämfört med traditionella metoder har CNN-modeller vid spoofing detektion visat bättre resultat med lägre EER, bland annat genom att minska felet som uppstår när systemet ska avgöra om ett ljud är äkta eller manipulerat. Det beror på att CNN är bra på att fånga upp små förändringar i ljudet som ofta avslöjar om det är syntetiskt eller uppspelat [8].

2.4.2 Recurrent Neural Networks (RNN) och Long Short-term Memory (LSTM)

Tal är en ljudsekvens som förändras över tid. För att kunna analysera sådana ljudsekvenser används modellerna RNNs och LSTMs. De är utvecklade för att minnas tidigare delar av ljudet, vilket gör dem bra på att förstå samband mellan olika delar över tid i talet.

Oord et al. presenterade modellen WaveNet som byggde rekursiva kopplingar för att realistiskt kunna skapa ljudvågor [20]. Den metoden har sedan används för spoofing detektion där målet är utöver att känna igen en röst, att upptäcka om en röst är äkta eller manipulerad.

När LSTM används för spoofing detektion har den visat sig vara bra på att upptäcka tidsmässiga avvikelser i talet, det vill säga små förändringar som kan tyda på att rösten inte är naturlig.

I praktiska tester har LSTM modeller lyckats minska felaktiga avvisningar, särskilt i miljöer där ljudkvaliteten varierar [8]. Detta visar att genom dessa modeller kan man förbättra noggrannheten och göra systemen mer tillförlitliga i verkliga användningsmiljöer.

2.4.3 Transformer-baserade-modeller

Transformer modeller, som introducerades av Vaswani et al., har förändrat möjligheterna att modellera långsiktiga beroenden i talet, det vill säga tidsmässiga samband mellan ljuddelar som ligger långt ifrån varandra [21]. Till skillnad från de tidigare modellerna CNN och LSTM kan transformer-modeller hantera mycket långa beroenden i talet. Detta gör dem extra effektiva för analys av komplexa ljudsekvenser. Modeller såsom wave2vec 2.0 som bygger på transformer-modeller, har visat bättre resultat än både CNN och LSTM metoderna vid spoofing-detektion [14]. Denna förbättrade prestanda beror till stor del på användningen av self-attention-mekanismer, som kan fånga in komplexa tidsmässiga samband i talet [21][14].

2.4.4 End-to-end-inläring och självövervakande modeller

En annan framväxande strategi inom spoofing-detektion är End-to-End modeller som har blivit allt mer populära och som kan lära sig direkt från rå ljuddata. Ett exempel är den självövervakade inläring som Baevski et al. presterade, kallat wave2vec 2.0, som använder sig av självövervakad inläring för att lära sig hur tal låter [14].

Det som gör den här metoden speciell är att den kan fungera utan att någon måste lyssna igenom och märka alla ljudfiler i förväg, alltså ljudfiler där någon redan har skrivit ner vad som sägs, vem som pratar eller andra detaljer. Istället tränas modellen på omärkta ljudinspelningar och modellen lär sig genom att försöka förutsäga vissa dolda delar av ljudet, vilket gör att den kan hitta mönster på egen hand. Detta visar att självövervakande modeller har god förmåga att skilja mellan äkta och manipulerade röster, tack vare att de är flexibla

och fungerar bra i många olika typer av miljöer och kan anpassas till varierande ljudkvalitet och inspelningsförhållanden [14].

Med tanke på hur viktigt det har blivit med säker röstautentisering inom bland annat bank, finansväsendet och telekommunikation, spelar dessa djupinlärningsbaserade metoder en avgörande roll. De hjälper till att skydda ekonomiska transaktioner, minska risken för bedrägerier och bibehålla förtroendet hos användare och kunder i digitala tjänster där rösten används som en säkerhetsfaktor.

2.5 Dataset och utvärderingsmått

2.5.1 Dataset

Dataset spelar en viktig roll inom forskning vid detektion av röstspoofing. De ger möjlighet för forskare att använda standardiserade plattformar med olika typer av spoofing-situationer under varierande förhållanden, både i kontrollerade laboratoriemiljöer och i mer realistiska situationer. Genom att använda sådana dataset kan man jämföra hur väl olika modeller presterar, samtidigt ger de värdefulla insikter i systemets prestanda i klassificeringsnoggrannhet och förmåga att fungera stabilt under varierande förhållanden. Dessa dataset innehåller olika typer av attacker, inspelningssätt och tekniska variationer, vilket hjälper dem att bli användbara för att upptäcka både styrkor och svagheter i detekteringsmetoder.

2.5.1.1 Standarddataset som används som jämförelse

Ett av de mest använda dataseten inom forskningen om röst-spoofing är ASVspoof 2019-LA [22], som utvecklades inom ramen för ASVspoof-utmaningen av Yamagishi et al [9]. Datasetet består uteslutande av engelskt tal, vilket används konsekvent i både äkta och manipulerade ljudexempel. Begreppet Logical Access syftar på en specifik delmängd inom ASVspoof-utmaningen som fokuserar på syntetiska algoritmiskt genererade spoofing-attacker, snarare än fysiska replay-attacker. Delmängden ASVspoof 2019-LA innehåller attacker baserade på text-to-speech (TTS) och voice conversion (VC), vilket gör den till en omfattande och välkontrollerad resurs för utvärdering av detektionssystemet. Samtidigt som detta dataset har blivit brett och användbart för att testa olika modeller och jämföra deras prestanda, har det också visat sig att det finns utmaningar med att generalisera resultat, särskilt när systemet tränats på ett dataset så kan det ha svårt att prestera bra på ett annat dataset [8].

För att minska detta problem utvecklades ASVspoof 2021-LA [23], som bygger vidare på samma LA-definition som introducerades i ASVspoof 2019-LA. Detta säkerställer konsekvens i attacktyperna, samtidigt som graden av realism ökas. ASVspoof 2021-LA baseras på engelskt tal precis som ASVspoof 2019-LA, vilket möjliggör en kontrollerad utvärdering av robusthet utan att introducera språkliga variationer. Till skillnad från sin föregångare utformades ASVspoof 2021-LA för att försöka minska skillnaden mellan laborietester och verkliga tillämpningar. I ASVspoof 2021-LA har man samlat in ljud

från riktiga telefonsystem. Det innebär att inspelningarna innehåller riktiga förvrängningar som är orsakade av olika codecs (används vid komprimering av ljud, kommunikationskanaler som mobilnät eller VoIP, varierande bithastigheter och olika samplingsfrekvenser).

Vanligtvis tränas ofta anti-spoofing-modeller med hjälp av träningen och utvecklingsdatan från ASVspoofer 2019-LA och därefter utvärderas modellen på ASVspoofer 2021-LA för att se hur väl de klarar att hantera okända förhållanden [23]. ASVspoofer 2021-LA är utformat utan att innehålla bakgrundsljud, vilket innebär att det blir lättare att fokusera på hur själva ljudöverföringen påverkar resultatet.

Utöver LA-delmängden introducerar ASVspoofer 2021 även en ny delmängd Deepfake (DF) [23] som fokuserar på det snabbt växande problemet med deepfake-tal. Här har både äkta och manipulerade ljudfiler bearbetats med förlust-komprimering, vilket gör data extra utmanande eftersom det kan inkludera olika typer av förvrängningar.

Uppsättningen med DF-utvärdering kombinerar ljudexempel från utvärderingsdelen av ASVspoofer 2019-LA med data från Voice Conversion Challenge (VCC) 2018 och 2020. Med hundratals spoofing-system och ett brett spektrum av källomäner utgör DF-delmängden en mycket krävande testmiljö för att utvärdera robustheten och generaliseringsförmågan hos ett spoofing-detektionssystem.

2.5.1.2 Dataset för syntetiskt tal (TTS och GAN)

Ett annat dataset som används inom spoofingforskning är FakeorReal original (FoR) [24]. Detta dataset består av engelska ljudinspelningar som innehåller både äkta och falska tal, där de falska rösterna har genererats med hjälp av TTS-algoritmer. Datasetet finns i tre olika versioner, där varje version har behandlats på olika sätt innan användning. Denna version gör att forskare kan undersöka hur själva förbehandlingen av ljud påverkar hur bra ett system är på att upptäcka spoofing. Samtidigt blir det möjligt att testa hur stabila olika detekteringsmetoder är när indata varierar.

För att kunna fördjupa förståelse av forskningen kring TTS-attacker introducerades datasetet WaveFake som är en mer kontrollerad miljö för att fokusera på hur GAN-baserade metoder för röstsyntes påverkar spoofing-detektion [25]. Detta dataset innehåller ljud från endast två olika högtalare och inspelningarna är helt fria från bakgrundsljud. WaveFake skapas med hjälp av sex olika GAN-baserade TTS algoritmer på både engelska och japanska. WaveFake är alltså ett värdefullt verktyg för att studera hur effektiva GAN-modellen kan vara och vilka utmaningar det innebär för anti-spoofing-detektion.

TIMIT-TTS är ett annat dataset som används för att undersöka hur ljudförsämring påverkar detekteringsförmågan [26]. Det består av syntetiskt tal som har genererats med hjälp av tolv avancerade TTS algoritmer. Dessa algoritmer fungerar främst som spektrogramgenerators men skillnaderna i det syntetiska talet uppstår genom variationer i hur vokoderna bearbetar ljudet. För att ytterligare utmana detekteringsystemen har datasetet även genomgått

efterbehandling där man har lagt till exempel som Gaussiskt brus, MP3-komprimering och efterklang. Detta gör TIMIT-TTS till en värdefull resurs för att testa hur robusta detekteringssystem är mot olika typer av ljudförvrängning.

2.5.1.3 Dataset som liknar verkliga miljöer

För att förstå hur detektion av röstspoofing fungerar i verkliga miljöer används ibland ett dataset som är mindre kontrollerat. In-the-Wild (ITW) [27] är ett exempel och det är utformat för att simulera realistiska förhållanden och innehåller bakgrundsljud och ljudprov från olika publika källor, såsom sociala medier och videoplattformar. Det består av både äkta och manipulerade ljudinspelningar från bland annat offentliga personer som kändisar och politiker som talar engelska. Det är användbart för att testa hur robusta detekteringssystem är i miljöer där ljudkvaliteten varierar och inte är perfekt och där störningar förekommer precis som i verkliga livet.

För att undersöka hur väl detekteringssystem fungerar när ljudkvaliteten är låg har forskare tagit fram dataset ADD2022-LF [28], som är en del av ADD2022-utmaningen. Det är en internationell tävling där forskare testar hur bra olika system fungerar för att upptäcka röstspoofing i svåra ljudmiljöer. Datasetet innehåller ljudexempel med verkliga störningar, som bakgrundsljud och musik. Detta gör det till en bra testmiljö för att se hur robusta systemen är i situationer som liknar den verkliga världen. Även om datasetet inte finns tillgängligt online för allmänheten, används det som ett viktigt riktmärke för att utvärdera hur väl olika system presterar när ljudkvaliteten inte är perfekt.

2.5.1.4 Dataset för olika språk

FMFCC-A [29] är ett mandarinspråkigt dataset för spoofing-detektion, skapat med TTS- och VC-algoritmer. Det är uppdelat i tränings-, utvecklings- och utvärderingsdelar, där utvärderingsdelen innehåller algoritmer som inte finns i träningsdelen. Hälften av utvärderingsdelen har komprimerats/dekomprimerats eller förstärkts med Gaussiskt brus för att bättre efterlikna verkliga förhållanden.

Utifrån FMFCC-A [29] har man även utvecklat ett ännu mer avancerat dataset som kallas Chinese Fake Audio Detection (CFAD) [30]. Det erbjuder en djupare utvärdering genom att inkludera mer information och data från flera olika områden, vilket minskar risken för skeva resultat. CFAD innehåller ljud som genererats med 12 olika spoofing-algoritmer, inklusive både TTS och VC. Det förbättrar på två viktiga sätt jämfört med FMFCC-A: för det första kommer det äkta ljudmaterialet från sex olika områden, vilket ger en mer varierad och rättvis databas, och för det andra innehåller datasetet detaljerade anteckningar om bland annat typ av attack, källa till det äkta ljudet, vilken typ av brus som används, samt brusnivå (signal-to-noise ratio) och vilka ljudkomprimeringar som förekommer. Dessa detaljer gör det möjligt att testa systemets förmåga på ett mer nyanserat sätt.

Utöver störningar i ljudmiljön är språk och dialekter också en utmaning. Därför har man skapat Latin-American Voice Anti-spoofing-dataset [31], som fokuserar på röster med fem

olika latinamerikanska spanska accenter. Det här är ett steg i rätt riktning för att testa hur bra systemet fungerar för att hantera olika typer av uttal och dialekter. Det gör att man får en tydligare bild av hur robusta systemen är när de ställs inför språklig och regional mångfald.

För att verkligen anpassa systemen ytterligare har man utvecklat ett ännu större dataset som heter Multi-Language Audio Anti-spoofing Dataset (MLAAD) [32]. Det här dataset innehåller spoofade ljudklipp på hela 23 olika språk, skapade med hjälp av 54 olika TTS-algoritmer. Det gör MLAAD till ett av de mest mångsidiga testverktygen hittills. Det används inte bara för att testa hur bra modeller fungerar på språk de aldrig tränats på, utan också för att träna modellerna så de klarar fler språk och scenarier. Datasetet har visat vara relevant vid globala användningsområden.

2.5.2 Utvärderingsmått

För att kunna bedöma hur väl ett system för detektion av röstspoofing fungerar behövs tydliga och objektiva utvärderingsmått. Utvärderingsmått spelar en viktig roll, där de ger forskare verktyg för att på ett rättvist sätt jämföra modeller för att värdera deras styrkor och svagheter. De tillhandahåller standardiserade kriterier och hjälper till att väga hur systemet hanterar falska godkännanden, det vill säga när manipulerat tal släpps igenom, samtidigt som de hanterar falska avvisanden där äkta tal stoppas. De används således för att mäta den övergripande noggrannheten i detektionen.

Mått som Equal Error Rate (EER) och Tandem Detection Cost Function (t-DCF) är viktiga och används ofta för att de fångar den viktiga balansen mellan systemsäkerhet och användbarhet. Förutom dessa används också klassiska klassificeringsmått som precision, recall och F1-score, där de ger en mer detaljerad information om hur modeller presterar i olika situationer. Här följer en kort beskrivning över de viktigaste utvärderingsmått som används inom det området.

2.5.2.1 Equal Error Rate (EER)

Enligt Kinnunen och Li är EER ett centralt mått för att kunna mäta balansen mellan falska godkännanden (FAR) och falska avslag (FRR) [11]. EER står för den punkt där andelen falska godkännanden (FAR) och andelen falska avslag (FRR) är lika, något som ger en mer balanserad bedömning av ett systems noggrannhet. Om EER är lågt så indikerar detta att systemet är mer pålitligt och säkert. EER påvisar således systemets förmåga att kunna minimera båda typerna av fel med att felaktigt acceptera falskt ljud som äkta samt även felaktigt avvisa äkta ljud som förfalskat. Detta har gjort EER till ett av de mest använda måtten i detekteringsforskning.

FAR och FRR kan formellt uttryckas enligt:

$$FAR(\tau) = FA / (FA + TN)$$

$$FRR(\tau) = FR / (FR + TP),$$

där FA betecknar falska godkännanden, FR falska avvísningar, TP sanna positiva och TN sanna negativa klassificeringar. Symbolen τ representerar beslutströskeln.

Equal Error Rate definieras som den punkt där dessa två fel är lika stora:

$$EER = FAR(\tau^*) = FRR(\tau^*),$$

där τ^* är det tröskelvärde där $FAR=FRR$.

2.5.2.2 Tandem Detection Cost Function (t-DCF)

Det här måttet används för att utvärdera den samlade prestandan som finns hos system för spoofing detektion och automatisk högtalarverifiering (ASV) när dessa används i samma autentiseringssystem. Detta mått, som togs fram av Yamagishi et al. är användbart i verkliga tillämpningar där man behöver avgöra om talet är äkta och samtidigt verifiera talarens identitet. Genom att kombinera resultaten från båda delarna ger t-DCF en mer helhetsbaserad bild av systemets robusthet och praktiska användbarhet [9].

Den förenklade matematiska representationen av t-DCF kan uttryckas som:

$$tDCF(\tau) = C1 * P_{miss_cm}(\tau) + C2 * P_{fa_cm}(\tau),$$

där P_{miss_cm} är sannolikheten att spoofing-detekteringssystemet missar ett förfalskat prov och P_{fa_cm} är sannolikheten att ett spoofat prov felaktigt accepteras. Konstanterna $C1$ och $C2$ är kostnadsparametrar definierade enligt ASVspoof-standardens.

Minimum t-DCF erhålls genom att beräkna funktionen över olika tröskelvärden och välja det lägsta värdet:

$$\min-tDCF = \min_{\tau} tDCF(\tau)$$

2.5.2.3 Klassificeringsmått

Bishop och Nasrabadi har i tidigare litteratur betonat hur viktigt det är att använda måtvärden som F1-score, precision och recall, speciellt i situationer där datasetet innehåller ett mer obalanserat antal exempel från olika klasser och betydligt fler äkta talprover än falska [18].

Via dessa mått ges en djupare förståelse om styrkor och svagheter hos en viss modell:

- Precision visar hur säkert ett system är på att välja ut spoofade röster. Det mäter alltså hur stor andelen av rösterna som systemet har klassat som spoof som faktiskt är spoofade.
- Recall visar hur bra systemet är på att hitta alla spoofade röster i datamängden.
- F1-score är ett balanserat mått som kombinerar både precision och recall. Det används ofta när datamängden är obalanserad, med exempel av det när det finns fler äkta röster än spoofade och man vill få en rättvis helhetsbild av modellens förmåga.

De klassiska klassificeringsmått definieras enligt:

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

$$\text{F1} = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}),$$

där TP, TN, FP och FN representerar de fyra möjliga utfallen i en confusion matrix.

2.6 Begränsningar hos befintliga metoder och framtida riktningar

2.6.1 Identifierade begränsningar

Trots att det har skett mycket positiv utveckling inom området detektion av röstspoofing, finns det fortfarande flera viktiga tydliga begränsningar kvar. Forskare har sett att de flesta av systemen har svårt att fungera utanför de miljöer som de är tränade i. Samtidigt har de svårt att generalisera till nya okända attack-typer och att många av de mer avancerade modellerna kräver mycket datorkraft för att fungera effektivt. Dessa problem leder ofta till högre felfrekvenser och försvårar användningen av systemen i praktiska tillämpningar i realtid.

Begränsningar medför också höga kostnader vid drift och underhåll, vilket kan vara extra kritiskt i känsliga sammanhang, som till exempel inom finanssektorn där tillförlitlighet är avgörande.

Li et al. fann att detektionssystem väldigt ofta ser en ökning av EER när de testas på tidigare osedda spoofing-attacker [8]. Detta är en betydande brist i förmågan hos nuvarande metoder att generalisera över olika och föränderliga attacktyper. Wu et al. rapporterade att även avancerade CNN-baserade modeller är väldigt sårbara för attacker där insignalen medvetet manipuleras för att lura system, vilket kan leda till högre falska acceptansfrekvenser [33]. Denna sårbarhet visar på behovet av ett mer stabilt skydd mot dessa typer av attacker. Baevski et al. lade även fram att även om transformatorbaserade modeller uppnår mer noggrannhet, så är deras höga beräkningskostnader en utmaning för denna realtidsdistribution, speciellt med tanke på resursbegränsade edge-enheter [14]. Detta begränsar dess praktiska användbarhet i ett flertal riktiga program.

2.6.2 Framtida forskningsinriktningar

Som ett svar på de begränsningar som finns hos de systemen som finns nu för upptäckt av röstspoofing så pekar litteraturen på flera bättre vägar för framtida forskning. Dessa har att göra med att förbättra generaliseringsförmågan genom att använda sig av domänanpassning,

förbättra modellens transparens med AI (XAI), utveckla stabila kontradiktoriska försvarsmekanismer och även skapa lätta arkitekturer för realtidsbearbetning i miljöer som är mer resursbegränsade. Det är viktigt att kunna hantera dessa forskningsriktningar, inte bara när det gäller att ta fram tekniken för upptäckt av förfalskning utan också för att stärka den ekonomiska säkerheten i röstbaserade finansiella tjänster [4][5].

Yamagishi et al. och Kang et al. påtalar hur viktigt det är med domänanpassningstekniker för att kunna göra modellens överförbarhet bättre över olika datamängder och även inspelningsförhållanden [34][9]. Tekniken skulle kunna bidra till att få bort gapet som finns just nu mellan mer experimentella miljöer som är mer kontrollerade och riktiga situationer, något som då skulle göra att tillförlitligheten i detektionssystemen blir allt högre.

Goodfellow et al. lägger fram som ett förslag att integrera XAI-metoder för att kunna öka tolkningsbarheten av djupinlärningsmodeller [35]. Eftersom man kan göra modeller mer transparenta så kan forskare bygga upp förtroende för sina beslut, underlätta hantering av modeller och på ett bättre sätt förstå faktorerna som för fram detektionsprestanda.

Tak et al. pratar om utvecklingen av stabila försvarsstrategier för att kunna motverka fientliga attacker [15]. Ju mer förfalskningstaktiken utvecklas är det viktigt att se till att detekteringssystemen fortfarande är motståndskraftiga mot fientliga störningar för att kunna ha trovärdiga prestanda i hotande landskap.

Kang et al. lyfter fram behovet av beräkningsmässigt effektiva modeller som kan fungera i realtid på enheter med begränsad datorkraft, såsom mobiler och edge-enheter, utan att förlora noggrannhet i detektionen [34]. Sådana arkitekturer gör det möjligt att faktiskt använda spoofing-detektionssystem i praktiska, verkliga tillämpningar. Trots viktiga framsteg finns det fortfarande flera utmaningar kvar, till exempel begränsad generalisering till osedda attacker, känslighet för attacker där insignalen medvetet manipuleras, samt höga beräkningskrav. Dessa begränsningar visar på nödvändigheten av denna studie, som syftar till att utforma ett robust och effektivt djupinlärningsbaserat system för detektion av röstspoofing som kan hantera dessa brister och driva utvecklingen framåt. Om framtida forskning dessutom anpassas till de ekonomiska kraven på kostnadseffektivitet och systemtillförlitlighet kan dessa framsteg bidra till att stärka konsumenternas förtroende och säkra finansiella transaktioner [4][5].

3. Metod

Detta kapitel beskriver metoden som användes för att utveckla och utvärdera modeller som kan upptäcka falska röster. Först motiveras valet av dataset och modellerna utifrån den teoretiska genomgången i kapitel 2. Därefter beskrivs hur data förbereddes, hur modellerna byggdes och hur träning samt utvärdering genomfördes. Syftet är att ge en tydlig beskrivning av arbetsflödet, så att läsaren ska kunna följa hela processen och förstå hur experimenten är upplagda.

3.1 Val av dataset och modeller

Urvalet av dataset och modeller baseras på att spoofade röster ofta ger små men syntetiska avvikelser i båda frekvensinnehåll och tidsmönster, såsom nämndes i kapitel 2. För att kunna analysera dessa skillnader valdes modeller som kan hantera både lokala mönster i ljudrepresentationer och sekventiella samband över tid.

Som dataset valdes ASVspoof 2019-LA, eftersom det innehåller både äkta och syntetiskt genererat tal samt används inom forskning om detektion av röstspoofing [9][22]. Datasets tydliga struktur och protokoll gör det lämpligt för en kontrollerad utvärdering och jämförelse mellan modelltyper.

I denna studie utvärderas fyra modeller: CNN, LSTM, GRU och BiGRU. CNN som är ett VGG-inspirerat konvolutionellt neuralt nätverk används för att identifiera lokala mönster i MFCC-representationer, medan LSTM och GRU används för att analysera tidsmönster i sekvenser, alltså hur talets egenskaper förändras över tid. BiGRU inkluderades för att i denna studie undersöka om bidirektionell sekvensanalys kan förbättra detektionen.

Mer resurskrävande modeller, till exempel transformerbaserade system, togs inte med i den slutliga utvärderingen eftersom fokus i detta arbete låg på modeller som är mer rimliga att träna och använda i praktiken, samtidigt som de fortfarande ger meningsfull jämförelse [14].

3.2 Förbehandling av data

Det här första avsnittet förklarar stegen som används för att förbereda ASVspoof 2019-LA [22], som sedan används för att träna modellerna. Datasetet går igenom en tydlig förbehandlingsprocess uppdelad i fyra huvudsteg: datainsamling och protokolltolkning, där ljudfiler samlas in och metadata läses av för att kunna komma fram till hur varje fil ska användas; organisering och balansering av klasser, där antalet äkta och manipulerade ljudprov jämnas ut; mappstrukturering, som ordnar filerna så att de kan läsas in enkelt under träning; samt funktionsutvinning med hjälp av Mel-frekvens-cepstrum-koefficienter (MFCC), en metod som omvandlar ljudet till en form som är lättare för modellen att analysera och som fångar viktiga egenskaper i talet [11].

3.2.1 Datainsamling och tolkning av protokoll

Det kompletta datasetet laddades ner från University of Edinburgh DataShare-arkivet. Efter att filerna packats upp lästes en protokollfil (ASVspoof 2019.LA.cm.train.trn.txt), som innehåller metadata för varje ljudprov, in i en pandas DataFrame, en tabell i Python som ofta används för att hantera data. Tydliga kolumnnamn tilldelades (speaker_id, filename, system_id och class_name), och en överflödig kolumn togs bort. En ny kolumn, *filepath*, skapades genom att slå ihop basmappen för ljudfilerna med varje filnamn, så att hela sökvägen till varje fil bildades. Slutligen skapades en binär målvariabel, där 0 markerade äkta ljudprov och 1 markerade manipulerade ljudprov.

3.2.2 Sortering och balansering av klasser

För att underlätta träning av modellerna sorterades ljudfilerna i två separata mappar (./0 och ./1) utifrån deras målvariabler. Varje mapp kontrollerades och skapades om den inte redan fanns, och filerna kopierades sedan till sina respektive mappar. En första granskning av data visade en obalans mellan klasserna, med 2 580 äkta ljudprov och 22 800 manipulerade ljudprov. För att åtgärda detta raderades ett fast antal slumpmässiga filer från varje mapp (1 580 från den äkta mappen och 21 800 från den manipulerade), så att de två klasserna kom närmare varandra i antal och risken för att modellen skulle bli partisk mot den större klassen minskade.

Även om ett alternativ hade varit att behålla samtliga äkta ljudprov och enbart reducera den manipulerade klassen, valdes ett balanserat urval från båda klasserna för att skapa stabila träningsförhållanden och minska risken för överanpassning. Detta möjliggjorde även en mer rättvis jämförelse mellan de olika modellarkitekturena.

Efter att klasserna balanserats samlades ljudfilerna i en gemensam mapp med separata underkataloger för äkta respektive manipulerade ljud. Denna struktur gjorde det enklare att läsa in och hantera datan automatiskt under modellträningen.

3.2.3 Extraktion av egenskaper med MFCC

Ljutfunktioner skapades med hjälp av MFCC, som fångar ljudets korttidsenergi och är särskilt lämpliga för tal- och detektion av röstspoofing [11][12].

Vid extraktionen användes en samplingsfrekvens på 22 050 Hz och en spårlängd på 30 sekunder. För att analysera signalens frekvensinnehåll användes en FFT-fönsterstorlek på 2 048 prover och en hopplängd på 512 prover. Vidare extraherades 13 MFCC-koefficienter per tidsruta och varje ljudfil delades upp i 10 segment.

Dessa inställningar valdes för att ge en stabil och informativ representation av talets spektrala egenskaper, vilket är centralt vid detektion av manipulerat tal. En samplingsfrekvens på 22 050 Hz är tillräcklig för att fånga de frekvenser som är viktiga för mänskligt tal, utan att göra beräkningarna onödigt krävande. Kombinationen av fönsterstorlek och hopplängd ger en bra balans mellan tids- och frekvensinformation, vilket är viktigt för att kunna upptäcka

skillnader mellan äkta och manipulerat tal. Att använda 13 MFCC-koefficienter är ett etablerat standardval inom talbehandling och ger en tydlig men samtidigt kompakt representation av ljudets spektrala egenskaper. Uppdelningen i tio segment per ljudfil gjordes för att alla indata skulle få samma storlek och för att modellen skulle bli mer stabil under träningen.

Varje ljudfil lästes in i minnet och delades upp i tio lika långa segment. För varje segment beräknades MFCC och omformades till en tabell med storleken (antal tidsrutor \times antal MFCC-koefficienter). Segment som gav rätt antal rader behölls, medan segment som inte stämde med förväntad storlek slängdes för att säkerställa att alla data hade samma format. De beräknade MFCC-tabellerna och tillhörande etiketter sparades i en Python-ordbok tillsammans med en lista som visar vilka klasser som finns. Slutligen sparades ordboken i en JSON-fil för att lätt kunna användas under modellträningen.

3.3 Djup-inlärningsmodeller

I denna studie analyserades flera skilda djupinlärningsmodeller för att göra binär klassificering av MFCC-funktioner som extraherats från ljudsignaler. Modellerna som testades för detta var konvolutionella nätverk (CNN), som inspirerades av VGGNet, vilket använder små och lika filter i alla lager. Även olika rekurrenta nätverk, såsom LSTM, GRU och BiGRU användes. Alla dessa var framtagna för att kunna hantera sekvenser och tidsberoenden i data.

Alla modeller skapades med hjälp av TensorFlow och Keras, som är två öppna ramverk för djupinlärning och dessa gör det enkelt att skapa och kunna träna modeller. Modellerna och deras inställningar valdes för att uppnå en balans mellan detekteringsprestanda och beräkningskrav, med fokus på praktisk användbarhet. Under avsnittet nedan beskriver varje modell mer i detalj, med teorin bakom, varför den utformades på det sättet och vilka justeringar som gjordes i denna studie.

3.3.1 Konvolutionella neurala nätverk (CNN)

Ett konvolutionellt nätverk (CNN) användes för att ta fram lokala mönster i ljudets tids- och frekvensinformation. Modellen var ett CNN inspirerat av VGGNet [19]. Denna typ av nätverk bygger på en konvolutionell arkitektur, även kallad faltningssarkitektur, där modellen använder konvolutionella lager för att analysera indata.

Det egna CNN-nätverket byggdes för att stegvis fånga mönster i ljudet genom flera lager, där varje lager gjorde mönstren mer abstrakta. Detta innebär att de första lagren identifierar enklare och mer lokala egenskaper i ljudets frekvensrepresentation, medan de djupare lagren kombinerar dessa till mer komplexa mönster. Denna stegvisa abstraktion är relevant vid spoofing-detektion, för att identifiera talets subtila skillnader i både tids- och frekvensdomänen. Genom att analysera dessa lokala mönster kan modellen skilja mellan äkta och manipulerat tal.

Modellen förenklades jämfört med originalet VGG16 [19] för att minska antalet parametrar och anpassa kapaciteten till datasetets storlek. Eftersom ASVspoof 2019-LA är ett kontrollerat dataset med begränsad mängd träningsdata [22] skulle en alltför stor modell riskera överanpassning. Genom att använda en mer kompakt arkitektur kunde en balans uppnås mellan detekteringsförmåga och generaliseringsförmåga.

3.3.1.1 VGG-inspirerad CNN

Den VGG-inspirerade CNN-modellen valdes eftersom konvolutionella nätverk är bra på att analysera MFCC-baserade spektrogram, samtidigt som en förenklad version av VGGNet kan användas för att minska både minnesanvändning och beräkningskostnad. Utifrån detta byggdes ett eget CNN-nätverk inspirerat av VGGNet. Modellen skapades med TensorFlow och Keras och använde små 3×3 filter som gled över data för att upptäcka lokala mönster. ReLU-aktiveringar och padding användes för att behålla storleken på funktionerna efter varje lager. Nätverket hade tre block som ökade antalet filter steg för steg: först 32 filter, sedan 64 och slutligen 128, med maxpooling-lager mellan som minskade storleken på datan. Efter dessa block gjordes datan om till en lång lista och skickades vidare till ett fullt anslutet lager med 256 neuroner för att lära sig mer avancerade mönster. Ett dropout-lager slog av hälften av neuronerna slumpmässigt under träningen för att minska risken för överanpassning. Slutligen användes ett lager med sigmoid-aktivering som gav sannolikheten för om ljudet var äkta eller manipulerat. Denna modell behöll VGGNets idé om att hitta mönster steg för steg, men var enklare och snabbare, anpassad för datasetets storlek och uppgiften.

Det föreslagna CNN-nätverket som var inspirerat av VGG var en förenklad version av VGG16. Antalet parametrar minskades från ungefär 138 miljoner till 0,8 miljoner genom att använda färre konvolutionsblock och mindre fullt anslutna lager, se tabell 3.1.

Tabell 3.1: Jämförelse mellan originalet VGG16 och den modifierade VGG-nätverk

Komponent	Original VGG16	Modifierad VGG-modell
Konvolutionsblock	(2*64,2*128,3*256,3*512, 3*512)	2*(32),2*(64),2*(128)
Fullt anslutna lager (FC)	[4096, 4096, antal klasser]	[256, 1] (sigmoid)
Dropout	Ingen	0,5 efter FC (256)
Aktiveringsfunktion	ReLU	ReLU
Utgång	Softmax	Sigmoid (för binärt resultat)
Antal parametrar	ca 138 miljoner	811 233

Denna förenkling har både fördelar och nackdelar. Modellen kan inte lära sig lika djupa och detaljerade mönster som VGG16 kan, vilket kan göra det svårare att hitta mycket små tecken på manipulerat ljud. Men för ASVspoof-datasetet behövs inte så mycket kapacitet, och en större modell skulle kunna överanpassa sig, alltså bara komma ihåg träningsdata istället för att fungera på nya ljud.

Den enklare modellen tränades snabbare, använde mindre minne och minskade risken för att inlärningen stannar av. Dropout-lagret hjälpte också till att minska överanpassning eftersom träningsdatan var liten.

Trots förenklingen var modellen fortfarande bra på att hitta mönster i ljudets frekvens och tid, något som är relevant vid identifiering mellan på äkta och manipulerade röster. Sammanfattningsvis gav denna design en effektiv och stabil modell som fungerade bra för uppgiften, även om det fanns brister vid mycket avancerad eller ny typ av manipulation.

3.3.2 Rekurrenta neurala nätverk (RNN)

Två typer av rekurrenta neurala nätverk (RNN) byggdes: LSTM och GRU samt den bidirektionella varianten BiGRU. Dessa nätverk användes för att hitta mönster över tid i sekventiella data, alltså samband mellan händelser som sker efter varandra.

LSTM och GRU valdes för att de kan bevara information från tidigare steg under långa sekvenser [13][37]. Den bidirektionella varianten BiGRU, kan dessutom bearbeta sekvensen både framåt och bakåt, vilket innebär att modellen kan ta hänsyn till både tidigare och senare delar av ljudsignalen samtidigt [38].

Dessa modeller gjorde det möjligt för nätverket att lära sig hur ljudfunktioner förändrades över tid i ljudsekvenser och att fånga tidsmönster på ett effektivt sätt.

3.3.2.1 LSTM

LSTM-modellen valdes för att den är väl lämpad för att analysera sekventiella ljuddata och kan fånga långsiktiga tidsmönster [13], samtidigt som den kräver mindre beräkningsresurser än mer avancerade råljudsbaseade eller transformerbaseade modeller. För att modellen skulle kunna förstå mönster över tid och långsiktiga samband i ljuddata byggdes en djupt LSTM-modell med TensorFlow/Keras. Nätverket kunde hantera ljudsekvenser av olika längd eftersom det började med ett maskeringslager som ignorerade padding (värde = 0) så att dessa steg inte påverkade inlärningen.

För att modellen skulle bli mer robust lades lite slumpmässigt brus (Gaussian noise) till i indata under träningen. Bruset hade låg styrka och användes enbart som en regulariseringsmetod, för att hjälpa modellen att inte anpassa sig för hårt till träningsdatan. Samma nivå av brus användes för alla modeller där brus applicerades, vilket gjorde jämförelsen mellan modellerna rättvis.

Vi använde också dataaugmentation inspirerad av SpecAugment, som TimeMasking och FrequencyMasking applicerades. Det innebär att små delar av ljudet tillfälligt döljs i tid eller frekvens under träningen. Detta hjälpte modellen att klara av variationer och nya ljud. Maskeringen hölls på en låg nivå och samma inställningar användes för alla modeller.

Syftet var att efterlikna naturliga variationer i tal, till exempel bakgrundsstörningar eller varierande ljudkvalitet, utan att förändra ljudet så mycket att viktig information förlorades.

Själva LSTM-arkitekturen bestod av två lager. Det första lagret hade 256 enheter och skickade hela sekvensen vidare till nästa lager. Ett dropout-lager (30 %) följde för att minska risken för överanpassning. Det andra lagret hade 128 enheter och gjorde om sekvensen till en fast vektor som representerade hela tidsförloppet, följt av ett nytt dropout-lager (30 %).

För klassificering användes ett fullt anslutet lager med 256 ReLU-aktiverade neuroner och ett dropout-lager (50 %). Slutligen användes ett lager med sigmoid-aktivering som gav sannolikheten för om ljudet var äkta eller manipulerat.

Denna arkitektur kombinerade minne över tid, maskering av sekvenser och kraftig regularisering för att lära sig komplexa tidsmönster i ljud, samtidigt som risken för överanpassning hölls nere, som visades i tabell 3.2.

Tabell 3.2: Jämförelse mellan original-LSTM och den modifierade LSTM-modellen.

Komponent	Original LSTM	Modifierad LSTM-modell
Enheter(units)	Beroende på uppgift (t.ex 128)	256→128 (två lager)
Dropout	Ingen	0,3 efter varje LSTM-lager
Dense-lager	Ett enkelt utgångslager	FC (256) + dropout (0,5)
Utgång	Varierar beroende på uppgift	Sigmoid
Antal parametrar	Ca 72 000–150 000	506 881

Den modifierade LSTM-modellen byggde vidare på grund-LSTM med fler lager, dropout och dataaugmentation av indata. De två lagren (256 → 128 enheter) gav tillräckligt med minne för att upptäcka långsiktiga samband utan att modellen blev alltför stor. Under träningen hjälpte dropout och maskering till att stabilisera inläringen och således minska risken för att överanpassning skulle ske, medan slumpmässigt brus och SpecAugment gjorde modellen mer stabilare genom att simulera variationer som fanns i ljudet.

När det kom till att upptäcka manipulerade röster kunde modellen på ett bra sätt fånga tidsberoenden som fick fram falska ljud. Då modellen var rätt liten (0,5 miljoner parametrar) kunde den då ha svårt med mycket komplexa spoofing-varianter eller helt nya typer av attacker, jämfört med mer djupa eller bidirektionella modeller.

3.3.2.2 GRU

GRU-modellen valdes som ett enklare alternativ till LSTM, eftersom den har färre parametrar och lägre beräkningskostnad, men ändå kan fånga viktiga tidsmässiga mönster i MFCC-baserad taldata [37]. Utifrån detta byggdes en GRU-modell med flera lager i TensorFlow/Keras. Modellen började med ett maskeringslager som inte tog hänsyn till steg med värdet 0, så att sekvenser med olika längd kunde hanteras på korrekt sätt.

För att modellen skulle bli mer stabil och kunna fungera bra på nya data lades lite slumpmässigt brus (Gaussian noise) till ljudindatan. Vi använde också dataaugmentation som var inspirerad av SpecAugment: TimeMasking och FrequencyMasking, där delar av sekvensen eller frekvensen doldes på ett slumpmässigt sätt. Detta hjälpte till att simulera variationer som kunde förekomma i verkliga ljud.

Nätverket bestod av två GRU-lager. Det första lagret hade 256 enheter och skickade hela sekvensen vidare för att bevara tidsinformation. Ett dropout-lager (30 %) följde för att minska risken för överanpassning. Det andra lagret hade 128 enheter och gjorde om sekvensen till en fast vektor, följt av ett nytt dropout-lager (30 %).

För klassificering användes ett fullt anslutet lager med 256 ReLU-aktiverade neuroner och ett dropout-lager (50 %). Slutligen användes ett lager med sigmoid-aktivering som gav sannolikheten för om ljudet var äkta eller manipulerat.

Denna GRU-modell kombinerade förmåga att förstå tidsmönster med låg parameterkostnad, vilket gjorde den lämplig för realtidsapplikationer eller system med begränsade resurser, som visas i tabell 3.3.

För att omvandla denna information till ett klassificeringsresultat används ett dense-lager med 256 neuroner och ReLU-aktivering. Eftersom lagret hade hög kapacitet lades ett dropout-lager med 0,5 till för att förhindra att överanpassning skedde.

Tabell 3.3: Jämförelse mellan original-GRU och den modifierade GRU-modell

Komponent	Original GRU	Modifierad GRU -modell
Enheter	Beroende på uppgift	256→128 (två lager)
Dropout	Ingen eller uppgiftsberoende	0,3 efter varje GRU -lager
Dense-lager (FC head)	Ett enkelt utgångslager	FC (256) + dropout(0,5)
Utgång	Varierar beroende på uppgift	Sigmoid
Antal parametrar	Ca 55 000–110 000	389 633

Den modifierade GRU-modellen balanserade effektivitet och förmågan att förstå tidsmönster. Modellen hade färre parametrar (~0,39 miljoner) jämfört med LSTM, men kunde ändå lära sig långsiktiga samband. Under träningen gick GRU snabbare eftersom den hade enklare styrning av informationen. Den staplade designen med dropout gjorde att modellen kunde lära sig bra representationer utan att överanpassa sig. När det gällde att upptäcka manipulerat ljud fungerade GRU ofta lika bra som LSTM, men den enklare strukturen kunde göra att modellen missade mycket små variationer i avancerade spoofing-attacker. Modellen var ändå bra att använda i system med begränsade resurser.

3.3.2.3 BiGRU

BiGRU-modellen valdes för att kombinera fördelarna med bidirektionell analys och den mer resurseffektiva GRU-strukturen, vilket gör modellen lämplig för praktisk och robust spoofing-detektion. För att modellen skulle kunna förstå både tidigare och framtida samband i ljudsekvenser byggdes ett BiGRU-modellen med TensorFlow/Keras. Modellen började med ett maskeringslager som ignorerade tidssteg med värdet 0, så att padding inte påverkade träningen.

För att göra modellen mer robust lades lite slumpmässigt brus (Gaussian noise) till på indata. Dessutom användes också dataaugmentation inspirerad av SpecAugment: TimeMasking och FrequencyMasking, där delar av sekvensen eller frekvensen doldes slumpmässigt. Detta hjälpte till att simulera variationer som kunde förekomma i verkliga ljud.

Kärnan i nätverket bestod av två BiGRU-lager. Det första lagret hade 256 enheter i varje riktning och skickade hela sekvensen vidare med information från både tidigare och senare steg. Ett dropout-lager (30 %) följde för att minska risken för att överanpassning skedde. Det andra lagret hade 128 enheter per riktning och omvandlade sekvensen till en fast vektor som inkluderade information från båda riktningarna, följt av ett nytt dropout-lager (30 %).

För klassificering användes ett fullt anslutet lager med 256 ReLU-aktiverade neuroner och ett dropout-lager (50 %). Slutligen användes ett lager med sigmoid-aktivering som gav sannolikheten för om ljudet var äkta eller manipulerat.

Denna BiGRU-modell kombinerade effektiviteten hos GRU med möjligheten att läsa sekvenser framåt och bakåt, vilket gjorde att den kunde fånga upp detaljerade tidsmönster i ljuddata [38], som visades i Tabell 6.

Tabell 3.4: Jämförelse mellan original-BiGRU och den Modifierade BiGRU-modellen.

Komponent	Original BiGRU	Modifierad BiGRU -modell
Enheter per riktning	till exempel 128	256→128 (två lager)
Dropout	Ingen eller uppgiftsberoende	0,3 efter varje GRU -lager
Dense-lager (FC head)	Ett enkel utgångs-lager	FC (256) + dropout(0,5)
Utgång	Varierar beroende på uppgift	Sigmoid
Antal parametrar	Ca 109 000–220 000	975 361

BiGRU byggde vidare på GRU genom att läsa av sekvenser både framåt och bakåt. Den hade fler parametrar (~0,97 miljoner) än en vanlig GRU, något som gjorde att modellen kunde förstå både tidigare och framtida tidsberoenden.

Träningen tog även längre tid och krävde mer beräkningar än för GRU, men med hjälp av regularisering kunde träningen hållas stabil.

När det kom till att upptäcka manipulerat ljud gjorde den bidirektionella läsningen att modellen kunde hitta subtila tidsvariationer i falskt tal, vilket gav bättre noggrannhet. Nackdelen dock var att den ökade komplexiteten gjorde modellen mindre lämplig för system som behövde snabbare realtidsprestanda.

3.4 Implementationsmiljö

Arbetet genomfördes i Python med hjälp av TensorFlow och Keras. Utvecklingen och experimenten kördes lokalt på en Windows-dator med Visual Studio Code och Jupyter Notebook som utvecklingsmiljöer.

Vid behov kan samma kod även köras i Google Colab, vilket möjliggör användning av GPU-acceleration. Detta kan minska träningstiden vid mer beräkningskrävande experiment, exempelvis vid upprepade modellträning eller jämförelse mellan flera arkitekturer.

3.5 Modellens träningsflöde

Detta avsnitt beskriver hela processen för att träna och utvärdera modellen som klassar ljud som äkta eller manipulerat, med hjälp av redan beräknade MFCC-funktioner. Denna process inkluderar hantering av data, förstärkning av ljuddata, design av modeller, speciella utvärderingsmått för området samt tydlig visualisering av resultaten.

För att kunna jämföra modellerna på ett rättvist sätt användes de utvärderingsmått som beskrevs i kapitel 2. I denna studie valdes Equal Error Rate (EER) och minimum t-DCF som huvudmått, eftersom de är vanligt förekommande inom ASVspoof och ger en bra balans mellan säkerhet och användbarhet [9][22]. Definitioner och formler för måtten presenteras i avsnitt 2.5.2.

Utöver dessa beräknades även noggrannhet, precision, recall och F1-poäng. Dessa mått ger en mer detaljerad bild av hur modellerna presterar, särskilt när det gäller att skilja mellan äkta och manipulerat tal [18].

3.5.1 Förbereda och läsa in data

Indata och etiketter sparades i en JSON-fil som innehöll Mel Frequency Cepstral Coefficients (MFCC) och deras tillhörande binära klassetiketter. När filen laddades omvandlades MFCC-funktionerna till NumPy-arrayer och formaterades om för att ta bort ensamma extra dimensioner, så att de passade som input till modellen, som visas i kodexempel 3.1 nedan.

```
def load_data(json_path):
    with open(json_path, "r") as f:
        data = json.load(f)
    X = np.array(data["mfcc"])[..., np.newaxis] # Add channel dimension
    y = np.array(data["labels"])
    return X, y
```

Kodexempel 3.1: Kod för att ladda MFCC-funktioner och deras tillhörande klassetiketter från den förbehandlade JSON-filen.

Efter en lyckad inläsning bekräftades formerna på funktions- och etikettmatriserna för att kunna säkerställa att datamängden var redo för modellträning.

3.5.2 Uppdelning av dataset

Datasetet delades upp med stratifierad sampling för att behålla samma fördelning av klasser i alla delar. Först reserverades 10 % av datan som testset. De återstående 90 % delades sedan upp i tränings- och valideringsset, där 20 % användes för validering, som visas i kodexempel 3.2.

En fast slumpmässig seed-inställning användes för att säkerställa att uppdelningen går att reproducera, som visas i kodexempel 3.3.

```
X_train_full, X_test, y_train_full, y_test = train_test_split(
    X, y, test_size=0.1, stratify=y, random_state=SEED
)
X_train, X_val, y_train, y_val = train_test_split(
    X_train_full, y_train_full, test_size=0.2, stratify=y_train_full, random_state=SEED
)
```

Kodexempel 3.2: Kod för att dela upp datasetet i tränings-, validerings- och testset med stratifierad sampling för att behålla klassfördelningen.

```
SEED = 42
tf.random.set_seed(SEED)
np.random.seed(SEED)
```

Kodexempel 3.3: Kod för att sätta ett fast slumpmässigt seed.

3.5.3 Modellens kompilering

Modellens arkitektur definierades med en platsfunktion `build_model()` (som fylls med en passande modell, t.ex. BiGRU eller CNN). Modellen tog emot indata med samma form som träningssetet och kompilerades med Adam-optimizeren, binary cross-entropy som förlustfunktion samt viktiga klassificeringsmått: noggrannhet, precision och recall som visas i kodexempel 3.4.

```
model = build_model(X_train.shape[1:])
model.compile(
    optimizer='adam',
    loss='binary_crossentropy',
    metrics=['accuracy',
             tf.keras.metrics.Precision(name='precision'),
             tf.keras.metrics.Recall(name='recall')]
)
model.summary()
```

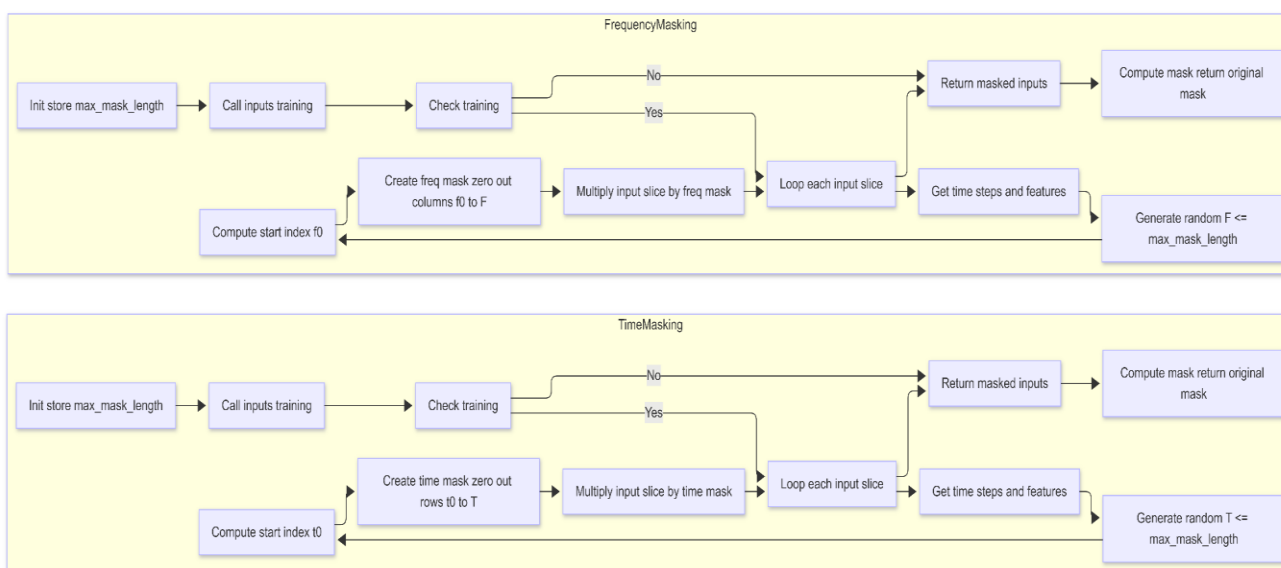
Kodexempel 3.4: Kod som visar hur modellen kompileras med Adam-optimizeren, binary cross-entropy som förlustfunktion och utvärderingsmått som noggrannhet, precision och återkallande.

3.5.4 Förstärkning av data

För att göra modellen mer robust och bättre på att generalisera lades två egna förstärkningslager till: ett TimeMasking-lager som slumpmässigt döljer en del av ljudet i tidsaxeln, och ett FrequencyMasking-lager som slumpmässigt döljer en del av ljudet i

frekvensaxeln, vilket illustreras i figur 3.1. Den fullständiga implementeringen återfinns i bilaga A som visas i kodexempel 3.5.

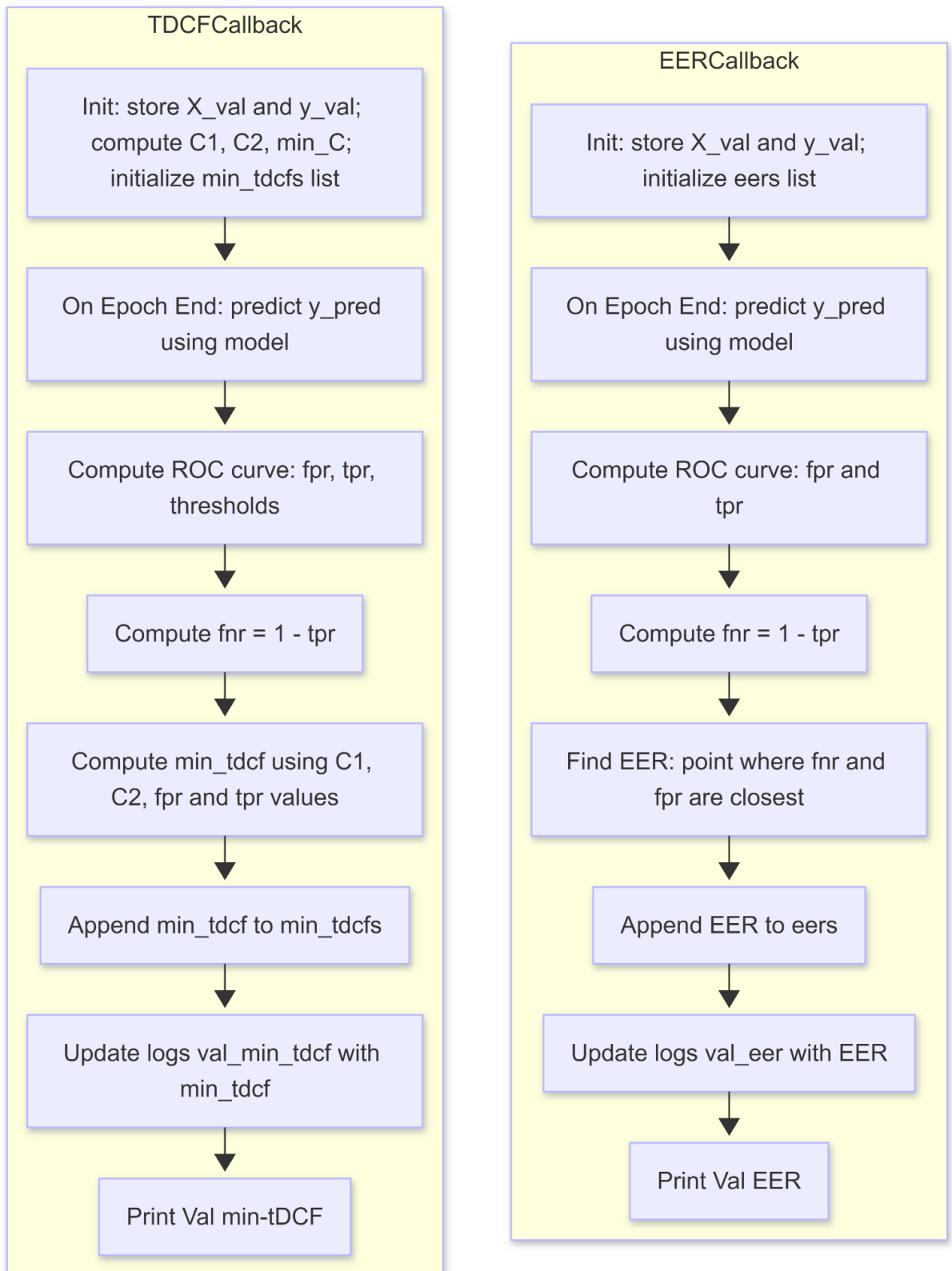
Dessa lager fungerar med Keras, stöder maskering och används endast under träning för att simulera verkliga variationer i ljudet, och är inspirerat av SpecAugment.



Figur 3.1. Flödesschema över TimeMasking och FrequencyMasking-lagren som används för dataförstärkning under träning.

3.5.5 Egenbyggda callbacks för utvärdering anpassad till området

För att följa ASVspoof 2019-utmaningens krav definierades två egna callbacks. Den första, EERCallback, beräknar Equal Error Rate (EER) på valideringssetet genom att hitta punkten där False Acceptance Rate (FAR) är lika med False Rejection Rate (FRR) [9]. Den andra, TDCFCallback, beräknar min-tDCF (minimum tandem Detection Cost Function), ett kostnadsbaserat mått som kombinerar fel från ASV och spoof-motåtgärder, med officiella ASVspoof-parametrar [9]. Dessa callbacks sparar och skriver ut respektive mått i slutet av varje träningsepoker, vilket illustreras i figur 3.2. Den fullständiga implementeringen återfinns i bilaga B som visas i kodexempel 3.6.



Figur 3.2. Flödesschema för de egenutvecklade EER och min-tDCF callbackarna som används för utvärdering under träning

3.5.6 Early Stopping och träning av modellen

Träningen kördes i högst 30 epoker med en batchstorlek på 32. En epok, eller träningsepok, innebär att modellen har gått igenom hela träningsdatasetet en gång. När resultat presenteras som exempelvis 30/30 betyder det att modellen har tränats i samtliga 30 epoker som angavs som maximal gräns. Om det i stället står 25/30 innebär det att träningen avbröts efter 25 epoker, trots att den maximala gränsen var 30. Detta beror på användningen av så kallad early stopping.

Early stopping användes för att övervaka prestandamåttet min-tDCF på valideringsdatan och avbryta träningen om ingen förbättring observerades under fem på varandra följande epoker. Eftersom denna mekanism stoppar träningen när valideringsprestandan slutar förbättras, kan olika modeller avsluta träningen efter olika antal epoker. Det förklarar varför vissa modeller når hela 30/30 epoker, medan andra stoppas tidigare, till exempel vid 29/30, 27/30 eller 25/30.

Att träningen avslutas tidigare innebär inte att modellen tränats ofullständigt. Tvärtom indikerar det att modellen redan har nått sin bästa valideringsprestanda och att de vikter som gav detta resultat automatiskt återställdes.

De bästa vikterna återställdes automatiskt, och visas i kodexempel 3.7.

```
metrics_callbacks = [  
    eer_callback,  
    tdcf_callback,  
    callbacks.EarlyStopping(  
        monitor='val_min_tdcf',  
        patience=5,  
        mode='min',  
        restore_best_weights=True  
    )  
]
```

Kodexempel 3.7: Kod för att implementera early stopping baserat på valideringsmåttet min-tDCF, där träningen avbryts om ingen förbättring observeras under fem på varandra följande epoker och de bästa vikterna återställs automatiskt.

3.5.7 Slutlig utvärdering och tröskelvärdesbestämning

Den tränade modellen utvärderades därefter på den oberoende testuppsättningen. Inledningsvis genererades prediktioner med ett standardiserat beslutströskelvärde på 0,5. Ett beslutströskelvärde, eller threshold, anger den gräns vid vilken modellen avgör vilken klass ett ljudprov tillhör. I denna studie innebär ett tröskelvärde på 0,5 att ljud klassificeras som manipulerat (klass 1) när modellens utdata är lika med eller överstiger 0,5, och som äkta (klass 0) när värdet understiger denna gräns.

Valet av tröskelvärde påverkar direkt balansen mellan falska godkännanden och falska avvísningar. Efter klassificeringen beräknades Equal Error Rate (EER) samt minimum t-DCF med hjälp av de officiella ASVspooft-formlerna, vilka presenteras i kodexempel 3.8 nedan.

```
test_eer, test_min_tdcf = compute_asvspoof_metrics(y_test, y_test_scores, **ASV_PARAMS)
```

Kodexempel 3.8: Kod för att beräkna Equal Error Rate (EER) och minimum tandem Detection Cost Function (min-tDCF) på testsetet med officiella ASVspooft-parametrar [9].

Det är även så att vanliga prestandamått beräknades — noggrannhet, precision, recall och F1-poäng—baserat på prediktioner som jämfördes med tröskelvärdet [18].

3.5.8 Visualisering och analys

Utvärderingen inkluderar flera viktiga visualiseringar för att kunna ge en tydlig bild av modellens prestanda. Träningskurvor visar hur förlust och noggrannhet utvecklades under träningen och valideringen, något som ger en bättre insikt i inlärningsprocessen. Mått över epoker, som Equal Error Rate (EER) och min-tDCF, visas i diagram för att trenderna ska kunna följas under träningen.

Receiver Operating Characteristic (ROC)- och Precision-Recall-kurvor visar hur modellen presterar vid olika tröskelvärden. ROC-kurvan visar sambandet mellan True Positive Rate (TPR) och False Positive Rate (FPR) vid olika tröskelvärden. AUC (Area Under the Curve) beskriver arean under ROC-kurvan och används som ett sammanfattande mått på modellens förmåga att skilja mellan de två klasserna, där ett högre värde indikerar bättre prestanda. Precision-Recall-kurvan illustrerar sambandet mellan precision och återkallelse och är särskilt användbar för att analysera hur modellen presterar vid olika beslutströsklar, särskilt vid obalanserade datamängder.

DET-kurvan (Detection Error Tradeoff) används för att tydligt visualisera avvägningen mellan falsk acceptans och falskt avvísande och är ett vanligt verktyg inom biometriska system för att jämföra olika felmönster. För att ytterligare analysera klassificeringsresultaten genererades en confusion matrix vid standardtröskeln 0,5. Confusion matrix visar hur många ljudprov som klassificerats korrekt respektive felaktigt och sammanfattar fyra utfall: korrekt avvísade förfalskningar, korrekt accepterade äkta prover, falska godkännanden samt falska avvísningar [18].

Utöver detta analyserades även poängfördelningar i form av histogram, vilka visar hur modellen fördelar sina utdata för äkta respektive manipulerade ljudprov. Tillsammans ger dessa visualiseringar en fördjupad förståelse för modellens beteende, inklusive konvergens, eventuell överanpassning samt valet av lämpliga beslutströsklar.

3.5.9 Export av modellen

Den slutligt tränade modellen sparades i HDF5-format för att möjliggöra framtida användning och reproducerbarhet. Genom att exportera den tränade modellen kan den laddas och testas utan att behöva tränas om från början.

3.5.10 Rapportering

Modellens prestanda sammanställdes med hjälp av etablerade utvärderingsmått, inklusive EER, min t-DCF, noggrannhet, precision, recall och F1-poäng. En fullständig klassificeringsrapport genererades för att analysera modellens beteende för respektive klass.

4. Resultat

I detta kapitel presenteras resultaten för de fyra modifierade djupinlärningsmodeller som utvärderats i studien: VGG-inspirerad CNN, LSTM, GRU och BiGRU. Samtliga modeller har tränats, validerats och testats med hjälp av ASVspoof 2019-LA.

Resultaten presenteras med både tröskeloberoende utvärderingsmått, i form av Equal Error Rate (EER) och minimum normalized tandem Detection Cost Function (min-tDCF), samt klassiska klassificeringsmått såsom noggrannhet, precision, recall och F1-poäng.

För att förbättra läsbarheten och möjliggöra en tydligare jämförelse mellan modellerna sammanfattas de viktigaste testresultaten i tabeller. Därefter följer en mer detaljerad beskrivning av varje modell, där resultaten illustreras och förklaras med hjälp av figurer.

I figurerna i detta kapitel representerar värden mellan 0 och 1 andelar av prover i textmängden. Till exempel anger noggrannhet (accuracy) andelen korrekt klassificerade röstprover. På motsvarande sätt beskriver precision hur stor andel av de identifierade positiva proverna som är korrekta, medan recall visar hur stor andel av de verkliga positiva proverna som modellen lyckas identifiera. I träningsfigurerna visar x-axeln antalet träningsepoker, medan y-axeln visar olika utvärderingsmått, såsom förlust, noggrannhet, EER och min-tDCF. dessa mått används för att följa hur modellernas prestanda utvecklas under träningen.

4.1 Sammanfattande testresultat

Tabell 4.1 ger en sammanfattning av resultaten på den oberoende testuppsättningen för samtliga modeller. Tabellen ger en översiktlig bild av modellernas prestanda i relation till varandra och utgör underlag för den jämförande analys som presenteras i kapitel 5.

GRU och BiGRU uppvisar de lägsta EER-värdena (0,0182), vilket indikerar en god förmåga att skilja mellan äkta och manipulerade röstprover. BiGRU uppnår dessutom den högsta testnoggrannheten (0,9848) och det högsta F1-värdet (0,9882), vilket visar en balanserad klassificeringsprestanda. LSTM uppvisar det högsta EER-värdet (0,0424), trots goda träningsresultat. Detta indikerar en mer begränsad generaliseringsförmåga jämfört med övriga modeller.

Dessutom presenteras en mer detaljerad redovisning av klassificeringsutfallen i tabell 4.2. Tabellen visar fördelningen mellan korrekt klassificerade prover samt antalet falska godkännanden och falska avvisningar för respektive modell baserat på confusion matrices ($n = 461$).

Tabell 4.1. Sammanfattning av testresultat för samtliga modeller (ASVspooft 2019-LA).

Modell	EER	min-tDCF	Noggrannhet	Precision	Recall	F1-poäng
VGG-inspirerad	0,0242	0,0236	0,9718	0,9609	0,9966	0,9784
LSTM	0,0424	0,0388	0,9631	0,9827	0,9595	0,9709
GRU	0,0182	0,0214	0,9761	0,9897	0,9730	0,9813
BiGRU	0,0182	0,0235	0,9848	0,9898	0,9865	0,9882

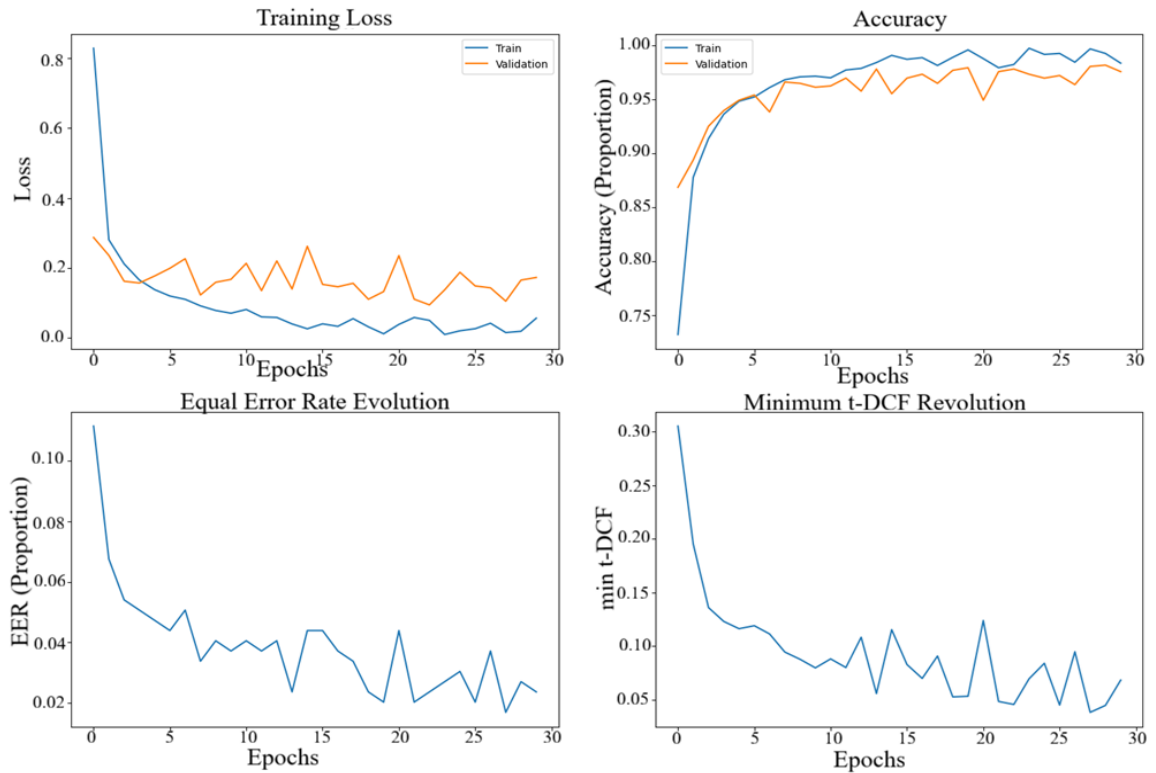
Tabell 4.2. Felutfall baserat på confusion matrix (n = 461).

Modell	Korrekt spooft	Korrekt äkta	Falska godkännanden	Falska avvisningar
VGG-inspirerad	153	295	12	1
LSTM	160	284	5	12
GRU	162	288	3	8
BiGRU	162	292	3	4

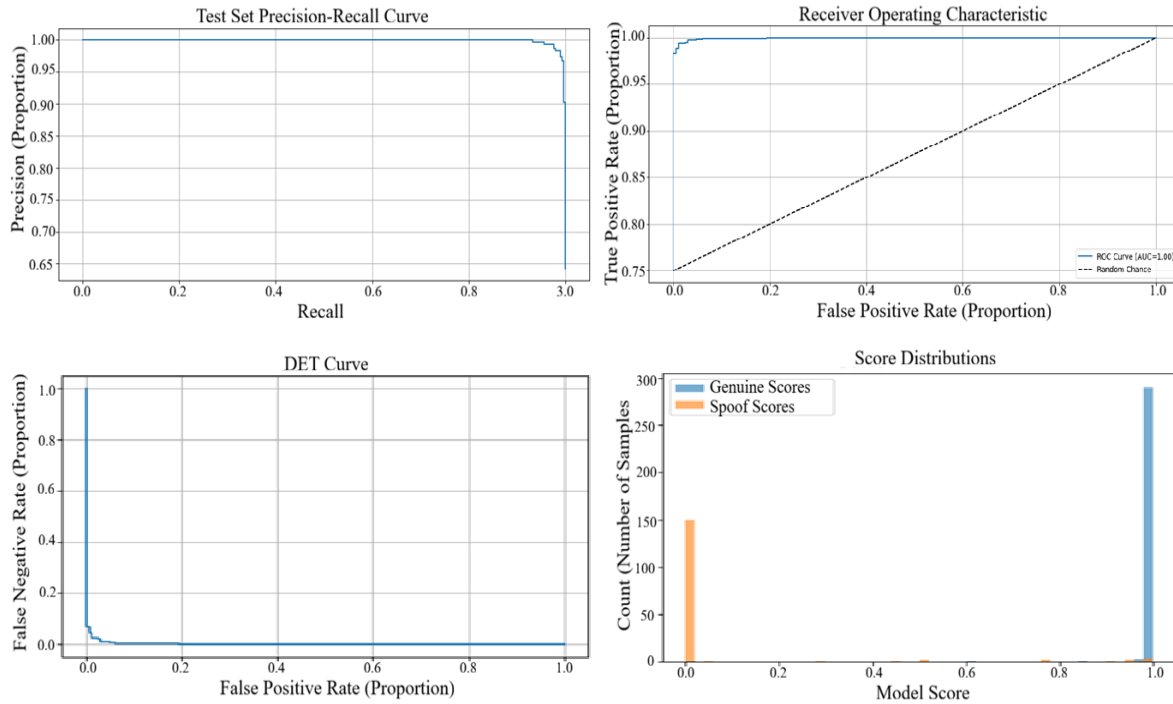
4.2 VGG-inspirerad CNN

Den modifierade VGG-inspirerade modellen utvärderades med avseende på tränings-, validerings- samt den oberoende testuppsättningen. Figur 4.1 visar modellens tränings- och valideringsresultat. I figurens övre delar framgår att både tränings- och valideringsförlusten minskar successivt över epokerna (30/30), samtidigt som noggrannheten ökar. Kurvorna följer varandra relativt väl, vilket tyder på att modellen lär sig relevanta mönster utan tydliga tecken på överanpassning. De nedre delarna av figur 4.1 visar utvecklingen av Equal Error Rate (EER) och min-tDCF under träningen. Båda måtten minskar gradvis och stabiliseras mot slutet av träningsprocessen, vilket indikerar en förbättrad förmåga att skilja mellan äkta och förfälskade röstprover. Resultaten på den oberoende testuppsättningen sammanfattas i tabell 4.1.

Figur 4.2 presenterar modellens utvärderingskurvor. Figuren innehåller flera utvärderingskurvor, inklusive ROC-kurvan, som beskriver sambandet mellan sann positiv frekvens och falsk positiv frekvens. Den visar även precision–återkallningskurvan för spoof-klassen. Vidare presenteras en DET-kurvan, där avvägningen mellan falska godkännanden och falska avvisningar framgår. Slutligen visas poängfördelningen för äkta respektive spoof-prover och illustrerar en tydlig separation mellan de två klasserna.



Figur 4.1. Tränings- och valideringsresultat för den modifierade VGG-inspirerade modellen.

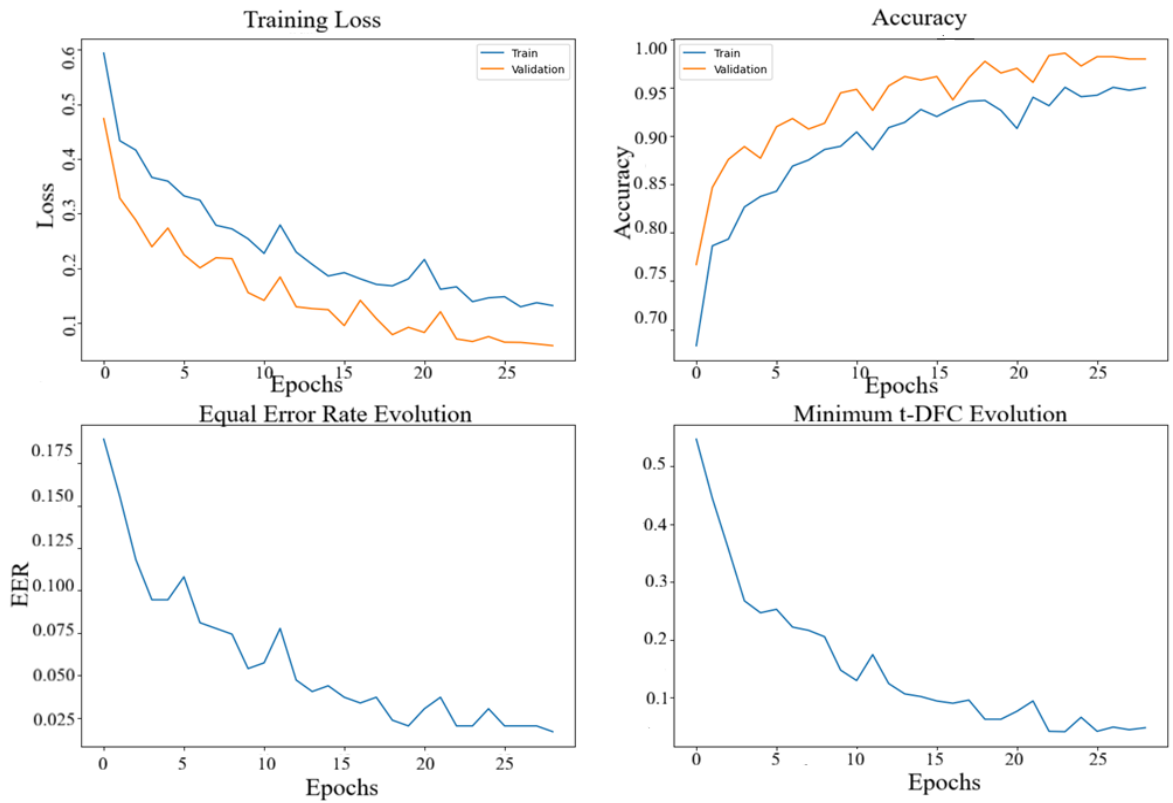


Figur 4.2. Utvärderingskurvor för den modifierade VGG-inspirerade modellen.

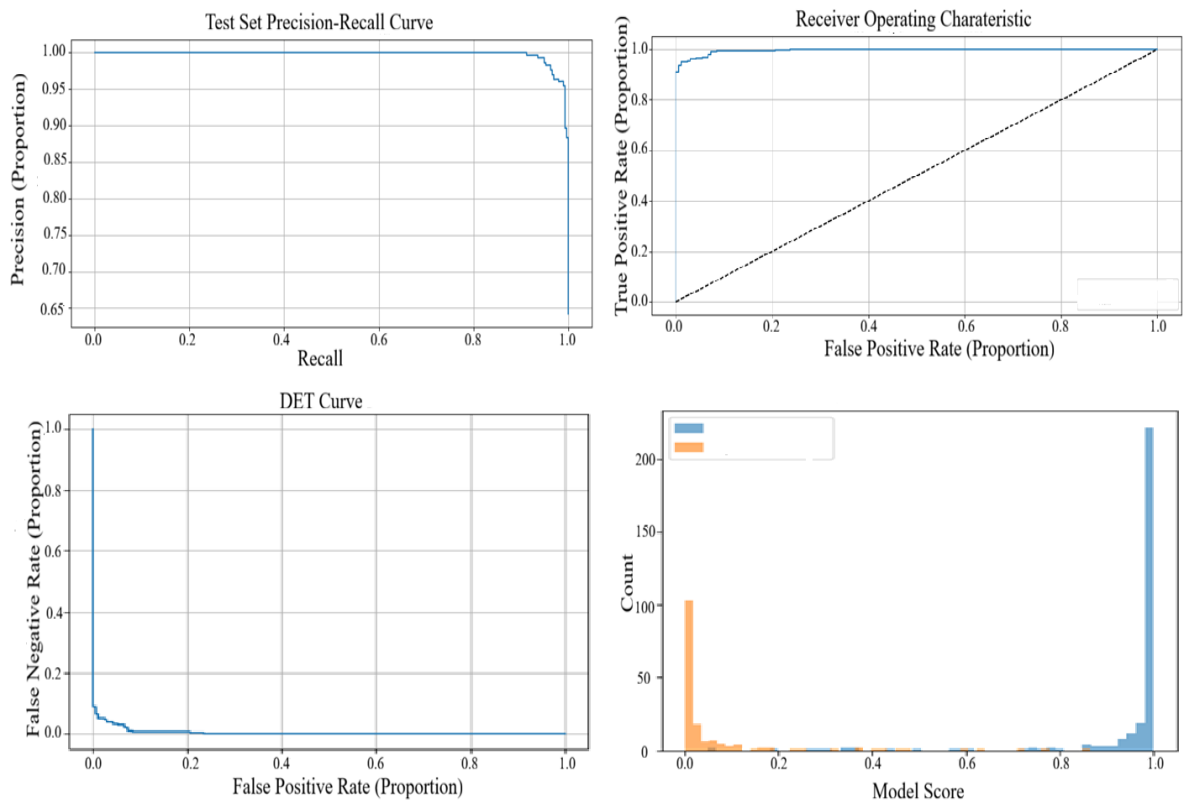
4.3 LSTM

Den modifierade LSTM-modellen uppvisade god prestanda under både träning och validering. Figur 4.3 visar tränings- och valideringsresultaten för modellen (29/30 epoker). I figurens övre delar framgår att förlusten minskar samtidigt som noggrannheten ökar, särskilt för träningsdatan. Skillnaden mellan tränings- och valideringskurvorna är dock något större än för flera av de övriga modellerna, vilket kan tyda på en begränsad generaliseringsförmåga. De nedre delarna av figur 4.3 visar att Equal Error Rate (EER) och min-tDCF förbättras under träningen, men planar ut relativt tidigt i träningsprocessen.

På den oberoende testuppsättningen presterade modellen svagare jämfört med övriga modeller, vilket framgår av tabell 4.1 där modellen uppvisar det högsta EER-värdet. Figur 4.4 presenterar utvärderingskurvorna för LSTM-modellen. Kurvorna visar att separationen mellan äkta och spoof-prover är mindre tydlig än för de övriga modellerna, särskilt i poängfördelningen, vilket återspeglar modellens sämre förmåga att skilja mellan klasserna på testdata.



Figur 4.3. Tränings- och valideringsresultat för den modifierade LSTM-modellen.

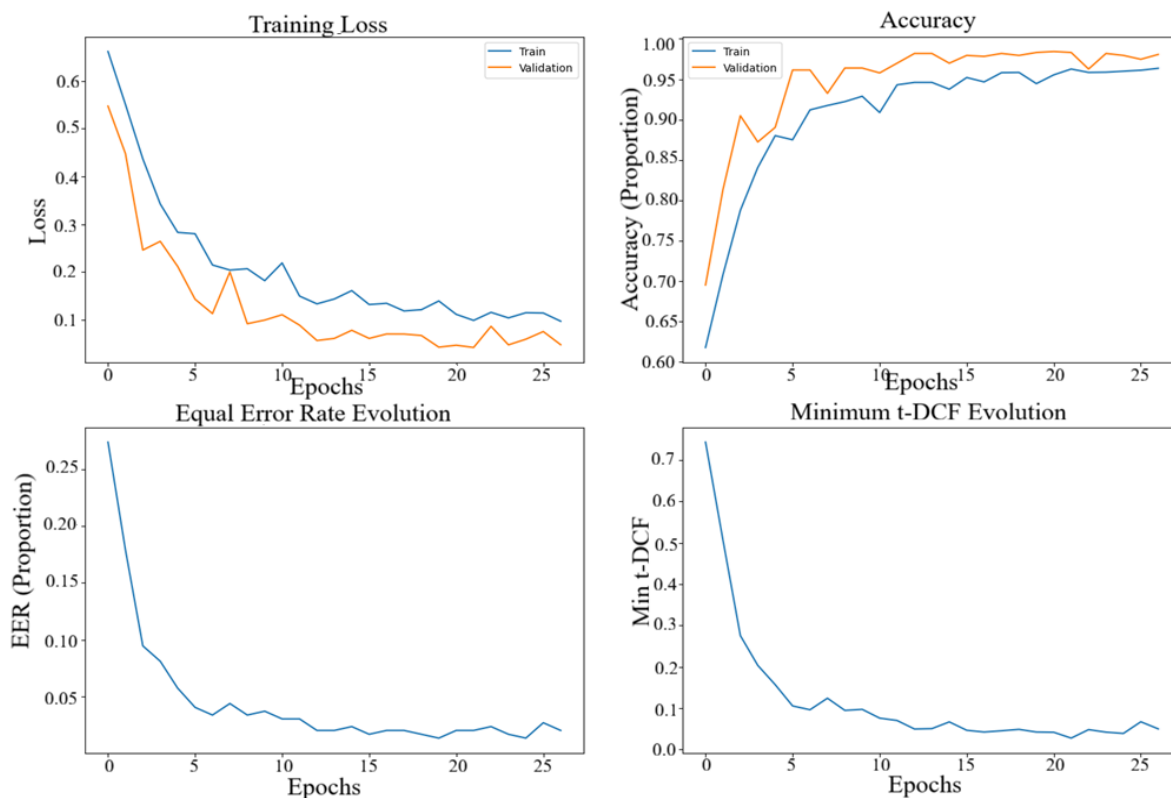


Figur 4.4. Utvärderingskurvor för den modifierade LSTM-modellen.

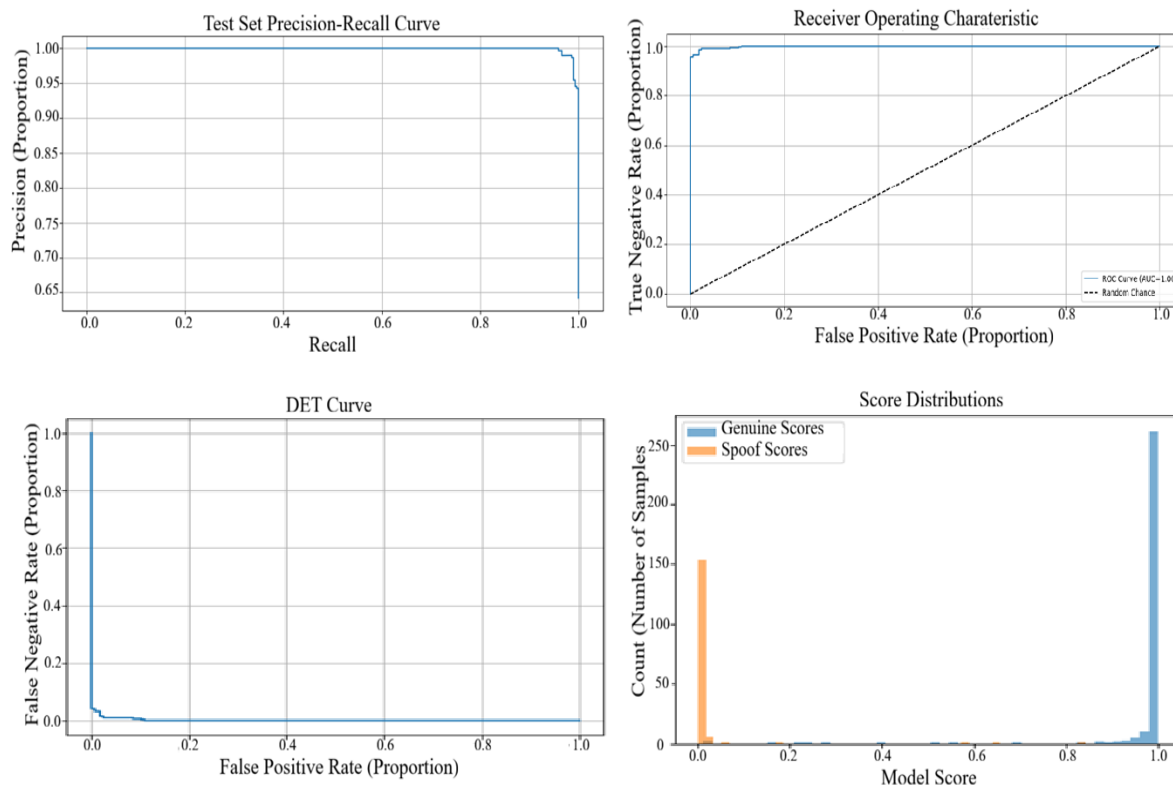
4.4 GRU

Den modifierade GRU-modellen tillhörde de bäst presterande modellerna i studien. Figur 4.5 visar modellens tränings- och valideringsresultat (29/30 epoker). I figurens övre delar framgår en snabb och stabil minskning av förlusten, samtidigt som noggrannheten tidigt når höga nivåer. De nedre delarna av figur 4.5 visar att både Equal Error Rate (EER) och min-tDCF når mycket låga värden och stabiliseras tidigt under träningsprocessen, vilket indikerar en effektiv inläring.

Testresultaten i tabell 4.1 bekräftar modellens höga prestanda, där modellen uppvisar låga EER- och min-tDCF-värden samt hög precision och recall. Figur 4.6 presenterar utvärderingskurvorna för GRU-modellen. Kurvorna visar en tydlig och konsekvent separation mellan äkta och spoof-prover, vilket ytterligare bekräftar modellens starka prestanda på den oberoende testuppsättningen.



Figur 4.5. Tränings- och valideringsresultat för den modifierade GRU-modellen.



Figur 4.6. Utvärderingskurvor för den modifierade GRU-modellen.

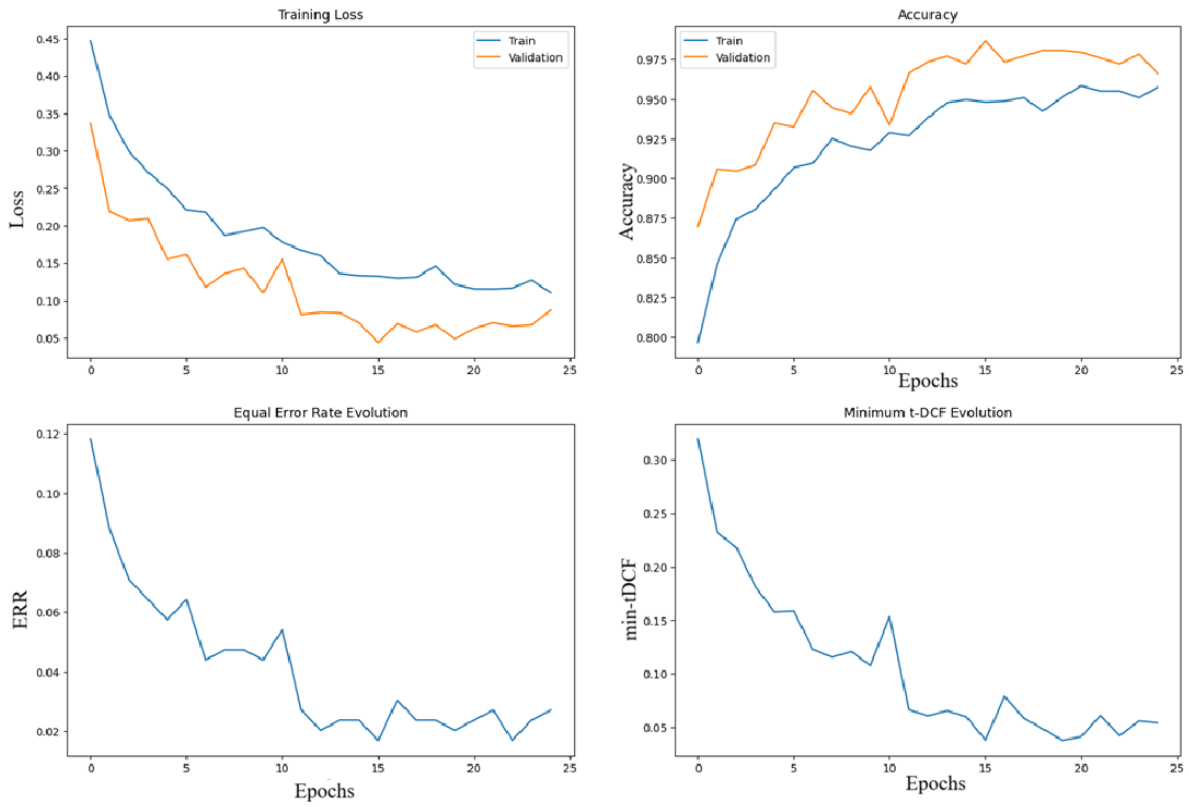
4.5 BiGRU

Den modifierade BiGRU-modellen uppvisade den högsta totala testnoggrannheten av samtliga modeller.

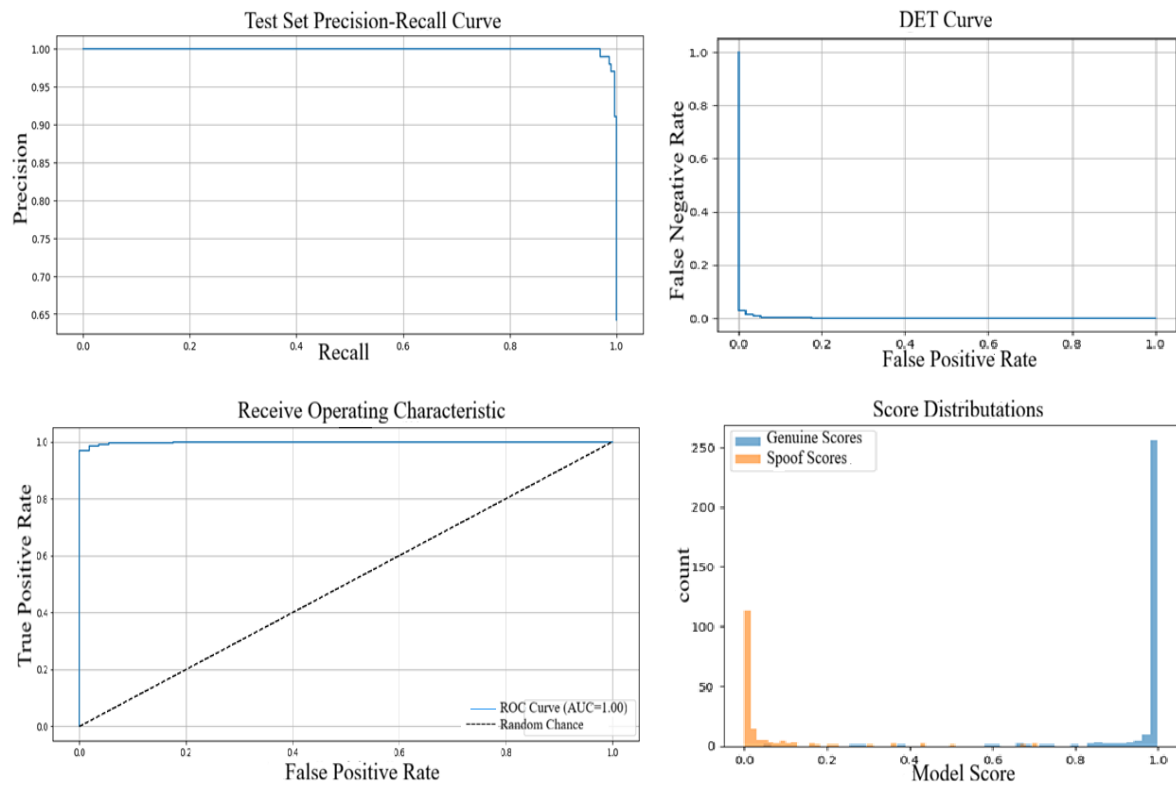
Figur 4.7 visar tränings- och valideringsresultat för modellen (25/30 epoker). De övre delarna av figuren visar stabil konvergens med låg förlust och hög noggrannhet för både tränings- och valideringsdata. De nedre delarna av figur 4.7 visar att EER och min-tDCF når låga och stabila värden under träningen.

Resultaten i tabell 4.1 visar att modellen kombinerar låg felnivå med hög precision och recall. Figur 4.8 presenterar utvärderingskurvor för BiGRU-modellen. Kurvorna visar en tydlig separation mellan äkta och spoof-prover, vilket indikerar att modellen har en stark förmåga att skilja mellan de två klasserna.

Figur 4.7. Tränings- och valideringsresultat för den modifierade BiGRU-modellen.



Figur 4.7. Tränings- och valideringsresultat för den modifierade BiGRU-modellen.



Figur 4.8. Utvärderingskurvor för den modifierade BiGRU-modellen.

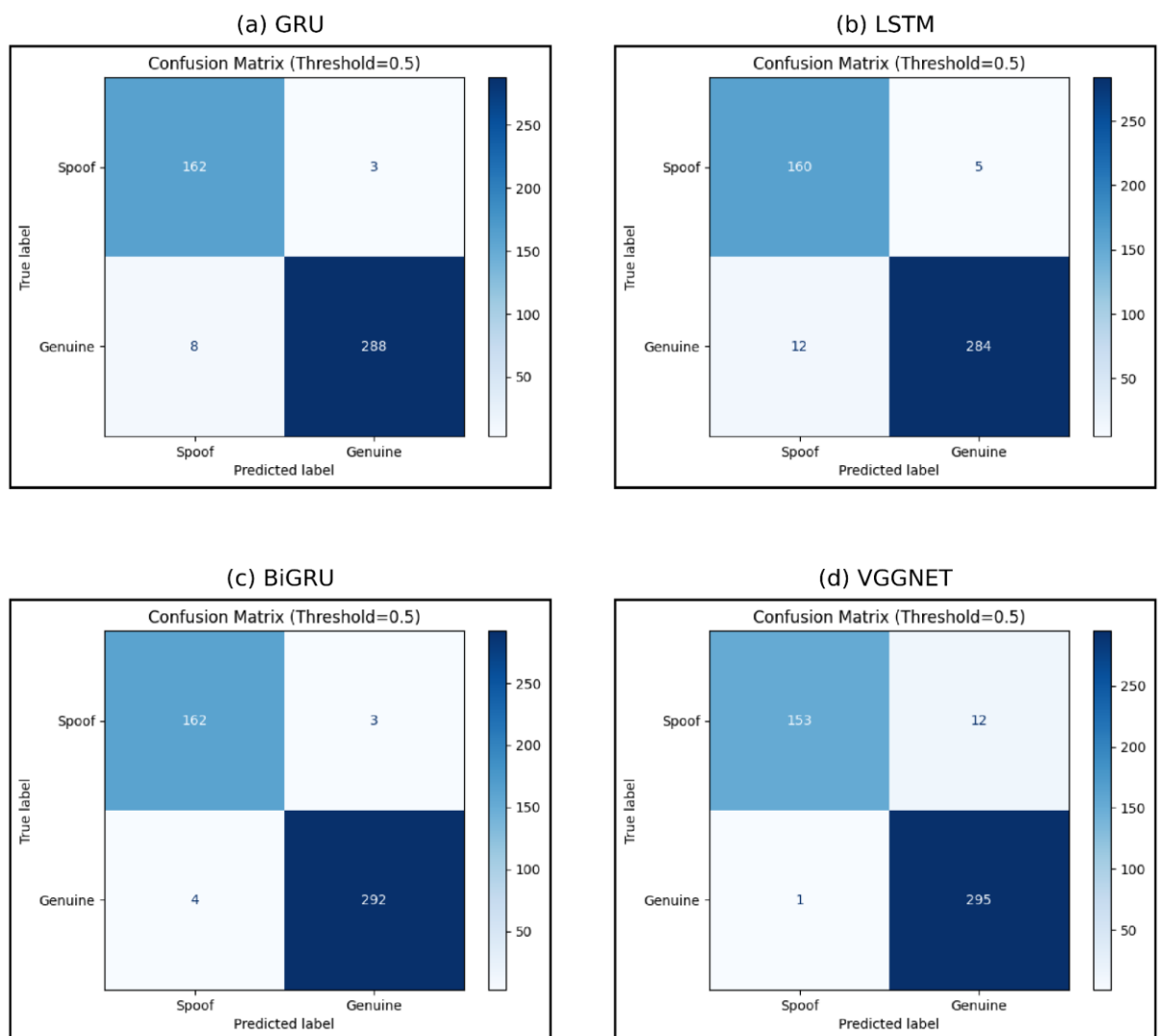
4.6 Fördelning av klassificeringsfel

Figur 4.9 visar confusion matrices för samtliga modeller på den oberoende testuppsättningen ($n = 461$). Fördelningen av fel redovisas även i tabell 4.2.

Den VGG-inspirerade modellen klassificerar en stor andel av testproverna korrekt, men uppvisar fler falska godkännanden jämfört med de rekurrenta modellerna. LSTM uppvisar det högsta antalet falska avvísningar (12), vilket bidrar till den lägre testprestandan.

GRU och BiGRU uppvisar det lägsta antalet falska godkännanden (3). BiGRU visar dessutom få falska avvísningar.

En mer detaljerad tolkning av dessa skillnader presenteras i kapitel 5.



Figur 4.9. Confusion matrices för de modifierade modellerna (BiGRU, VGG-inspirerad CNN, GRU och LSTM) baserat på den oberoende testuppsättningen ($n = 461$).

5. Diskussion

I det här avsnittet läggs en jämförande analys fram av de fyra modeller som man gått igenom för att upptäcka förfalskningar i enlighet med ASVspooft 2019-standarderna. Modellerna inkluderar ett VGG-inspirerat Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) och Bidirectional GRU (BiGRU). Diskussionen grundas på flera prestandaindikatorer: Equal Error Rate (EER), minimum normalized tandem Detection Cost Function (min t-DCF), noggrannhet, precision, träffsäkerhet, F1-poäng, confusion matrix härledda felmönster.

5.1 Jämförelse mellan EER och min-tDCF

Equal Error Rate (EER) och minsta normaliserade min-tDCF är viktiga utvärderingsmått i ASVspooft-utmaningen, för att påvisa balansen mellan säkerhet och användbarhet i talverifieringssystem. GRU- och BiGRU-modellerna fick det lägsta test-EER på 0,0182, vilket visar att de har förmågan att skilja på manipulerade och äkta försök. LSTM-modellen hade den högsta EER på testuppsättningen (0,0424), trots stark tränings- och valideringsprestanda, vilket kan tyda på problem med generalisering. När det kommer till min-tDCF, överträffade GRU-modellen återigen andra med ett värde på 0,0214, följt av BiGRU (0,0235). Den VGG-inspirerade modellen gjorde också bra ifrån sig konkurrenskraftigt med en min-tDCF på 0,0236, vilket stärkte dess effektivitet trots att den bestod av en faltningsarkitektur, det vill säga en konvolutionell arkitektur. Samtliga dessa resultat sammanfattas i tabell 4.1.

Även om skillnaderna i EER och min-tDCF mellan vissa modeller är numeriskt små, kan de vara av stor betydelse i praktiska tillämpningar. I storskaliga system, till exempel inom banktjänster eller kundsupport, kan även marginella förbättringar leda till färre felaktiga beslut och därmed ökad säkerhet. Resultaten tyder därför på att modeller som GRU och BiGRU, trots små numeriska skillnader jämfört med övriga modeller, kan erbjuda en tydlig praktisk fördel i verkliga användningssammanhang.

5.2 Analys av klassificeringsmått

När modellerna utvärderades vid ett beslutströskelvärde på 0,5 fick alla de höga värden i standardklassificeringsmått, vilket framgår av resultaten i tabell 4.1. BiGRU-modellen fick den högsta totala testnoggrannheten på 0,9848, med ett utmärkt F1-poäng på 0,9882, en precision på 0,9898 och en träffsäkerhet på 0,9865, vilket visar på ett bra och säkert klassificeringsbeteende. LSTM-modellen, med hög precision (0,9827), visade en minskad återkallelse (0,9595), något som resulterade i ett högre antal falska avslag. GRU-modellen fick den högsta testprecisionen (0,9897), med en stark återkallelse (0,9730), något som tyder på en låg andel falsk positiv frekvens och därmed en stabil och tillförlitlig acceptans av äkta försök. Skillnaderna i precision och recall mellan modellerna belyser hur väl de balanserar säkerhet och användarvänlighet. En hög precision, som hos GRU- och BiGRU-modellerna, innebär att andelen falska godkännanden är låg, vilket är särskilt viktigt i säkerhetskritiska

tillämpningar. Samtidigt visar LSTM-modellens lägre recall att fler äkta försök felaktigt avvisas, vilket kan ha en negativ inverkan på användarupplevelsen. Sammantaget indikerar resultaten att modeller med en mer balanserad kombination av precision och recall, såsom GRU och BiGRU, är bättre lämpade för praktiska autentiseringssystem.

5.3 Insikter om confusion matrix

Confusion matrix visar olika typer av fel som görs av varje modell, vilket är relevant för att förstå hur resultaten påverkar praktisk användning (se tabell 4.2).

BiGRU- och GRU-modellerna visade det lägsta antalet falska godkännanden (3), vilket innebär att dessa modeller mycket sällan klassificerar ett förfalskat ljud som äkta. Detta är särskilt viktigt i säkerhetskritiska tillämpningar, där ett falskt godkännande kan leda till obehörig åtkomst. BiGRU uppvisade också lågt antal falska avslag (4), något som bekräftar dess robusthet. Däremot visade LSTM-modellen ett rätt högt antal falska avslag (12), och detta påverkade dess användbarhet i verkliga system, där frekventa felaktiga avvisningar riskerar att skapa frustration hos användarna. Trots god precision indikerar detta att modellen har svårigheter att balansera säkerhet och användbarhet, enligt dess högre EER-värde (se tabell 4.2).

Sammanfattningsvis visar analysen av confusion matrix, att modeller som GRU och BiGRU har ett mer balanserat felmönster, med både låg risk för säkerhetskritiska fel och begränsad negativ påverkan på användarupplevelsen. Detta stärker deras lämplighet för verkliga röstbaserade autentiseringssystem.

5.4 AUC-, DET- och poängfördelningar

Alla modeller fick AUC-poäng på 0,99 för både ROC- och precisionsåterkallningskurvor, vilket visar på en stark förmåga över tröskelvärden. DET-kurvorna (Detection Error Tradeoff) för alla modeller visade konsekventa prestanda, där BiGRU och GRU visade tydligare separation vid låga falska positiva och falska negativa frekvenser. Poängfördelningsdiagrammen för dessa modeller visade en tydlig separation mellan förfalskning och äkta poäng. Modeller som GRU och BiGRU uppvisade en mindre överlappning i poängområden, vilket stöder deras överlägsna EER- och t-DCF-prestanda.

Den tydliga separationen i både DET-kurvor och poängfördelningar innebär att dessa modeller i mindre grad hamnar i osäkra gränfall, där små förändringar i beslutströskeln kan ge stora skillnader i felutfall. Detta är särskilt viktigt i praktiska autentiseringssystem, där stabila och förutsägbara beslut bidrar till både högre säkerhet och bättre användarupplevelse.

5.5 Sammanfattning av iakttagelserna

1. GRU och BiGRU överträffade andra modeller när det kommer till både EER och min-tDCF, något som gör dem till de mest lämpliga kandidaterna för anti-spoofing-implementering.

2. BiGRU-modellen fick den högsta testnoggrannheten och F1-poängen, vilket indikerar en på en balanserad prestanda över alla utvärderingsmått.
3. CNN-baserade (VGG-inspirerade) modellen gjorde också ifrån sig konkurrenskraftigt, vilket visar på att tidsseriemodellering inte är den enda effektiva metoden.
4. Trots starka träningsprestationer visade LSTM en svagare generalisering, något som visade på behovet av bättre regularisering eller justering.
5. Alla modeller fick höga AUC-poäng, och detta bekräftar deras tillförlitlighet över ett brett spektrum av tröskelvärden.

Sammanfattningsvis framgår det av resultaten att modellerna skiljer sig åt i hur väl de balanserar prestanda, generaliseringsförmåga och modellkomplexitet. GRU- och BiGRU-modellerna presterade genomgående väl på den oberoende testuppsättningen och uppvisade stabila felmönster, vilket indikerar en god förmåga att generalisera till nya data. Detta gör dem mer användbara vid praktisk implementering i röstbaserade autentiseringssystem där både säkerhet och tillförlitlighet är centrala krav.

De bidirektionella modellerna, i synnerhet BiGRU, visade fördelaktighet av möjligheten att analysera ljudsekvenser i både framåt- och bakåtriktning. Detta verkar ha bidragit till en bättre separation mellan äkta och manipulerade röster. Samtidigt medför denna typ av modell ett högre beräknings- och minnesbehov jämfört med enkelriktade modeller, vilket kan utgöra en nackdel i system med begränsade resurser, exempelvis på mobila eller inbyggda plattformar.

Den konvolutionella modellen VGG-inspirerade CNN-modellen visade att konkurrenskraftiga resultat kan uppnås även utan rekurrenta strukturer. Modellen har en enklare arkitektur och kan därför erbjuda snabbare beslutfattande vid användning, vilket är fördelaktigt i realtidsapplikationer. Däremot kan dess begränsade förmåga att fånga långsiktiga tidsberoenden göra den mindre effektiv vid mer avancerade spoofing-attacker.

LSTM-modellen uppvisade trots goda träningsresultat en svagare prestanda på testuppsättningen, vilket indikerar en mer begränsad generaliseringsförmåga i sammanhanget. Det indikerar på att modellen kan vara känslig för överanpassning och att ytterligare åtgärder, såsom mer omfattande regularisering eller dataförstärkning, kan krävas för att förbättra dess robusthet.

5.6 Modellernas beräkningskrav och praktisk användbarhet

De modeller som användes i denna studie valdes inte enbart utifrån uppnådd prestanda, utan även med hänsyn till beräkningskrav och praktisk användbarhet i system med begränsade resurser, såsom mobila enheter eller inbyggda system. Detta utgjorde en viktig utgångspunkt vid valet av arkitekturer och motiverar användningen av relativt kompakta djupinlärningsmodeller framför mycket stora och komplexa nätverk.

GRU- och LSTM-baserade modeller har generellt färre parametrar än mer komplexa djupa nätverk och kräver därmed mindre minne och lägre beräkningskapacitet. Särskilt GRU-modellen, som uppvisade stark prestanda trots sin enklare struktur, framstår som ett attraktivt alternativ för realtidsapplikationer där låg latens och energieffektivitet är avgörande faktorer.

Den bidirektionella modellen BiGRU uppnådde i flera fall bättre resultat än den enkelriktade GRU-modellen, men detta sker på bekostnad av ökade beräknings- och minneskrav. Denna typ av modell är därför mer lämpad för serversystem eller offline-analys, där tillgången till beräkningsresurser är mindre begränsad.

Den VGG-inspirerade konvolutionella modellen representerar en alternativ avvägning mellan prestanda och effektivitet. Modellen har ofta snabbare beräkning vid användning och lämpar sig väl för parallellisering, vilket gör den attraktiva i realtidsnära system. Samtidigt kan den begränsade förmågan att fånga långsiktiga tidsberoenden innebära sämre prestanda vid mer komplexa spoofing-attacker jämfört med rekurrenta modeller.

Sammantaget visar resultaten att det finns tydliga avvägningar mellan prestanda, generaliseringsförmåga och beräkningskostnad. Valet av modell bör därför anpassas efter det specifika användningsområdet, där faktorer som tillgänglig hårdvara, svarstid och energiförbrukning vägs mot krav på säkerhet och noggrannhet.

5.7 Relatering till tidigare forskning

Resultaten i denna studie kan relateras till tidigare forskning inom detektion av röstspoofing, där djupinlärningsbaserade modeller såsom konvolutionella och rekurrenta neurala nätverk visat god prestanda på ASVspooft-dataset [8]. ASVspooft 2019-LA används ofta som benchmark för utvärdering av anti-spoofing-system [9], [22], och tidigare studier har visat att både konvolutionella och sekvensbaserade modeller kan uppnå låg Equal Error Rate (EER) inom denna experimentella miljö.

De resultat som uppnåtts i detta arbete, särskilt för GRU- och BiGRU-modellerna, ligger i linje med denna forskningsriktning och bekräftar att relativt kompakta djupinlärningsmodeller kan uppnå god detekteringsförmåga under kontrollerade förhållanden.

Till skillnad från vissa mer resurskrävande arkitekturer har fokuset här legat på modeller med lägre beräkningskostnad. Detta visar att effektiv spoofing-detektion inte nödvändigtvis kräver mycket stora eller komplexa nätverk, vilket är relevant i sammanhang där beräkningsresurser är begränsade.

Sammantaget placerar sig denna studie inom ramen för befintlig forskning och bidrar med en systematisk jämförelse av flera resurseffektiva modeller med samma experimentella upplägg.

5.8 Begränsningar och val av dataset

I denna studie användes ASVspoof 2019-LA-datasetet för både träning och utvärdering av modeller. Detta gjordes för att hålla arbetet avgränsat och genomförbart inom ramen för examensarbetet. ASVspoof 2019 är ett välkänt och ofta använt dataset inom området, vilket gör det lämpligt för att jämföra olika modeller på ett tydligt sätt.

Nyare dataset, såsom ASVspoof 2021, innehåller mer varierade och realistiska attackscenarier, men är också mer omfattande och tidskrävande att arbeta med. Av tidsskäl och med fokus på att jämföra flera olika djupinlärningsmodeller valdes därför ASVspoof 2019. Detta innebär att resultaten främst visar hur modellerna presterar inom samma dataset, och att utvärdering på nyare dataset kan ses som framtida arbete.

5.9 Påverkan på samhällsnivå

Att använda stabila och trovärdiga tekniker mot just spoofing kan ha olika effekter på flera sektorer och även påverka den finansiella stabiliteten, operativa metoder, teknisk innovation och även den globala cybersäkerheten. I detta avsnitt beskrivs de bredare ekonomiska effekterna på samhällsnivå av att ha effektiva system för att kunna upptäcka röstförfalskning.

5.9.1 Kostnadsbesparingar och riskreducering

En av fördelarna med ett system för upptäckt av röstförfalskning är minskningen av ekonomiska förluster samt en reduktion av operativ ineffektivitet.

- **Förebyggande av bedrägerier:** Autentiseringssystem som är röstbaserade används i allt högre grad inom dessa områden: bank, kundsupport, hälso- och sjukvård och andra känsliga områden. Dessa system är väldigt utsatta och känsliga för förfalskningsattacker som görs med hjälp av syntetiska eller konverterade röster. Attacker som faktiskt lyckas kan leda till obehörig åtkomst till väldigt känslig information, identitetsstöld och ekonomiska bedrägerier. BiGRU-modellen som utvärderades i denna studie, kan minska dessa risker. Effektiva detekteringssystem kan potentiellt bidra till att minska ekonomiska förluster kopplade till röstbaserade bedrägerier.
- **Operativ effektivitet:** Vanliga processer för identitetsverifiering, såsom manuell granskning av transaktioner eller röstprover som kan misstänkas vara felaktiga, är tidskonsumerande och dessa granskningar är även sårbara för mänskliga fel. Modeller som är automatiserade för upptäckt av förfalskningar minskar behovet av mänskligt arbete i verifieringsarbetsflöden, vilket gör det möjligt för organisationer att kunna effektivisera verksamheten, sköta personal på ett effektivare sätt och även minska på driftskostnaderna.
- **Kundernas förtroende och bevarande:** Förtroende är en väldigt viktig del i alla autentiseringssystem. Användare har lättare för att interagera med och använda system som de ser som säkra och trovärdiga. Integrering av förfalskningsdetekteringsmodeller förbättrar systemets integritet och även

förtroendet för röstbaserade autentiseringssystem, vilket kan ha positiv påverkan på användarupplevelsen.

5.9.2 Marknadsspecifika tillämpningar samt fördelar

Användningen av teknik för upptäckt av röstförfalskning kan ha stor inverkan inom flera sektorer med höga risker och ett högt värde.

- **Finansiella tjänster:** Banker och finansinstitut är intressanta mål för attacker som är röstbaserade, speciellt då det är så att de oftare använder biometrisk autentisering för online- och mobilbanker. Stabila åtgärder mot förfalskning kan skydda kundkonton, tillgångar och se till att efterlevnad av finansiella bestämmelser som PSD2 (Payment Services Directive) och KYC-protokoll (Know Your Customer) sköts. Dessa skydd är viktiga för användarsäkerheten, samt hjälper att undvika juridiskt ansvar och spridning av rykten.
- **Telekommunikation:** Telekommunikationsleverantörer hanterar en stor mängd med rösttrafik och är speciellt mottagliga för förfalskningsbaserade bedrägerier, såsom robotsamtal, nätfiske och SIM-bytesbedrägerier. Genom att ha dessa modeller för identifiering av förfalskningar i telekominfrastrukturen kan man förbättra säkerheten för tjänster som VoIP-autentisering, röstmeddelanden och kundsupport, samtidigt som man då minskar antalet bedrägerier och även förbättrar servicekvaliteten.
- **Konsumentelektronik och IoT-enheter:** Spridningen av smarta hemenheter, inklusive röststyrda assistenter (t.ex. Amazon Alexa, Google Assistant), skapar nya attackytor för spoofing-hot. En angripare kan använda syntetiska kommandon för att kunna låsa upp dörrar, komma åt personuppgifter eller manipulera enheter som är anslutna. Genom att implementera beräkningsmässigt effektiva och resursbegränsade spoofing-detekteringsmodeller direkt i dessa enheter kan produktsäkerheten och användarnas förtroende förbättras, vilket ökar marknadsanvändningen samt tillförlitligheten hos enheten.

5.9.3 Bredare ekonomiska och samhällliga konsekvenser

Om man bortser från de uppenbara fördelarna för organisationer och användare bidrar utvecklingen och användningen av detekteringssystem för röstförfalskning även till en allt bredare ekonomisk tillväxt och även digital global säkerhet.

- **Sysselsättning samt kompetensutveckling:** Efterfrågan på AI-baserade säkerhetslösningar ökar alltmer i samhället och det skapar nya jobbområden inom flera olika områden, såsom datavetenskap, maskininlärning, ljudkriminalteknik, cybersäkerhet och mjukvaruteknik. Utbildningsinstitutioner och utbildningsprogram kan även utvecklas för att stödja det alltmer växande behovet av yrkesverksamma och kompetenta inom talbehandling och AI-etik.
- **Innovation och tekniska framsteg:** Det finns fördelar med investeringar i detektering av röstförfalskning, då det uppmuntrar till teknisk innovation inom områden som gränsar till dessa. Framsteg inom djupinlärningsarkitekturer för förfalskningsdetektering kan t.ex. inspirera till förbättringar av känsligenkänning, talsyntes, språkmodellering och även multimodal biometrisk fusion. AI-ramverk

(XAI) som utvecklats för förfalskningsdetektering görs även om till bredare användningsfall i beslutsfattande.

- **Global cybersäkerhet och dataskydd:** Röstbaserade interaktioner blir allt vanligare inom statliga tjänster, försvarskommunikation och hälso- och sjukvård, och det handlar då om nationell och global säkerhet att säkra dessa områden mot just dessa spoofing-attacker. Den ökade användningen av anti-spoofing-system bidrar till ett alltmer säkert digitalt ekosystem, vilket minskar riskerna för att felaktig information sprids, identitetsstöld och obehörig övervakning. Det följer också internationella cybersäkerhetsmål, bland annat de som nämns i EU:s dataskyddsförordning och FN:s ramverk för digitalt samarbete.

5.9.4 Sammanfattning av påverkan

Sammanfattningsvis är det så att själva integreringen av effektiva system för upptäckt av röstförfalskning, särskilt de som använder sig av djupinlärningsmodeller som GRU och BiGRU, kan ge fördelar enligt nedan:

- Förhindra ekonomiska bedrägerier från att uppstå samt minska driftskostnaderna.
- Förbättra konsumenternas förtroende, samt användningen av tjänsterna.
- Stärka säkerhetsinfrastrukturen inom finans, telekom och IoT.
- Bidrar till jobbskapande och utbildningstillväxt inom AI samt cybersäkerhet.
- Främja innovation inom områdena maskininlärning och talbearbetning.
- Bidra till nationella och globala cybersäkerhetsinitiativ.

Dessa punkter ovan belyser vikten av att utveckla högpresterande modeller, och att de implementeras på korrekt sätt, samt att skalbarhet och integration i verkliga system görs korrekt. Röstgränssnittet ökar som medel för att interagera mellan människa och dator och har blivit en teknisk nödvändighet. Det är därav ett samhällsligt ansvar att skydda individer från spoofing-försök.

6. Slutsats och fortsatt arbete

6.1 Slutsats

Denna studie presenterade en bred utvärdering av ett antal djupinlärningsarkitekturer för att kunna upptäcka förfalskningar i automatiska talarverifieringssystem, enligt ASVspooF 2019-utmaningsramverket. Fyra modeller implementerades och testades: ett VGG-inspirerat Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) och Bidirectional GRU (BiGRU). Varje modell bedömdes baserat på prestandamått såsom noggrannhet, precision, träffsäkerhet, F1-poäng, Equal Error Rate (EER), minsta normaliserade tandemdetekteringskostnadsfunktion (t-DCF), resultat av confusion matrix, AUC-poäng och DET/poängfördelningskurvor.

Resultaten visade att alla modeller kunde uppnå hög klassificeringsprestanda. De GRU-baserade modellerna, speciellt BiGRU, gjorde dock ifrån sig bättre än de andra modellerna baserat på utvärderingskriterierna. BiGRU-modellen uppnådde den lägsta test-EER (0,0182), lägsta test-t-DCF (0,0235) och den högsta noggrannheten (0,9848) och F1-poängen (0,9882) bland alla testade arkitekturer. Dessa resultat återspeglar både en låg andel felklassificeringar och en stark balans mellan falskt avvisande och genuint accepterande.

Den VGG-inspirerade CNN-modellen visade konkurrenskraftig prestanda, något som visar på att konvolutionella arkitekturer (faltningarkitekturer), även om de inte är sekvensbaserade, fortfarande är genomförbara för förfalskningsdetektering när de är korrekt skapade. LSTM-modellen gav blandade resultat. Trots hög tränings- och valideringsprestanda visade LSTM-modellen ett högre test-EER (0,0424). Detta tyder på att återkommande modeller kan kräva en noggrann regularisering och träningsstrategier för att förhindra att överanpassning sker. Ur ett systemperspektiv indikerar låga t-DCF-värden som observerats i de bästa presterande modellerna deras lämplighet för verkliga talarverifieringsdistributioner, där kostnadskänsliga fel som felaktiga godkännanden av falska försök behöver minimeras. Resultaten visar också att de modeller som använts i studien ger en bra balans och beräkningskrav jämfört med större och mer resurskrävande modeller, vilket gör dem praktiska att använda i verkliga system.

6.2 Framtida arbete

Även om resultaten är lovande finns det fortfarande ett antal möjligheter till förbättring och vidareutveckling. Framtida arbete kan utforska kombinationen av flera modellutdata, exempelvis genom att slå samman utdata från CNN- och GRU-baserade arkitekturer för att dra nytta av komplementära styrkor och ytterligare minska EER samt t-DCF.

Att använda mekanismer för självuppmärksamhet eller hybriduppmärksamhet kan förbättra tidsmodellering genom att låta nätverket fokusera alltmer på informativa ramar, speciellt i yttranden med olika längd.

Vidare kan framtida arbete fokusera på att förbättra modellens robusthet för att få bort variabilitet, brus och okända förfalskningstyper genom tekniker som domänanpassning,

kontradiktorisk träning eller dataförstärkning. Utvärdering över datauppsättningar, exempelvis testning av modeller som tränats på ASVspoof 2019 mot ASVspoof 2021 eller datauppsättningar med logisk åtkomst, kan hjälpa till att bedöma generaliseringsförmågan hos dessa metoder.

Med ökad efterfrågan på transparenta AI-system kan framtida studier få fram AI-tekniker (XAI) för att bättre förstå vilka egenskaper som bidrar mest till modellens beslut och därmed öka tillförlitligheten i spoofing-detekteringssystem.

Referenser

- [1] Y. Bengio, “Learning deep architectures for AI,” *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.
- [2] P. Domingos, *The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World*. New York, NY, USA: Basic Books, 2015.
- [3] E. I. Altman, “Financial applications of machine learning: A literature review,” *Journal of Financial Data Science*, vol. 6, pp. 1–15, 2020, doi: 10.1016/j.jfds.2020.100020.
- [4] J. Bateman, *Deepfakes and Synthetic Media in the Financial System: Assessing Threat Scenarios*. Carnegie Endowment for International Peace, 2022.
- [5] R. Khan, M. Taqi, and A. Afzal, “Deepfakes in finance: Unraveling the threat landscape and detection challenges,” in *Navigating the World of Deepfake Technology*. IGI Global, 2024, pp. 91–120.
- [6] J. W. Lee et al., “Representation selective self-distillation and wav2vec 2.0 feature exploration for spoof-aware speaker verification,” arXiv preprint arXiv:2204.02639, 2022.
- [7] S. Kalaiarasu, N. A. A. Rahman, and K. S. Harun, “Deepfake impact, security threats and potential preventions,” in *AIP Conference Proceedings*, vol. 2920, 2024.
- [8] M. Li, Y. Ahmadiadli, and X.-P. Zhang, “Audio anti-spoofing detection: A survey,” arXiv preprint arXiv:2404.13914, 2024.
- [9] J. Yamagishi et al., “ASVspoof 2019: Automatic speaker verification spoofing and countermeasures challenge evaluation plan,” *ASVspoof*, vol. 13, 2019.
- [10] Z. Wu et al., “Spoofing and countermeasures for speaker verification: A survey,” *Speech Communication*, vol. 66, pp. 130–153, 2015.
- [11] T. Kinnunen and H. Li, “An overview of text-independent speaker recognition: From features to supervectors,” *Speech Communication*, vol. 52, no. 1, pp. 12–40, 2010.
- [12] M. Todisco, H. Delgado, and N. W. Evans, “A new feature for automatic speaker verification anti-spoofing: Constant Q cepstral coefficients,” in *Proc. Odyssey*, 2016.
- [13] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [14] A. Baevski et al., “wav2vec 2.0: A framework for self-supervised learning of speech representations,” in *Advances in Neural Information Processing Systems*, vol. 33, pp. 12449–12460, 2020.
- [15] H. Tak et al., “End-to-end spectro-temporal graph attention networks for speaker verification anti-spoofing and speech deepfake detection,” arXiv preprint arXiv:2107.12710, 2021.

- [16] J. Kong, J. Kim, and J. Bae, “HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis,” in *Advances in Neural Information Processing Systems*, vol. 33, pp. 17022–17033, 2020.
- [17] E. Casanova et al., “YourTTS: Towards zero-shot multi-speaker TTS and zero-shot voice conversion for everyone,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, PMLR, 2022.
- [18] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*, vol. 4. Springer, 2006.
- [19] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.
- [20] A. v. d. Oord et al., “Wavenet: A generative model for raw audio,” arXiv preprint arXiv:1609.03499, 2016.
- [21] A. Vaswani et al., “Attention is all you need,” *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [22] X. Wang et al., “ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech,” *Computer Speech & Language*, vol. 64, p. 101114, 2020.
- [23] X. Liu et al., “ASVspoof 2021: Towards spoofed and deepfake speech detection in the wild,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 31, pp. 2507–2522, 2023.
- [24] R. Reimao and V. Tzerpos, “For: A dataset for synthetic speech detection,” in *2019 Int. Conf. Speech Technol. Human-Comput. Dialog. (SpeD)*, IEEE, 2019.
- [25] J. Frank and L. Schönherr, “WaveFake: A dataset to facilitate audio deepfake detection,” arXiv preprint arXiv:2111.02813, 2021.
- [26] D. Salvi et al., “TIMIT-TTS: A text-to-speech dataset for multimodal synthetic media detection,” *IEEE Access*, vol. 11, pp. 50851–50866, 2023.
- [27] N. M. Müller et al., “Does audio deepfake detection generalize?” arXiv preprint arXiv:2203.16263, 2022.
- [28] J. Yi et al., “ADD 2023: The second audio deepfake detection challenge,” arXiv preprint arXiv:2305.13774, 2023.
- [29] Z. Zhang et al., “FMFCC-A: A challenging Mandarin dataset for synthetic speech detection,” in *Int. Workshop Digit. Watermarking*, Springer, 2021.
- [30] H. Ma et al., “CFAD: A Chinese dataset for fake audio detection,” *Speech Communication*, vol. 164, p. 103122, 2024.
- [31] P. A. Tamayo Flórez, “Voice anti-spoofing data-set built from Latin American Spanish accents implementing voice conversion and text-to-speech techniques,” 2022.
- [32] N. M. Müller et al., “MLAAD: The multi-language audio anti-spoofing dataset,” in *2024 Int. Joint Conf. Neural Netw. (IJCNN)*, IEEE, 2024.

- [33] Z. Wu et al., “Light convolutional neural network with feature genuinization for detection of synthetic speech attacks,” arXiv preprint arXiv:2009.09637, 2020.
- [34] T. Kang et al., “Experimental study: Enhancing voice spoofing detection models with wav2vec 2.0,” arXiv preprint arXiv:2402.17127, 2024.
- [35] I. Goodfellow et al., *Deep Learning*, vol. 1. MIT Press, 2016.
- [36] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [37] K. Cho *et al.*, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” *arXiv preprint* arXiv:1406.1078, 2014.
- [38] A. Graves, S. Fernández, and J. Schmidhuber, “Bidirectional LSTM networks for improved phoneme classification and recognition,” in *Int. Conf. Artif. Neural Netw.*, Springer, 2005.

Bilagor

Bilaga A -Kodexempel 3.5

```
# =====  
# Mask-Compatible Augmentation Layers  
# =====  
class TimeMasking(tf.keras.layers.Layer):  
    def __init__(self, max_mask_length=10):  
        super().__init__()  
        self.max_mask_length = max_mask_length  
        self.supports_masking = True # Enable mask propagation  
  
    def call(self, inputs, training=None):  
        if not training:  
            return inputs  
  
        def _apply_time_mask(input_slice):  
            time_steps = tf.shape(input_slice)[0]  
            features = tf.shape(input_slice)[1]  
  
            T = tf.random.uniform(shape=[], maxval=self.max_mask_length, dtype=tf.int32)  
            T = tf.minimum(T, time_steps)  
            t0 = tf.random.uniform(shape=[], maxval=time_steps - T, dtype=tf.int32)  
  
            # Create 2D mask with broadcasting  
            mask = tf.ones((time_steps, 1), dtype=tf.float32)  
            time_mask = tf.pad(  
                tf.zeros((T, 1), dtype=tf.float32),  
                [[t0, time_steps - t0 - T], [0, 0]],  
                constant_values=1  
            )  
            return input_slice * time_mask  
  
        return tf.map_fn(_apply_time_mask, inputs)  
  
    def compute_mask(self, inputs, mask=None):  
        return mask # Pass through original mask  
  
class FrequencyMasking(tf.keras.layers.Layer):  
    def __init__(self, max_mask_length=5):  
        super().__init__()  
        self.max_mask_length = max_mask_length  
        self.supports_masking = True # Enable mask propagation  
  
    def call(self, inputs, training=None):  
        if not training:  
            return inputs  
  
        def _apply_frequency_mask(input_slice):  
            time_steps = tf.shape(input_slice)[0]  
            features = tf.shape(input_slice)[1]  
  
            F = tf.random.uniform(shape=[], maxval=self.max_mask_length, dtype=tf.int32)  
            F = tf.minimum(F, features)  
            f0 = tf.random.uniform(shape=[], maxval=features - F, dtype=tf.int32)  
  
            # Create 2D mask with broadcasting  
            freq_mask = tf.pad(  
                tf.zeros((1, F), dtype=tf.float32),  
                [[0, 0], [f0, features - f0 - F]],  
                constant_values=1  
            )  
            return input_slice * freq_mask  
  
        return tf.map_fn(_apply_frequency_mask, inputs)  
  
    def compute_mask(self, inputs, mask=None):  
        return mask # Pass through original mask
```

Kodexempel 3.5: Definition av egna Keras-lager för time masking och frequency masking, inspirerade av SpecAugment, för att skapa variation och göra modellen bättre på att generalisera.

Bilaga B -Kodexempel 3.6

```
# =====
# 3. ASVspoof-Compliant Metrics Callbacks (Unchanged)
# =====
class EERCallback(callbacks.Callback):
    def __init__(self, validation_data):
        super().__init__()
        self.X_val, self.y_val = validation_data
        self.eers = []

    def on_epoch_end(self, epoch, logs=None):
        y_pred = self.model.predict(self.X_val, verbose=0).ravel()
        fpr, tpr, _ = roc_curve(self.y_val, y_pred)
        fnr = 1 - tpr
        eer = fpr[np.nanargmin(np.abs(fnr - fpr))]
        self.eers.append(eer)
        logs["val_eer"] = eer
        print(f"Val EER: {eer:.4f}")

class TDCFCallback(callbacks.Callback):
    def __init__(self, validation_data, **params):
        super().__init__()
        self.X_val, self.y_val = validation_data
        self.C1 = (params['pi_tar'] * (params['Cmiss_cm'] - params['Cmiss_asv'] * params['Pmiss_asv'])
                  - params['pi_non'] * params['Cfa_asv'] * params['Pfa_asv'])
        self.C2 = params['Cfa_cm'] * params['pi_spoof'] * (1 - params['Pmiss_spoof_asv'])
        self.min_C = min(self.C1, self.C2)
        self.min_tdcfs = []

    def on_epoch_end(self, epoch, logs=None):
        y_pred = self.model.predict(self.X_val, verbose=0).ravel()
        fpr, tpr, thresholds = roc_curve(self.y_val, y_pred)
        fnr = 1 - tpr
        min_tdcf = min((self.C1 * (1 - tpr[i]) + self.C2 * fpr[i]) / self.min_C
                       for i in range(len(thresholds)))
        self.min_tdcfs.append(min_tdcf)
        logs["val_min_tdcf"] = min_tdcf
        print(f"Val min-tDCF: {min_tdcf:.4f}")

# Callbacks setup
eer_callback = EERCallback((X_val, y_val))
tdcf_callback = TDCFCallback((X_val, y_val), **ASV_PARAMS)

metrics_callbacks = [
    eer_callback,
    tdcf_callback,
    callbacks.EarlyStopping(
        monitor='val_min_tdcf',
        patience=5,
        mode='min',
        restore_best_weights=True
    )
]
```

Kodexempel 3.6: Kod som definierar egna callbacks för Equal Error Rate (EER) och minimum tandem Detection Cost Function (min-tDCF), vilket möjliggör domänspecifik utvärdering under träning.

Bilaga C -Repository

<https://github.com/Maryambahno/voice-spoof-Detection-with-deeplearning-models/tree/main>.

TRITA – CBH-GRU-2026:140.

Stockholm, Sweden 2026

www.kth.se