Intelligent robust radiokommunikation

ERIK AXELL, KRISTOFFER HÄGGLUND,
PATRIK ELIARDSSON, ANDREAS ANDERSSON,
ARWID KOMULAINEN OCH MARCUS KARLSSON

FOI-R--5540--SE
ISSN 1650-1942 December 2023

Erik Axell, Kristoffer Hägglund, Patrik Eliardsson,
Andreas Andersson, Arwid Komulainen och
Marcus Karlsson

Intelligent robust radiokommunikation

Titel	Intelligent robust radiokommunikation
Title	Intelligent Robust Radio Communications
Rapportnr / Report No.	FOI-R--5540--SE
Månad / Month	December
Utgivningsår / Year	2023
Antal sidor / Pages	48
ISSN	1650-1942
Kund / Customer	Försvarsmakten
Forskningsområde	Ledningsteknologi
FoT område	Ledning och MSI
Projektnr / Project No.	E515221
Godkänd av / Approved by	Christian Jönsson
Ansvarig avdelning	Telekrig

Sammanfattning

Syftet med denna rapport är att öka kunskapen om hur maskininlärning (ML) kan användas för att effektivisera radiokommunikationssystem med hänsyn till de särkrav på robusthet som krävs för militära tillämpningar, avseende exempelvis garanterad tillgänglighet och skydd mot störning. Rapporten behandlar tre områden: tillgång till träningsdata, tekniker för det fysiska lagret samt tidluckeallokering i radionät.

Rapporten visar att algoritmer baserade på neurala nätverk kan utföra demodulation i impulsrik interferensmiljö med god prestanda och lägre beräkningskomplexitet än traditionella algoritmer. Vidare, kan mottagare som utför kanalestimering, kanalutjämning och demodulering via ett neuralt nätverk göras mer robust mot störning genom att inkludera ML-attacker (eng. *adversarial attacks*) i träningen av nätverket.

Ett exempel på hur sårbarheter kan nyttjas till egen fördel ges, i vilket sändaren adderar en störning till kommunikationssignalen i syfte att försämra prestanda för en fientlig, publikt välkänd, modulationsklassificerare. Störningen undertrycks av kommunikationsmottagaren för att inte försämra bitfelshalten i mottagningen. Resultat visar att tekniken fungerar, men behöver vidareutvecklas för att ge önskad prestanda.

Träningsdata av tillräcklig mängd och kvalitet är avgörande för hur en ML-algoritm presterar. Att mäta verkliga data i den mängd som behövs är svårt, men har fördelen att fånga verkliga signalbeteenden. Syntetiska data är lättare att skapa i tillräcklig omfattning, men det är svårt att modellera signal- och kanaleffekter på ett realistisk sätt. Vid nyttjande av publika dataset kan resultat jämföras med andras resultat, men det är svårt att kontrollera hur data har skapats och därmed om de innehåller felaktigheter. För att utvidga ett dataset i situationer där tillräcklig mängd saknas, kan så kallad datautökning nyttjas, men det resulterande datasetet måste valideras med relevanta metoder.

Denna rapport studerar även resursallokering i form av tidlucketilldelning. Ett flertal arbeten har publicerats avseende kanalaccess i 5G- och 6G-nät, i vilka *reinforcement learning* lyfts fram som en lämplig teknik för dessa uppgifter. Hur dessa tekniker kan översättas till taktiska mobila radionät, i vilka schemaläggning sker distribuerat, är ett arbete som påbörjats och planeras att fortsätta i kommande forskningsprojekt.

Nyckelord: maskininlärning, AI, radiokommunikation

Abstract

The purpose of this report is to increase knowledge of how machine learning (ML) can be applied to make radio communication systems more efficient, with respect to the robustness demands on military applications, such as availability and jamming protection. The report treats three subjects: access to training data, transmitter and receiver techniques for the physical layer and time-slot allocation in radio networks.

It is shown that algorithms based on neural networks are able to perform demodulation in impulse noise environments with good performance and lower computational complexity than traditional methods. Moreover, a receiver that performs channel estimation, channel equalization and demodulation through a neural network can be made more robust to jamming by adversarial training.

An example of how weaknesses can be exploited to one's advantage is given, in which a transmitter uses an adversarial attack with the aim to impair performance of a hostile, publicly well-known, modulation classifier. The attack is suppressed by the communications receiver to counteract an increased bit error rate. Results show that the technique works, but improvements are required to achieve desired performance.

A sufficient amount and quality of training data is crucial for the performance of ML algorithms. It is challenging to create the needed amount of data by measurements, but it has the advantage of capturing actual signal characteristics. Synthetic data is easier to generate in large amounts, but it is difficult to realistically model signal and channel effects. Use of public data sets allows for comparison with the results of others, but it is difficult to check whether they contain errors. Data augmentation is a way to expand a data set with limited amount of data. The augmented data set must be validated using relevant methods.

This report also studies radio resource allocation in the form of time-slot assignment. Several publications have dealt with channel access in 5G and 6G networks, in which reinforcement learning has been emphasized as a suitable solution. How these technologies can be translated to distributed scheduling in tactical mobile networks is a task that has been initiated and is planned to continue in a future research project.

Keywords: machine learning, AI, radio communications

Innehållsförteckning

1 Inledning

Införandet av maskininlärningsbaserad teknik i trådlösa kommunikationssystem ger nya möjligheter till ökad effektivitet och förbättrat utnyttjande av tillgängliga radioresurser, men riskerar också att införa nya sårbarheter. Attacker och sårbarheter med maskininlärning (ML) har studerats för andra teknikområden, exempelvis bildbehandling och språkbehandling [1]. För radiokommunikationssystem är attacker och sårbarheter inte lika välstuderade även om det börjar komma studier inom området [2].

Särskilt fokus för arbetet med ML är effektivisering av adaptiva radiosystem med bevarad eller förbättrad robusthet jämfört med traditionell teknik. Adaptiv resursallokering och konfiguration på olika lager i komplexa kommunikationssystem, exempelvis allokering av tidluckor eller frekvensband samt förbättrad trafikprediktion för militära mobila radionät, är områden som bedöms kunna dra nytta av ML. En annan intressant militärspecifik tillämpning av ML för kommunikationssystem är utvecklingen av intelligenta tekniker för att motverka t.ex. upptäckt och störning.

Alla tekniker baserade på ML är beroende av träningsdata av tillräcklig kvalitet och kvantitet. För radiokommunikationstillämpningar saknas i stor utsträckning fritt tillgängliga dataset och de som finns är framtagna för specifika tillämpningar. Dessutom finns det säkerhets- och robusthetsskäl att inte använda publika data för utveckling av militära radiosystem. För att utveckla robust ML-baserad teknik krävs därför också kontroll över träningsdata och sannolikt förmåga att skapa egna datamängder.

1.1 Syfte

Syftet med denna rapport är att öka kunskapen om hur ML kan användas för att förbättra radiokommunikationssystem med hänsyn till kommunikationsprestanda och de särkrav på robusthet som krävs för militära tillämpningar, avseende exempelvis garanterad tillgänglighet och skydd mot störning. Krav på robusthet innefattar både de exploaterbara sårbarheter som ML-teknik riskerar att införa samt traditionella hot mot radiosystem

1.2 Frågeställningar

Målet med rapporten är att ge, åtminstone preliminära, svar på följande frågeställningar:

(I) Hur bör träningsdata av tillräcklig kvalitet och kvantitet skapas för att garantera robust kommunikation?

(II) Hur kan robusthet analyseras och förbättras för ML-baserade algoritmer avseende både traditionella störhot och ML-anpassade attacker (eng. *adversarial machine learning*, AML)

(III) Hur kan ML-teknik nyttjas för effektivare resursallokering i mobila radionät?

1.3 Läsanvisningar

Denna rapport är skriven på ett översiktligt sätt utan att beskriva alla tekniska detaljer. Grundläggande kunskap om radiokommunikation och ML underlättar förståelsen av innehållet. För fördjupad bakgrundskunskap hänvisas till [3, 4].

Rapporten är indelad i kapitel med innehåll som kan läsas fristående från varandra. Kapitlen beskriver översiktligt resultaten från de aktiviteter som har genomförts under 2023 i syfte att besvara frågeställningarna i avsnitt 1.2.

Kapitel 2 behandlar frågeställning (I) om träningsdata. I kapitel 3 presenteras sändar- och mottagartekniker för det fysiska lagret, i vilket avsnitt 3.1 beskriver demodulation och avkodning i impulsbrus, avsnitt 3.2 studerar användningen av en neural mottagare och avsnitt 3.3 föreslår ett sätt att utnyttja AML för att försvåra modulationsklassificering. Detta kapitel adresserar frågeställning (II), och till viss del även frågeställning (III). I kapitel 4 beskrivs påbörjat arbete avseende allokering av tidluckor med ML-baserade algoritmer, vilket syftar till att besvara frågeställning (III), men med hänsyn till robusthet enligt frågeställning (II). Slutligen presenteras rapportens slutsatser i kapitel 5. Appendix A beskriver flertalet mjukvarubibliotek som är relevanta för studier av ML i radiokommunikationstillämpningar.

2 Träningsdata

Mängden och kvaliteten på träningsdata är avgörande för hur ML-algoritmer tränas och presterar. ML-algoritmer använder data för att lära sig mönster och förhållanden mellan in- och utdata. Om mängden data är otillräcklig kan algoritmen ha svårt att generalisera och prestera bra på tidigare osedda data. Om träningsdatasetet är av dålig kvalitet, exempelvis om det innehåller felaktiga, irrelevanta eller alltför brusiga data, lär sig algoritmen felaktiga mönster.

Bra träningsdata för ML behöver ha ett antal viktiga egenskaper för att säkerställa att algoritmerna kan generalisera och prestera väl på nya, okända data. Bland annat måste träningsdata vara relevanta och representativa för tillämpningen, innehålla en mångfald för att underlätta generalisering samt vara fria från felaktigheter. Hur detta ska åstadkommas är inte trivialt, men det finns många lärdomar från omvärlden om vilka problem som kan uppstå, hur problemen kan lösas samt vad som bör tas i beaktande avseende skapande och hantering av träningsdata och träning av algoritmer.

2.1 Generella riktlinjer

Mängden data och dess variation har stor inverkan på ML-algoritmers prestanda. Den mängd träningsdata som behövs för att erhålla önskad prestanda varierar beroende på applikation, men mer data ger generellt ökad prestanda, givet att kvaliteten är tillräcklig.

2.1.1 Uppdelning av data

Datamängden delas vanligen upp i tre disjunkta delar för träning, validering och test. Den första delen används till att träna och lära upp algoritmen. Den andra delen är till för att bedöma prestandan kontinuerligt under träningen och justera parametrar. Den sista delen avsätts till att utvärdera algoritmen [5, s.176].

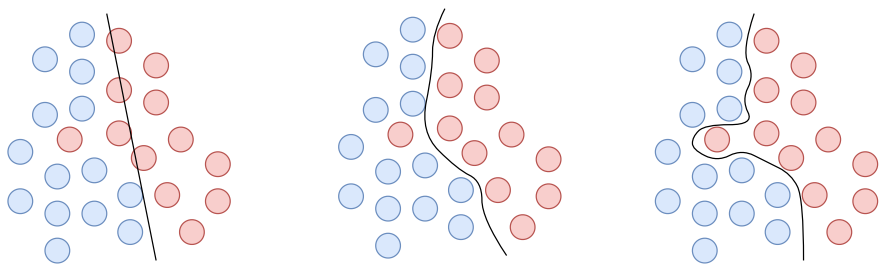
Det finns flera metoder att dela upp data i dessa tre delar och hur uppdelningen görs kan ha stor inverkan på algoritmens prestanda. Det är viktigt att de klasser och egenskaper som datasetet ska innehålla finns representerade i samtliga delar. En bra uppdelning gör det enklare att utvärdera algoritmen och gör samtidigt algoritmen mer robust. Med en naiv, slumpmässig uppdelning riskerar prestanda att variera kraftigt mellan träning och test.

En vanlig metod är (k-delad) korsvalidering (eng. cross validation). Med korsvalidering avsätts först en del till test och resterande data delas upp i k delar, där varje del återspeglar hela datamängdens fördelning. Algoritmen tränas och valideras k gånger med denna metod, med en ny valideringsdelmängd varje gång. Resultaten från varje träningscykel medelvärdesbildas för att ge en mer robust bedömning av algoritmens prestanda. Korsvalidering ger en bättre uppskattning av algoritmens generaliseringsförmåga jämfört med slumpmässig indelning, eftersom den tar hänsyn till prestanda över olika delar av datamängden.

Det finns flera varianter av korsvalidering, såsom stratifierad korsvalidering (där fördelning av klasser bibehålls i varje delmängd vilket är användbart när datamängden är obalanserad med avseende på klasserna) eller Monte-Carlo korsvalidering (där många delmängder väljs ut och resultatet aggregeras över flera iterationer). En annan valideringsmetod är så kallad *bootstrapping* [6], vilken skapar flera slumpmässiga urval av data från den ursprungliga datamängden. Urvalen skapas med återläggning, dvs. samma datapunkt kan ingå i flera urval. Detta är särskilt användbart när det finns en begränsad mängd tillgängliga data. Valet av valideringsmetod beror på många faktorer såsom datamängd, struktur och problemområde och är avgörande för att få en robust bedömning av en algoritms prestanda.

2.1.2 Datautökning

Två vanliga problem som kan uppkomma vid otillräcklig mängd träningsdata är underträning (eng. underfitting) och överträning (eng. overfitting), konceptuellt illustrerade i figur 2.1. Vid underträning kan algoritmen inte hitta sambandet mellan in- och utdata på grund av brist på träningsdata [7]. Vid överträning presterar algoritmen bra på träningsdatasetet, men presterar markant sämre på testdatasetet. Algoritmen har då tränats på ett begränsat dataset och lärt sig mönster i brusfördelningen på träningsdatasetet istället för att generalisera data, vilket gör att nya data inte går att prediktera korrekt. Lösningen på både under- och överträning är att ha en större mängd träningsdata av hög kvalitet och med stor variation så att alla nödvändiga fall är representerade.



(a) Underträning (b) Bra träning (eng. good fit) (c) Överträning

Figur 2.1: Visualisering av olika nivåer av träning på ett dataset med två klasser (blå och röd), i vilka (a) tränats för lite, (b) tränats bra så att algoritmen (linjen) generaliserar en funktion utifrån data och (c) tränats för mycket.

I vissa fall är det svårt att få tillräckligt mycket data för att träna ML-algoritmer på ett adekvat sätt. I försvarstillämpningar kan sekretess försvåra insamling av verkliga data eftersom militära försök i den skala som krävs är svåra att genomföra. Svårigheten med sekretess gäller även till viss del generering av syntetiska data, framförallt när modeller av system och signaler representerar verkliga militära situationer.

Datautökning, även känt som dataaugmentering, omfattar metoder för att öka datamängden genom att skapa nya datapunkter från befintliga data genom olika typer av transformationer. Inom bildbehandling är det vanligt att nya bilder skapas genom exempelvis rotation, förskjutning, spegling eller zoomning på existerande datapunkter. Inom radiokommunikation kan konjugering och komplexvärd rotation utföras på radiosignalens I- och Q-komponent eller ytterligare brus adderas till signalen [8]. Syftet med datautökning är att förbättra algoritmens förmåga att generalisera och prestera väl på olika scenarier genom att lägga till variation i träningsdata.

En risk med att skapa en ny datapunkt på detta sätt är att datapunkten byter klass eller blir helt oigenkännlig. Exempelvis kan en 180-graders rotation av bokstaven *m* närmare efterlikna *w*, medan samma rotation gör att bokstaven *g* blir något helt nytt. Det är viktigt att tänka på vilka transformationer som är rimliga för problemet i fråga och att kontrollera så att den resulterande datamängden ser ut som förväntat.

Inom bildbehandling kan ett begränsat dataset utökas med hjälp av så kallad *pooling*, i vilken en bild kan bli uppdelad i mindre segment genom att applicera ett glidande fönster över bilden och välja ut den pixel med den högsta eller lägsta intensiteten (så kallade max- och min-pooling) i fönstret för att skapa en ny bild. På så sätt kan en datapunkt producera flera nya bilder. I [9] utökas ett relativt litet dataset (på 7 500 bilder) mångfalt via tre olika typer av pooling (min-, max-, och medelvärdes-pooling). Utökningen i [9] skapade dubbletter då olika fönsterstorlekar resulterade i att samma pixel valts för flera av originalbilderna. Dubbletterna gav upphov till en övertränad algoritm. Lösningen på detta problem är att antingen använda en annan datautökningsteknik eller att upptäcka problemet via en lämplig valideringsmetod.

Ibland är det inte mängden data som är problemet utan fördelningen. Om fördelningen på insamlade data inte återspeglar verkligheten, kan översampling (eng. oversampling) och undersampling (eng. undersampling) användas. Översampling nyttjas för att öka antalet datapunkter av en eller flera klasser i ett dataset. I vissa fall är dubbletter inte önskvärda i ett dataset och i dessa fall kan brus adderas för att göra datapunkter unika [10, s.22]. Adderat brus kan också ha en fördel i vissa domäner, exempelvis radiokommunikation, i vilken brus och interferens påverkar den mottagna signalens struktur. En klassificeringsalgoritm som ska användas i en brusig miljö kan då ha nytta av datapunkter med varierande brusnivå i träningen för att lättare kunna göra skillnad på olika konstellationssymboler i en skarp situation [11].

I fall då datamängden är obalanserad, men det inte finns en möjlighet att översampla de mer ovanliga klasserna på ett rimligt sätt utan att införa dubbletter är undersampling ett alternativ. Undersampling innebär att exkludera vissa datapunkter från ett dataset för att förbättra balansen i datamängden. Det enklaste sättet är att slänga bort data för att få en jämnare fördelning.

Utöver anledningen att öka balansen genom att ta bort data kan det även vara rimligt att ta bort punkter som är alldeles för avvikande från resterande data. Detta gäller endast för träningsdata eftersom dessa datapunkter fortfarande är relevanta under testning [12].

Om datapunkter tas bort finns det en risk att viktig information som hade gjort algoritmen mer robust också tas bort. Detta bör endast göras om bedömningen är att dessa datapunkter riskerar att försämra algoritmen under inlärningsprocessen. Anledningen till att detta görs är att avvikande datapunkter kan vilseleda algoritmen när den tränas. Att avgöra om en datapunkt är för avvikande är en domän- och applikationsspecifik fråga, och är ofta inte helt rättframt. Bedömningen måste göras utifrån om datapunkten representerar ett rimligt och relevant fall eller inte. Inom radiokommunikation kan en datapunkt anses vara för avvikande om exempelvis signalstyrkan varierar kraftigt över det som förväntas representera ett normalläge eller om den representerar ett modulationsformat som en modulationsklassificerare inte är avsedd att klassificera. Ett alternativ till över- och undersampling är att data viktas olika mycket, så att algoritmen tar mer hänsyn till vissa datapunkter.

2.1.3 Dataförgiftning och dataextrahering

Att använda existerande dataset ger förmånen att kunna jämföra olika lösningar med varandra, men att basera algoritmer på öppna dataset utan tillräcklig granskning kan leda till sårbarheter. Dataförgiftning (eng. data poisoning eller data manipulation) innebär att felaktiga data injiceras i ett dataset med syfte att vilseleda den algoritm som tränas på detta.

Syftet med dataförgiftning är att underminera en algoritms förmåga att generalisera på nya, tidigare osedda data. En attack kan ske på olika sätt, exempelvis genom att införa störningar, felaktiga annoteringar av data eller andra förändringar som är avsedda att vilseleda algoritmen [13]. Konsekvensen blir att algoritmen tar hänsyn till korrumperade datapunkter under träning vilket ger försämrad prestanda under testning. Andra exempel på dataförgiftning är att felmarkera närliggande datapunkter eller att skapa en obalans i datasetet så att någon klass blir underrepresenterad med syfte att algoritmen inte ska kunna klassificera denna klass korrekt. Felmärkta datapunkter är relativt lätta att upptäcka, då dessa påverkar starkt hur väl algoritmen presterar när den testas. Ett obalanserat dataset kan vara svårare att upptäcka eftersom detta endast är synligt om datasetet som algoritmen testas på inte har samma fördelning av klasser som det dataset som använts under träning.

Dataextrahering (eng. data extraction) är en typ av attack mot maskininlärningssystem [14]. Till skillnad från att inrikta sig på att manipulera algoritmens effektivitet eller prestanda är målet att få insikter i den kunskap som ligger bakom algoritmen, särskilt de data på vilken algoritmen har tränats. En extraktionsattack angriper en algoritm genom att tillhandahålla specifika inmatningar för att observera algoritmens utmatningar. Genom dataextrahering kan angripare få ut känslig information om en algoritms träningsdata. Bland annat är det möjligt att utläsa individer eller entiteter i data (exempelvis personuppgifter), hitta mönster och teman, klassificeringskategorier, fördelningar, format, struktur och övrig känslig information. En algoritm som har tränats på känslig eller sekretessbelagd information kan utgöra en allvarlig risk för dataavslöjande [15]. Extraktionsattacker kan därför utgöra en betydande säkerhetsrisk och metoder för att skydda träningsdata och algoritmer mot den typen av hot, som dataför-

vrängning, datakryptering eller införande av falska sampel, bör studeras och övervägas vid utveckling och träning.

För att hantera risker med dataförgiftning och dataextrahering är det viktig att noggrant granska och validera det insamlade datasetet samt vid behov applicera lämpliga skyddsmetoder, innan det används för träning av algoritmer.

2.1.4 Validering av data

Innan träning av en ML-algoritm påbörjas är det viktigt att säkerställa att träningsdatasetet ser ut som förväntat och inte innehåller några felaktigheter. Det kan röra sig om brusiga data, felaktigt annoterade data, systematiska fel eller förekomst av dubbletter. Att manuellt undersöka ett dataset är ofta svårt, delvis på grund av mängden data men även typen av data kan försvåra analysen. För ett dataset innehållande olika radiosignaler är det exempelvis svårt för en människa att se på signalen om den har korrekt modulationstyp, bandbredd eller typ av brus. Effekten av dataset av låg kvalitet finns dokumenterat i flera papper, exempelvis [16, 17, 18].

Valideringsmetoder för ML-algoritmer kan också användas för att ge en indikation på datasetets kvalitet. Korsvalidering, som beskrevs i avsnitt 2.1.1, har vissa brister, vilket påvisas bland annat i [19, 20], i vilka ett begränsat dataset utökas via olika former av datautökning. I [9] belyses problemet att maskininlärning ofta studeras utifrån ett designperspektiv, och fokus i forskarvärlden läggs på att skapa algoritmer som är kraftfulla och komplicerade. En aspekt som dock har blivit åsidosatt är hur kvaliteten på det dataset som används för träning och validering påverkar prestanda av dessa algoritmer, framförallt när metoder för att utöka dataset har applicerats.

Två alternativ till korsvalidering som föreslås i [9] är *metamorphic training* som består av att kontrollera olika slags attribut (eng. metamorphic relations), exempelvis pålitlighet, variation och sanningsenlighet. Valideringen sker genom att skapa rimlighetstest för att undersöka hur algoritmerna presterar vid små, kontrollerade förändringar. Ett exempel är att lägga till 10 % nya bilder eller dubbletter av existerande bilder för att bedöma hur prestanda förändras. I situationer då datapunkter från datasetet avsatt för själva träningen kan misstänkas ha läckt in i validerings- och testdata är det viktigt att undersöka datasetet via denna typ av testning för att avgöra om algoritmerna presterar på ett rimligt sätt när datasetet ändras, utökas eller kommer från osäkra källor.

2.1.5 Träningsordning

En aspekt som kan påverka prestanda är i vilken ordning träning sker, dvs. i vilken ordning datapunkter presenteras för algoritmen. I [21] görs ett försök att kvantifiera effekten av att ändra ordningen på träningsdata för ett antal olika algoritmer. Scenariot är ett klassificeringsproblem, i vilket data från tre olika dataset används: ett bestående av 5 620 bilder på handskrivna siffror, ett bestående av 178 kemiska analyser för tre olika viner och ett bestående av över 60 000 bilder på handskrivna siffror. Källan för de två första dataseten är Scikit-learn [22], ett öppet ML-bibliotek som innehåller flera

enkla exempel på algoritmer, funktioner, dataset och verktyg för att utföra analyser och beräkningar för olika problem. Det sistnämnda är det välstuderade datasetet MNIST [23]. Samtliga dataset är välkända och välanvända för klassificeringsproblem.

I [21] utvärderas följande fyra klassificeringsalgoritmer:

- *Multi-layer perceptron* (MLP): Ett neuralt nätverk som försöker imitera en mänsklig hjärna genom att skapa anslutningar mellan in- och utdata via stora lager av neuroner. MLP har visats vara effektivt för ett flertal ML-applikationer.
- Beslutsträd (eng. decision trees): Samlingar av binära klassificeringssteg som är hierarkiskt organiserade i en trädstruktur för att skapa en sammansatt klassificeringsalgoritm, i vilken varje steg representerar en egenskap hos datamängden.
- Slumpskog (eng. random forest): En klassificeringsstruktur som väger samman beslut från flera slumpmässigt utvalda beslutsträd.
- Kvadratisk diskriminantanalys (eng. quadratic discriminant analysis, QDA): En metod som, med kvadratiska ytor, separerar och grupperar data i olika klasser.

Vid träning av algoritmerna i [21] används 20 % av tillgängliga data till test och resten till träning och validering. Urvalet till de båda dataseten sker helt slumpmässigt. Ordningen på träningsdata slumpas sedan och algoritmerna tränas och utvärderas. Detta genomförs 1 000 gånger.

För det lilla datasetet med handskrivna siffror visar resultaten en stor spridning mellan algoritmerna. Framförallt påvisas en stor variation på resultaten för samma algoritm. Exempelvis lyckas QDA klassificera mellan 65 % och 93 % av bilderna för data från samma dataset, då den enda skillnaden är ordningen på träningsdata. Övriga algoritmer visar en mindre spridning, mellan sex och sju procentenheter för samtliga.

Det andra datasetet är betydligt mindre med endast 178 bilder och resultaten för klassificeringen varierar också kraftigt för de flesta av algoritmerna. För MLP varierar prestanda från endast 10 % (vilket är sämre än att gissa på samma klass hela tiden) upp till 100 %. Författarna av [21] resonerar att spridningen kan vara orimligt stor på grund av för få datapunkter i datasetet.

Utvärdering på det betydligt större MNIST-datasetet, med över 60 000 bilder, visar på mindre spridning, mellan två och tio procentenheter för de fyra algoritmerna. I [21] konstateras att ordningen på träningsdata påverkar klassificeringsalgoritmernas prestanda i samtliga undersökta fall. Effekten minskar för större dataset, men den försvinner inte.

Rekommendationen från [21] är att genomföra flera tester för att garantera en algoritms prestanda eller effektivitet för ett visst problem. Det ska påpekas att vissa algoritmer påstås vara träningsdatainvarianta, dvs. att ordningen på träningsdata inte spelar någon roll [21]. Sådana algoritmer (exempelvis *support vector machines* och K-närmaste grannar) har inte undersökts i [21]. En lärdom, för att inte råka ut för problemet som illustreras i [21], är att se till att ha tillräckligt mycket data för träning. Vid misstanke om att den tillgängliga datamängden inte räcker till bör valet av algoritmstruktur övervägas till ett mer robust alternativ.

Inom radiokommunikation kan ordningen på träningsdata också utnyttjas för att öka robusthet, vilket visas exempelvis i [10]. En algoritm som ska utföra demodulation baserat på en bakomliggande brusmodell tränas på successivt starkare brus, så att algoritmen först lär sig modulationsformatet med högt SNR, för att sedan utsättas för brus med lägre och lägre SNR vilket visar sig ha betydelse för prestandan.

2.1.6 FAIR guiding principles

En sammansättning av forskare från olika forskningsområden inom akademin och industrin publicerade år 2016 de så kallade *FAIR guiding principles* för att förbättra användande, tillgänglighet och återskapande av data och forskningsresultat [24]. FAIR står för *findability*, *accessibility*, *interoperability* och *reusability* av digitala resurser, såsom källkod och data. Riktlinjerna syftar till bland annat tillgänglighet och reproducerbarhet av resultat, vilket även är applicerbart på maskininlärning. Bakgrunden till ansatsen är att forskarna har identifierat behov av att förbättra hur forskningsresultat och data hanteras. I stora drag kan de föreslagna riktlinjerna beskrivas enligt följande:

- Findable: Data ska vara väl beskrivna, annoterade och tydliga för att vara *enkla att hitta*. Data och metadata ska registreras och indexeras i ett lämpligt, sökbart hjälpmedel för ämnetsområdet.
- Accessible: Data och metadata ska *vara enkla att komma åt* via tydliga identifieringsparametrar enligt ett standardiserat protokoll. Protokollet ska vara öppet, gratis och universellt användbart. Metadata ska vara tillgängliga även om data inte längre är det.
- Interoperable: Riktlinjerna efterfrågar ett gemensamt språk för användning av dataset, med tydliga referenser till andra data, för att främja *enkel användning*.
- Reusable: Attribut och annotering av data och metadata ska vara begripliga och *enkla att återanvända*. Eventuella licenser ska vara tillgängliga eller tydligt beskrivna. Alla data ska uppnå ämnesområdets standard.

FAIR-riktlinjerna kan anses vara tämligen generella, men behovet är tydligt och efterfrågan på reproducerbara data och forskningsresultat är påtaglig i ett flertal forskningsområden. Råden är sunda och representerar en målbild som bör eftersträvas vid generering och nyttjande av träningsdata för kommunikationstillämpningar. Myndigheter, som FOI eller FMV kan ha nytta av ett gemensamt system eller interna riktlinjer för hur data skapas, annoteras, bevaras och indexeras likt det som föreslås i FAIR-systemet. På så sätt slipper olika forskningsprojekt skapa nya data för utveckling av ML-algoritmer som kan nyttja existerande dataset från andra försök.

En svårighet för att etablera gemensamma dataset vid en myndighet som FOI är att dataset med militär relevans ibland omfattas av sekretess. För utvärderingar och analyser av algoritmer kopplat till mer generella system som inte omfattas av sekretess kan dock gemensamma dataset eller riktlinjer för att generera data vara till nytta.

2.2 Träningsdata för radiokommunikation

ML för radiokommunikation är ett relativt nytt forskningsområde jämfört med användningen av ML inom andra teknikområden, exempelvis datorseende, bildbehandling, röstigenkänning, språkbehandling eller medicin. Användningen av ML för radiokommunikation står inför specifika utmaningar avseende att konstruera och nyttja träningsdata av tillräcklig kvalitet, som skiljer sig från andra teknikområden, på grund av områdets natur [25].

2.2.1 Syntetiska eller verkliga data

Ett träningsdataset för en algoritm som ska appliceras i ett trådlöst scenario måste representera de signalerna som algoritmen ska hantera. Det är svårt att hitta ett existerande, tillgängligt dataset som är representativt för alla problem som ska lösas inom radiokommunikation. En lösning till detta är att konstruera ett eget dataset, antingen syntetiskt eller uppmätt från verkliga system. Syntetiska data är enklare att skapa stora mängder av, men nackdelen är att det är svårt att få med komplexiteten och dynamiken som en signal utsätts för i ett verkligt scenario. Både att skapa syntetiska och verkliga data medför ett antal utmaningar som måste hanteras. Att identifiera hela den mångfald av verkliga förhållanden som påverkar sändning och mottagning för system i den miljön de verkar, tillsammans med hårdvarueffekter som systemen besitter, är en signifikant del av konstruktionen av ett passande dataset [26]. Några av utmaningarna för att konstruera ett dataset för ett radiokommunikationsscenario sammanfattas enligt följande:

- Realistisk modell av radiokanalen: Datasetet måste fånga dynamiken och komplexiteten hos trådlösa signaler, och få med de fysiska egenskaperna som påverkar en signal som skickas trådlöst. Effekter som fädning, interferens, brus och flervägsutbredning kan leda till väldigt olika påverkan beroende på den elektromagnetiska miljön systemet befinner sig i. Realistiska egenskaper hos interferens och brus är svåra att modellera på ett bra sätt och påverkar därför prestandan hos ML-algoritmer.

- Scenario: Effekterna som påverkar radiokanalen beror också till stor del på i vilket scenario systemet är tänkt att användas. Det är exempelvis skillnad på trådlös kommunikation i inomhusmiljö jämfört med system som ska användas utomhus.

- Vågformsegenskaper: Ett radiosystem är oftast tänkt att användas inom ett visst frekvensband med signalegenskaper som kan variera. Många moderna kommunikationsstandarder, som LTE[1] har tusentals konfigurationsinställningar [26]. En annan aspekt är att moderna system är adaptiva, och kan ändra exempelvis modulationsformat, kodningstyp, sändareffekt eller fördelning av kanalresurser utefter

[1] long-term evolution

hur radiokanalen varierar. Datasetet måste innehålla representativa data som täcker de variationer av signalerna som kan uppstå, och den mångfalden kan vara svår att uppnå.

- Militära aspekter: Att skapa dataset för militära radiosystem och scenarier medför svårigheter med sekretess och känslig information. Ska ML-algoritmen användas i skarpa lägen måste den tränas på skarpa data, oavsett om de är syntetiserade eller inte. Att samla in riktiga data som är relevanta för ett militärt scenario är utmanande. Mängden data måste vara tillräckligt stor och en sådan datainsamling kan omfattas av sekretess, beroende på vilket scenario eller system som mäts upp. Detta gäller även syntetiska data, givet att systemen modelleras med relevanta parametrar. En möjlighet är att konstruera dataset med generella systemparametrar som ungefär matchar det användningsområde som algoritmerna är tänkta att verka i, men risken är att prestandan blir undermålig eller svåranpassad till skarpa scenarier.

Det råder delade meningar i forskarvärlden om valet att nyttja syntetiska data eller använda verkliga data. I [27] påstår författaren Timothy J. O'Shea att träning med syntetiska data är ett mindre lämpligt angreppssätt inom maskininlärning i allmänhet, men att radiokommunikation är ett undantag. O'Shea argumenterar för att radiokommunikationssignaler i grunden är syntetiska och att flera steg i kommunikationskedjan är deterministiska och kan syntetiseras precis som ett verkligt system, inklusive aspekter som modulation, pulsformning och andra karaktäristiska sändningsegenskaper. De steg som inte är deterministiska, såsom kanaleffekter, går ändå att karaktärisera på ett tillfredsställande sätt [27]. Detta kan göras genom matematiska modeller för exempelvis tidsvarierande fädningskanaler med kompensering för frekvens- och tidssynkronisering, vilket sägs ge samma beteende som ett verkligt scenario.

I [28] genomförs experiment med riktiga data i ett trådlöst scenario, i vilket en algoritm tränas för att utföra digital modulationsklassificering. Träningsdata består av 250 000 sändningar med varierande signal-till-brusförhållanden (SNR), sändarhårdvara och interferens. Algoritmen utvärderas i [28] i ett flertal fall, i vilka det neurala nätet tränas på vissa SNR och på en viss hårdvarutyp men utvärderas inte under exakt samma förhållanden som det tränats på. Resultaten visar att klassificeringsprestanda är hög för de fall i vilka träningsdata överensstämmer bra med testfallen. Prestanda degraderas om datasetet inte fångar upp dynamiken i det verkliga scenariot, exempelvis när hårdvaran byts från en högkvalitativ till en billigare mottagare, eller för olika nivåer av interferens. I [28] dras slutsatsen att det är svårt att skapa generella ML-algoritmer för radiokommunikation som fungerar för specifika tillämpningar, med specifika radiomottagare och systemparametrar. Rekommendationen från [28] är att antingen utveckla ett system för så generella parametrar som möjligt, eller utvidga träningsdatasetet i den grad att fler specialfall kan inkluderas för mer robust träning. Dataseten som har använts i både [27] och [28] finns tillgängliga online.

Brist på verkliga, uppmätta data behöver inte nödvändigtvis vara ett problem. Genom så kallad *meta-learning* kan en algoritm tränas för att utföra nya uppgifter el-

ler uppdatera förmågor baserat på en begränsad mängd träningsdata [29]. Via meta-learning tränas inte en algoritm för att utföra en specifik uppgift, utan mer att prestera väl på en bredare klass av uppgifter, eller att verka i ett specifikt problemområde. Grundidén är att en meta-learner ska kunna generalisera och snabbt anpassa sig till nya och liknande uppgifter. Till skillnad från klassisk maskininlärning, i vilken fokus ligger på att optimera algoritmens parametrar och vikter för den uppgift som ska utföras, ligger fokus för en algoritm som tränas via meta-learning på att optimera själva inlärningsprocessen snarare än specifika egenskaper hos en given uppgift. På så sätt kan algoritmen dra nytta av tidigare inlärningserfarenheter när den konfronteras med en ny uppgift, även om träningsdatamängden för den nya uppgiften är liten [30].

Det är även möjligt att kombinera syntetiska och verkliga data vid träning av ML-algoritmer, vilket kan vara användbart i fall när det endast finns en begränsad mängd verkliga data att tillgå. Ett vanligt koncept är att genomföra domänanpassning (eng. domain adaptation). Domänanpassning innebär att anpassa en ML-algoritm som har tränats på en källdomän för att förbättra prestanda på en annan, besläktat domän. En vanlig typ av domänanpassning är så kallad *transfer learning*. Det är en ML-teknik i vilken en förtränad algoritm som har tränats för en viss uppgift finjusteras (transfereras) för att utföra en annan uppgift [13]. Idén bakom transfer learning är att den kunskap som algoritmen har lärt sig från den ursprungliga uppgiften kan vara användbar för att förbättra prestandan på en ny, relaterad uppgift, särskilt om det finns begränsat antal tillgängliga data för den nya uppgiften. Med transfer learning kan en algoritm då tränas på syntetiska data, där en stor datamängd kan genereras, och sedan finjusteras med den begränsade verkliga datamängden [31].

2.2.2 Existerande dataset för radiokommunikation

Forskning och utveckling av ML-algoritmer för radiokommunikation underlättas om det finns etablerade dataset tillgängliga. Detta har påpekats i flera publikationer, bland annat i [25, 28, 32]. Stora, öppna databibliotek finns inom andra domäner, exempelvis objektklassificering eller tal- och handstilsigenkänning [23]. I [26] poängteras att det inte finns något ekvivalent dataset för trådlös kommunikation. Vissa forskare har publicerat partiella dataset, vilka oftast är en restprodukt från något specifikt forskningsprojekt. Dessa dataset är antingen ofullständiga eller omogna för utveckling av nya algoritmer. De flesta tillgängliga dataset är skapade vid universitet eller statliga myndigheter för användning på det fysiska lagret [33]. Ett sätt att skapa publika dataset är genom kontinuerlig datainsamling, där allmänheten kan nyttja och generera egna data för att bidra till datasetet. Denna typ av insamlingsförsök erbjuder ofta mjukvaruverktyg till privatpersoner för att lägga till datapunkter i datasetet och är vanlig inom mobiltelefoniapplikationer då en användare passivt samlar in information genom att enbart ha mobilen på sig [33]. Denna typ av dataset kan vara användbara i militära tillämpningar, eftersom dataseten kan bli stora och detaljrika med många unika datapunkter i verkliga miljöer baserat på människors realistiska rörelsemönster och beteenden. Nackdelen är att data som samlas in på det här sättet måste kontrolleras noga

med avseende på giltighet och pålitlighet [34].

Inom IEEE Communications Society har ett initiativ upprättats för att främja utveckling av maskininlärning inom trådlös kommunikation [35]. Syftet med detta initiativ är att etablera en uppsättning gemensamma problem med tillhörande dataset som forskare kan använda för att utvärdera och jämföra ML-algoritmer. Tre exempel på dataset från [26] och [35] avsett för tillämpningar inom radiokommunikation är:

- *Transfer Learning for RF Domain Adaptation* [36]: Syntetiskt dataset ursprungligen framtaget för att träna så kallad *transfer learning* [13]. Datasetet är genererat via ett publikt signalbehandlingsbibliotek, liquid-dsp[2], och innehåller totalt 13,8 miljoner datapunkter av 22 olika modulationstyper (2-, 4-, 8- och 16-PSK, 4-DPSK, 16-, 32- och 64-QAM, 16- och 32-APSK, FSK, GFSK, MSK, GMSK, två varianter av FM samt fyra varianter av AM). Till varje modulationstyp har adderats vitt brus med SNR i intervallet -10 dB till 20 dB och frekvensoffset med ± 10 % av samplingstakten.

- RadioML [25]: Ett av de mest citerade och använda dataseten för trådlös kommunikation, som har blivit ett referensdataset för att utvärdera och jämföra ML-algoritmer inom modulationsklassificering. Detta dataset är skapat av O'Shea et al. år 2016 och existerar i tre upplagor (RadioML 2016.04C, 2016.10A samt 2018.01A), varav 2016.10A har använts i störst omfattning. RadioML 2016.10A innehåller elva modulationstyper (8-PSK, DSB- och SSB-AM, BPSK, CPFSK, GFSK, 4-PAM, 16- och 24-QAM, QPSK samt WBFM) med olika SNR och sampeltakt. RadioML 2018.01A är en uppgradering med 24 modulationstyper. Dataseten har utstått kritik för att innehålla felaktigheter, framförallt med generering och annotering av konstellationssymboler. Detta poängteras av upphovsrättstagaren DeepSig [37], som tar avstånd från dataseten och rekommenderar forskare att skapa sina egna dataset. Trots detta använts RadioML som ett riktmärke för prestanda i många publikationer.

- För mobiltelefoni existerar ett flertal dataset, exempelvis [38] som innehåller uppmätta GSM-, UMTS- och LTE-signaler för olika SNR (1-15 dB) i urban och sub-urban miljö. Datasetet skapades för ett forskningsprojekt i vilket ett neuralt nätverk utvecklades för att klassificera de olika mobiltelefonityperna. Datasetet innehåller över 60 000 datapunkter av varje signaltyp. Ytterligare dataset för mobiltelefoni återfinns i [26].

2.3 Rekommendationer

Att skapa och använda dataset för robust radiokommunikation medför flera utmaningar och det är inte självklart hur ett dataset av tillräcklig kvalitet ska konstrueras. En faktor är om datasetet ska utgöras av verkliga eller syntetiska data, vilket medför olika för- och nackdelar. Uppmätta data i det scenario som ML-algoritmen är tänkt att användas

[2] https://liquidsdr.org/

ger bäst resultat, då de faktiska kanaleffekterna kan inkluderas i träningen. Militära applikationer har ofta inte den möjligheten. Ofta är det svårt att planera och genomföra militära försök i den skala som är nödvändig för att generera data i tillräcklig mängd.

Insamling och uppmätning av data för militära ändamål kan också omfattas av sekretess. I sådana fall kan ett syntetiskt dataset vara att föredra, då det är viktigt att få med så varierade kanal- och systemeffekter som möjligt för att fånga upp relevanta egenskaper för träningen. Ett syntetiskt dataset kan även skräddarsys med eller utan känslig systeminformation. Om datasetet ska nyttjas av andra kan systemet modelleras med generella systemparametrar för att undvika spridning av känslig information, till skillnad från tillämpningar där det är nödvändigt att skarpa system modelleras för att fånga upp relevanta systemegenskaper. En annan avvägning är att konstruera ett eget dataset eller att använda ett publikt dataset. Ett egengenererat dataset har fördelen att innehållet kan anpassas till det scenario som är intressant, till skillnad från att nyttja ett publikt dataset som kan innehålla okända felaktigheter. Risken är också att de utvecklade ML-algoritmerna måste anpassas till de tillgängliga dataseten istället för tvärtom.

Även om det går att undvika sekretess- och datahanteringsproblematiken återstår vissa hinder för att skapa kvalitetsdata inom radiokommunikation, exempelvis otillräcklig datamängd eller svårighet att modellera realistiska kanalförhållanden för syntetiska dataset. I båda fallen måste risken för felaktigt märkta och dåliga datapunkter också vägas in.

Rekommendationer för att generera robusta träningsdata för ML-algoritmer avsedda för trådlös kommunikation omfattar därför följande punkter:

- Utred vilka scenarion, system, frekvenser, modulationstyper, sändar- och mottagarförhållanden som är relevanta för problemet.
- Modellera de effekter som påverkar en signal, exempelvis fädning, interferens, brus och flervägsutbredning, för att skapa realistisk kanal- och signalmodellering.
- Tillse korrekt annotering av data samt att fördelningen av egenskaper hos data, exempelvis olika modulationstyper, frekvenser eller signal-till-brusförhållanden är rimlig och felfri. Tydlig indexering på vad datamängden innehåller och hur den är uppmärkt underlättar arbetet.
- Träna algoritmen på olika ordningar av datasetet för att få en överblick av prestandan, alternativt nyttja en struktur som är okänslig för träningsordning.
- Nyttja datautökning vid brist på tillräcklig mängd data.
- Använd valideringsmetoder för att kontrollera kvaliteten på datasetet.
- Följ eller skapa gemensamma riktlinjer (t.ex. FAIR-riktlinjerna eller ett verksamhetsanpassat system). Ett gemensamt tillvägagångssätt för att konstruera, hantera och använda data för maskininlärning möjliggör ett enklare samarbete och minskar risken för dubbelarbete.

Genom att följa rekommendationerna elimineras inte alla problem med träningsdata av otillräcklig kvalitet och kvantitet. Att ta hänsyn till de aspekter som påverkar ett trådlöst system i generering och hantering av träningsdata leder dock ett steg närmare robust kommunikation.

3 Sändar- och mottagartekniker för det fysiska lagret

Detta kapitel beskriver tre exempel på användning av ML-teknik för det fysiska lagret i militär radiokommunikation. Avsnitt 3.1 beskriver demodulation och avkodning i impulsbrus, avsnitt 3.2 studerar användningen av en neural nätverksbaserad mottagare och hur en sådan kan göras mer robust mot störning och avsnitt 3.3 föreslår ett sätt att utnyttja ML-attacker för att försvåra modulationsklassificering.

3.1 Demodulation och avkodning i impulsbrus

Moderna plattformar, såväl militära som civila, består av ett flertal elektroniska system, vilka kan utgöra oavsiktliga störningskällor mot radiomottagare. Den omgivande miljön bidrar till den interferens som upplevs av en mottagare. Elektrifierad infrastruktur, såsom järnvägar, kraftledningar, elnät, vindkraftverk och solcellsanläggningar kan skapa interferens i radiosystem som befinner sig i närheten. Prestanda hos en radiomottagare påverkas negativt av sådan interferens. Prestandaförsämringen är särskilt påtaglig om interferensen är av en typ som mottagaren inte är utvecklad för att hantera, exempelvis med kraftig impulskaraktär som varierar över tid.

För att hantera varierande interferensmiljö krävs en adaptiv mottagare som kan anpassa demodulation och avkodning. Anpassning av mottagaralgoritmer efter bruskaraktär har visat stora prestandavinster [39]. Ett problem med traditionella algoritmer är att beräkningskomplexiteten är hög. Därför föreslås algoritmer baserade på neurala nätverk som kan utföra adaptiv mottagning i krävande och varierande interferensmiljö, men med mindre beräkningskomplexitet [10, 40]. Dessa algoritmer beskrivs översiktligt i detta avsnitt. För detaljerade beskrivningar hänvisar vi till [10, 40].

3.1.1 Systemmodell

Kanalen mellan sändare och mottagare antas addera impulsaktigt brus med en statistisk fördelning som varierar över tid och därför inte passar att modelleras som Gaussiskt. En modell som bättre representerar denna brusmiljö är den symmetriskt α-stabila (SαS) modellen [41]. SαS-modellen karaktäriseras bland annat av egenskapen *stabilitet*, vilken representeras av parametern α. Denna stabilitet kan anta värden $0 < \alpha \leq 2$, där ett lägre värde motsvarar mer impulsaktigt brus och vid $\alpha = 2$ överensstämmer modellen med den Gaussiska fördelningen.

Neurala nätverk kan approximera generella funktioner och är därför bra för att ersätta algoritmer hos en mottagare i kommunikationskedjan. I detta arbete ersätts demodulation av ett neuralt nätverk enligt systemskissen i figur 3.1. Modulationen utförs blockvis för en sekvens av komplexa symboler som motsvarar transmissionen i en tidsram. Nätverket utför så kallad *multi-label* klassificering med n klasser för en n bitar lång sekvens, där varje klass motsvarar en bit och sannolikheten att indata tillhör en klass motsvarar sannolikheten att biten för det indexet är ett.

De kodtyper som undersöktes var Hamming(7,4) och LDPC[1] med $1/2$ kodtakt och kodlängd på 240 bitar. Kanalavkodning är möjligt för nätverk att lära sig [42], dock förefaller det svårt att utveckla en algoritm som konvergerar mot en bra lösning för att utföra kombinerad demodulation och kanalavkodning.

Den föreslagna modellen, namngiven Lannet [10], är ett faltande neuralt nätverk (eng. convolutional neural network, CNN) som utför mjuk demodulation. Mjuk demodulation innebär att demodulatorn levererar mjuk information om bitarnas värden till kanalavkodaren, i form av sannolikheter för att en bit är ett respektive noll. Dessa sannolikheter används för att beräkna *log-likelihood ratio* (LLR), vilka är indata till kanalavkodaren [40]. Eftersom neurala nätverk har fasta dimensioner, men Lannet ska kunna utföra demodulation på godtyckligt antal komplexa symboler, så behöver datavektorn bearbetas innan den skickas till nätverket. Givet att antalet symboler är mindre än nätverkets indata-dimensioner, läggs ytterligare symboler till i slutet av indatavektorn. När nätverket demodulerat denna sekvens tas därefter de bitar som motsvarar tillagda symboler bort från utdata. Om antalet symboler istället är större än nätverkets dimension delas sekvensen upp i disjunkta segment som demoduleras var för sig. Dessa segment sammanfogas sedan och skickas till avkodaren.

Under träningen används transmissioner med högt SNR[2], vilka ordnas så att nätverket introduceras till transmissioner med högst SNR först och sedan med successivt lägre SNR. Förhoppningen är att detta ska ge nätverket en möjlighet att först lära sig modulationsschemat och sedan utsättas för data med lägre SNR och därigenom bli mer adaptivt och robust. I denna tillämpning har det visats fungera väl [10]. Algoritmen tränades även för olika värden på α, där α valdes för varje transmission enligt en likformig fördelning. Utförligare beskrivning av träningsmetod och hur denna samt parametrar är framtagna finns dokumenterat i [40, 10].

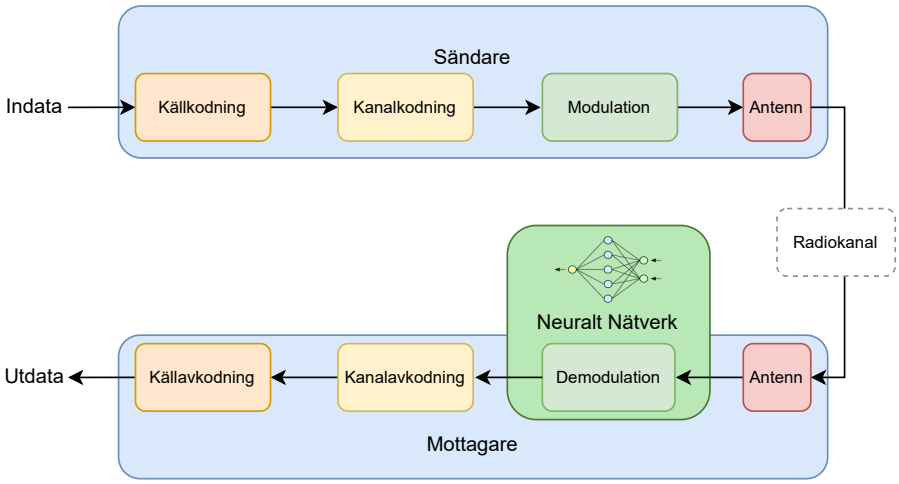
3.1.2 Resultat

Ursprungligen definierades två olika nätverksarkitekturer, benämnda Hannet och Lannet, baserat på vilken felrättande kod som användes. Hannet är anpassad för Hamming(7,4) och Lannet för LDPC med $1/2$ kodtakt och kodordslängd på 240 bitar.

Under utvecklingen av modellerna upptäcktes att olika nätverksarkitekturer tar mer eller mindre hänsyn till den underliggande strukturen hos en felrättande kod, trots att detta inte var intentionen vid träning. Även om arkitekturen för Lannet är den som var minst mottaglig för att lära sig kodstruktur, har Lannet tränats på transmissioner som inte har kanalkodats för att modellen ska fungera så bra som möjligt. Lannet förbättrades så pass att den presterar bättre än Hannet även för Hamming(7,4)-kod. Detta beror på att Hannet presterar bra endast då den tar hänsyn till Hammingkodens struktur men i sig själv presterar den inte lika bra på demodulation som Lannet. Detta kan ses i figur 3.2, vilken visar bitfelshalt (BER) som funktion av SNR. Algoritmerna utvärderas

[1] low density parity check

[2] Med bruseffekt avses medeleffekt för realiseringen av en brussekvens, eftersom variansen inte är väldefinierad för $\alpha \neq 2$

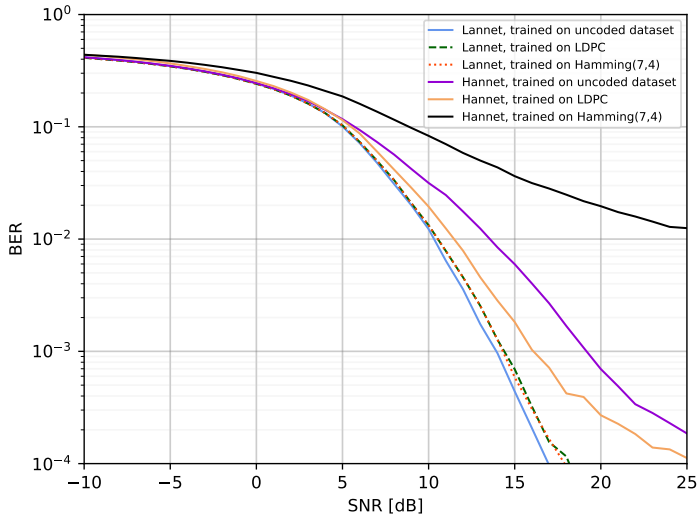


Figur 3.1: Visualisering av en kommunikationskedja, med sändaren överst och mottagaren nederst. Det neurala nätverket ersätter delar av demodulationsblocket.

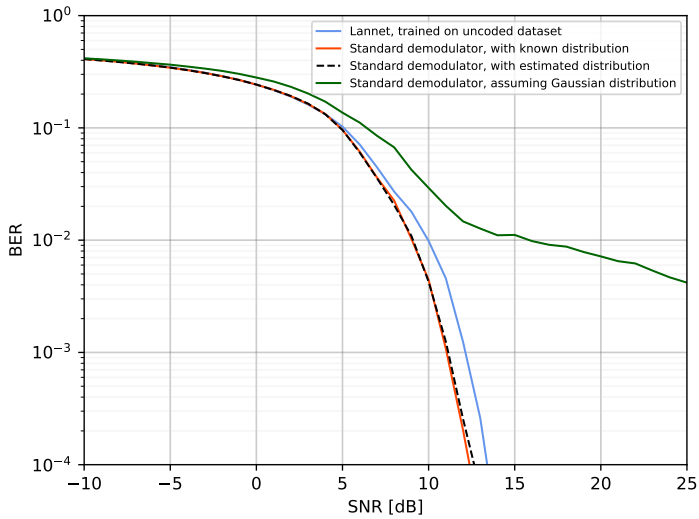
på demodulation av signaler med LDPC-kod, men nätverken har tränats på data med antingen samma kodtyp, med en annan kodtyp eller utan felrättande kod. En slutsats är att det krävs eftertanke vid val av nätverksarkitektur och träning. Framförallt behöver olika alternativ utvärderas för att upptäcka oväntade beteenden.

Vi har även undersökt huruvida nätverken kan utföra flera delar av kommunikationskedjan, i detta fall kanalavkodning och demodulation i ett enda steg. Detta kan ge beräkningsvinster men resultat som presenteras i [10] visar att det är bättre att isolera nätverkets uppgift till att endast utföra demodulation istället för att även utföra avkodning. Det är möjligt att göra båda dessa i kombination, men eftersom nätverken inte lyckas lära sig kodstrukturen fullt ut blir robustheten lägre. Det är eventuellt möjligt att göra detta mer robust genom att träna nätverken i flera steg, exempelvis att först träna ett nätverk att utföra demodulation och sedan träna det att utföra både demodulation och avkodning.

Eftersom Lannet presterar bättre än Hannet, gjordes även jämförelse mellan Lannet och de traditionella algoritmer som den är skapad för att ersätta. I figur 3.3 illustreras att Lannet presterar väl i förhållande till de traditionella, beräkningstunga algoritmerna och presterar bättre än en mindre beräkningstung statistisk standarddemodulator. Slutsatsen är att neurala nätverk är kapabla att approximera de funktioner som de traditionella demodulationsalgoritmerna utför, så att ML-algoritmerna presterar nästintill lika väl som dessa. De ML-baserade lösningarna är inte bundna till någon brusmodell, såsom statistiska algoritmer är, utan de kan anpassas genom att tränas för vilken statistisk fördelning som helst.



Figur 3.2: Bitfelshalt för neurala nätverksbaserade algoritmer som tränats på data med olika kodtyp, utvärderade för LDPC-kod.



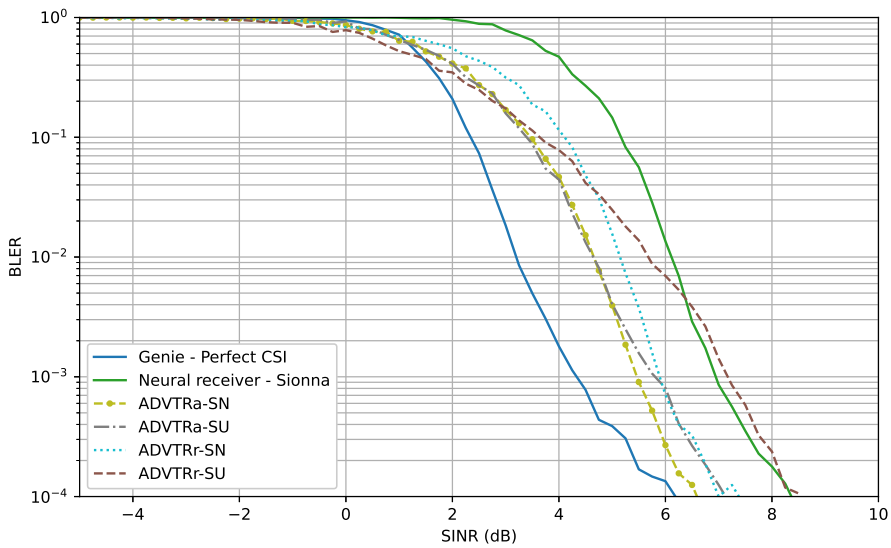
Figur 3.3: Bitfelshalt för Lannet i jämförelse med statistiska algoritmer som antingen känner till brusets fördelning, estimerar fördelningen eller antar Gaussisk fördelning.

3.2 Neural robust mottagare

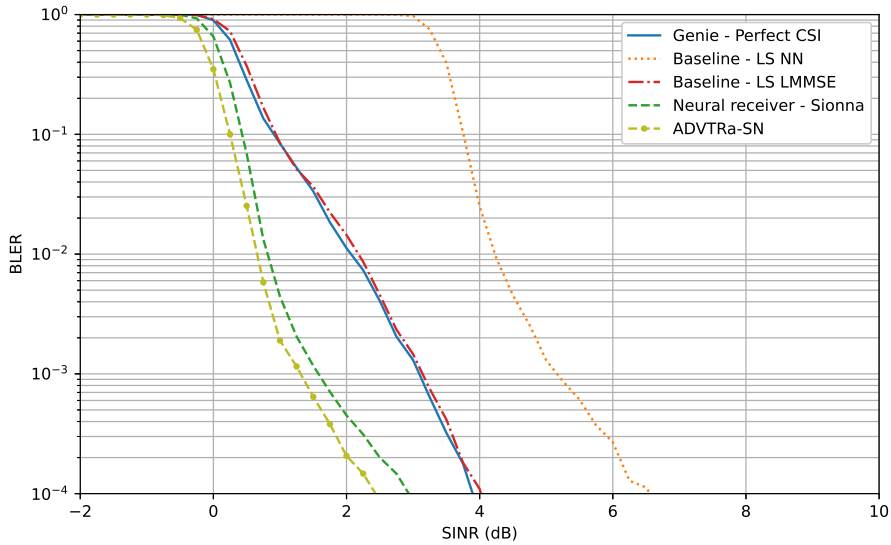
I ML-biblioteket Sionna, se appendix A.2.1 för beskrivning, finns ett exempel på en så kallad neural mottagare som gör kanalestimering och kanalutjämning kombinerat med demodulering i ett neuralt nätverk. Under 2023 genomfördes ett examensarbete [43] på FOI som utgick från detta exempel. I [43] undersöks om den neurala mottagaren från Sionna kan göras robust mot störningsattacker, dels ML-specifika attacker och dels klassisk brusstörning.

En metod kallad *adversarial training* används i [43] för att göra den neurala mottagaren mer tålig mot störning. Metoden går ut på att en delmängd av använda träningsdata utsätts för en attack, i detta arbetet genom *fast gradient sign method* (FGSM) [44]. I träningen av det neurala nätverket lär sig algoritmen att själv undertrycka eller tåla attacken. I arbetet undersöks olika sätt att inkludera attacken i träningsdata samt hur effektförhållandena mellan attack och signal påverkar robustheten i mottagare.

I figur 3.4 visas blockfelshalten som funktion av signal-till-brus-och-interferensförhållandet (SINR) för de olika tränade mottagarna. Som jämförelse visas även en *genie*-mottagare som är optimal med avseende på att den har perfekt kunskap om kanalen (eng. channel state information, CSI) och kan ta bort dess effekter. Effekten på det normalfördelade bruset är konstant sådant att SNR är 10 dB. Vid lägre SINR ökar effekten hos FGSM-attacken och således ökar blockfelshalten. Prestandamässigt är tre av de fyra tränade mottagarna bättre än den ursprungliga mottagaren, men genie-mottagaren bäst för SINR större än 2 dB.



Figur 3.4: Blockfelshalt för neurala mottagare tränade med FGSM-attackerade signaler för SIR i intervallet 2–10 dB och med konstant bruseffekt så att SNR är 10 dB.



Figur 3.5: Blockfelshalt för neurala mottagare utsatta för bredbandig brusstörning med konstant bruseffekt så att SNR är 10 dB.

Figur 3.5 visar blockfelshalten då de olika mottagarna utsätts för en bredbandig brusattack. Bruseffekten bortsett från attacken är konstant sådant att SNR är 10 dB, vilket är den bruseffekten som genie- och baseline-mottagarna antar i sin kanalutjämning och demodulation. I detta fallet är de neurala mottagarna bättre än de klassiska mottagarna. Anledningen till det är att de klassiska mottagarna har en felaktig skattning av bruseffekten eftersom störningen inte är inkluderad i skattningen samt att de neurala mottagarna har tränats på störda signaler.

3.3 Att undvika modulationsklassificering

Maskininlärning används framgångsrikt för att klassificera modulationstypen i en kommunikationssignal i [27, 11]. För en signalspanare är modulationstypen en av många viktiga egenskaper att estimera för att bedöma vilken typ av trådlöst kommunikationssystem som sänder signalen. Med antagandet att signalspanaren använder ML för att klassificera modulationstypen har en studie genomförts för att undersöka om det är möjligt att från sändaren försvåra klassificeringen [45]. Den egna mottagaren bör samtidigt kunna avkoda informationen med låg felsannolikhet. I detta kapitel ges en övergripande beskrivning av den genomförda studien i [45]. För ytterligare detaljer hänvisas till [45].

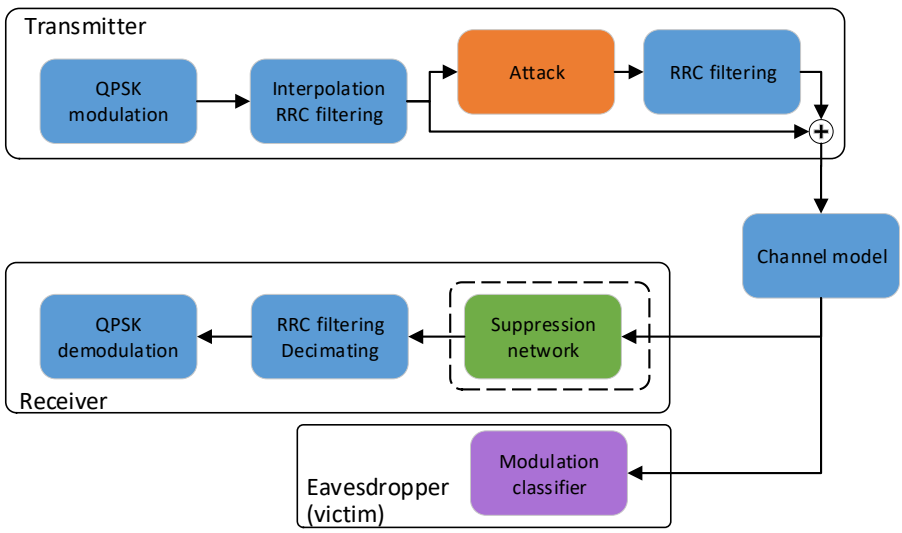
3.3.1 Systemmodell

En förenklad systemmodell antas i vilken kanalen mellan sändare och mottagare är densamma som mellan sändare och signalspanare. Kanalen adderar vitt Gaussiskt brus, inga andra kanaleffekter modelleras. Figur 3.6 visar en skiss av systemet.

I sändaren moduleras informationsbitarna för att sedan filtreras. Därefter genereras en FGSM-attack mot signalspanarens klassificeringsalgoritm. FGSM förutsätter att det neurala nätverk som signalspanaren använder för modulationsklassificering är känt i sändaren, vilket också antas i denna studie. Den genererade attacken filtreras sedan på samma sätt som den modulerade signalen för att erhålla samma spektrala egenskaper, och adderas till den modulerade signalen.

På mottagarsidan har ett neuralt nätverk (kallat SUPNET) utvecklats med syftet att undertrycka attacken som genererats av sändaren. Om attacken inte undertrycks av mottagaren kommer bitfelshalten att öka på grund av attacken.

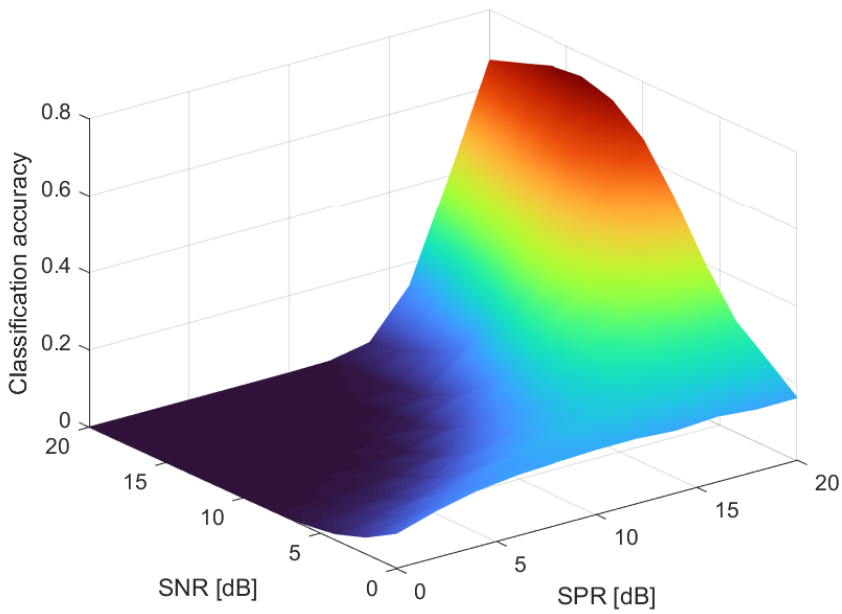
Signalspanaren antas använda en ML-baserad modulationsklassificerare enligt den som presenteras i [27]. Klassificeraren är utvecklad för att skilja mellan elva olika modulationstyper: WB-FM, AM-DSB, AM-SSB, CPFSK, GFSK, 4-PAM, BPSK, QPSK, 8-PSK, 16-QAM och 64-QAM.



Figur 3.6: Skiss av systemmodell för att undvika modulationsklassificering.

3.3.2 Resultat

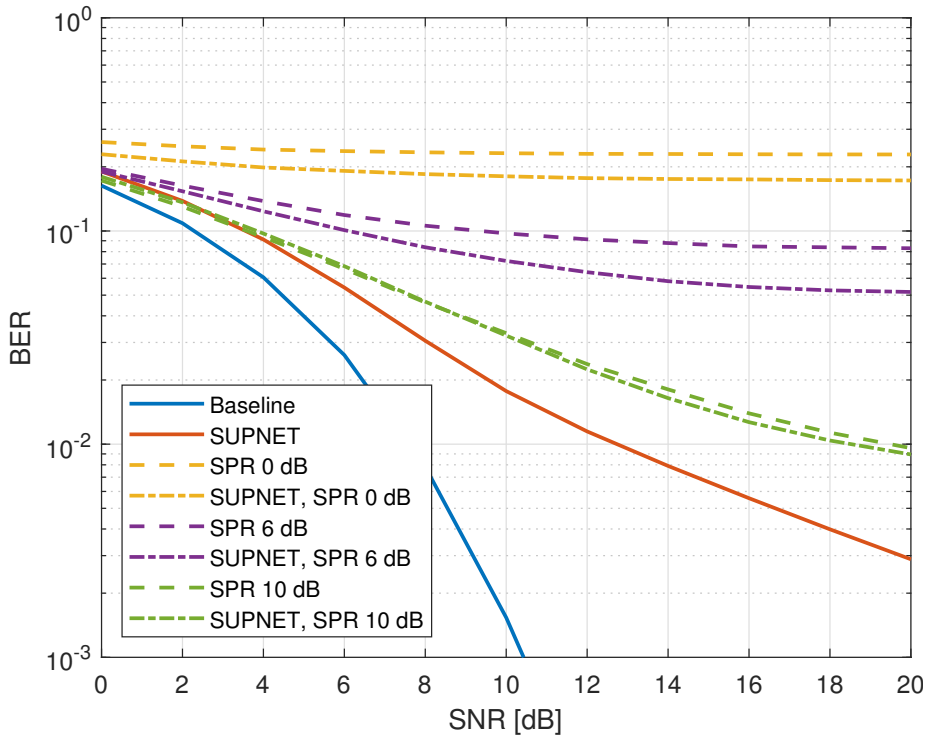
Prestanda, uttryckt i sannolikheten för korrekt klassificering av den QPSK-modulerade signalen, har utvärderats då modulationsklassificeraren utsätts för en FGSM-attack adderad på kommunikationssignalen. Prestanda påverkas av signal-till-brus-förhållandet (SNR) och signal-till-attack-förhållandet (SPR). Figur 3.7 visar klassificeringssannolikheten som funktion av SNR (E_S/N_0) och SPR (E_S/E_P). För SPR mindre än 10 dB är klassificeringssannolikheten mindre än 20 % för alla utvärderade SNR-värden. Signalspanarens prestanda ökar då SPR och SNR ökar, vilket beror på att signalen då blir starkare i förhållande till attacken och bruset. Från figur 3.7 är slutsatsen att FGSM-attacken försvårar för signalspanaren att bestämma vilken modulationstyp som används.



Figur 3.7: Sannolikhet för korrekt klassificering av QPSK-modulerade symboler för signalspanaren.

På mottagarsidan är målet att SUPNET ska minimera bitfel orsakade av att en attacksignal adderas. I figur 3.8 visas bitfelshalten i mottagaren som en funktion av SNR för olika fall med eller utan användning av SUPNET samt för olika SPR.

Utifrån figur 3.8 är slutsatsen att användningen av SUPNET ökar bitfelshalten när en attack inte adderas. Om sändaren adderar en attacksignal minskar däremot SUPNET bitfelshalten jämfört med motsvarande system utan SUPNET. Det finns utvecklingspotential i SUPNET för att öka dess förmåga att undertrycka attacken.



Figur 3.8: Bitfelshalt för mottagare utsatta för attacksignal med olika effekt, med eller utan SUPNET samt jämfört med en referensmottagare.

4 Allokering av tidluckor

I radionät med flera användare finns olika typer av resurser som behöver allokeras adaptivt mellan användarna i radionätet. Dessa resurser kan bland annat bestå av frekvenstilldelningar, nyttjanderätt av kanalen samt effekt och datatakt vid sändning. Allokering av dessa resurser behöver ske på ett sätt som gynnar radionätet globalt, trots att besluten typiskt fattas enskilt av varje användare baserat på dessas uppfattning om radionätets tillstånd. Distribuerade algoritmer för resurstilldelning i radionät blir ofta komplexa och de lösningar som finns bygger i stor utsträckning på designade, regelstyrda protokoll. Detta är en typ av problem där maskininlärning kan vara en tillämpbar metod för att hitta bättre lösningar än de befintliga.

Ett specifikt exempel på resursallokering som studerats är fördelning av tidluckor i radionät som använder tidluckebaserad tilldelning (eng. time-division multiple access, TDMA) av radiokanalen. Kanalanvändning styrs av en ram med tidluckor och ett schema bestämmer vilken nod som sänder i vilken tidlucka. För att utnyttja kanalen effektivt bör fördelningen av tidluckor åtminstone ta hänsyn till användarnas trafiklaster. Vidare kan det finnas användningsfall i vilka trafik med exempelvis olika prioritet och fördröjningskrav behöver hanteras.

4.1 Relaterat arbete

En litteratursökning för att hitta publicerade arbeten som relaterar till resursallokering i radionät med hjälp av maskininlärning genomfördes som ett första steg i arbetet. De arbeten som bedömts mest relevanta har gemensamt att de i olika former bygger på förstärkt inlärning (eng. reinforcement learning, RL). Vid användning av RL tränas en så kallad agent att fatta beslut med hjälp av en policy i syfte att maximera en belöning. RL passar generellt bra för problem där den optimala lösningen är okänd, när det är svårt att hitta en tydlig koppling mellan in- och utsignal eller när det är svårt att generera träningsdata. Dessa förutsättningar stämmer väl överens med tidluckeallokering och därför har fokus legat på just RL-algoritmer. För tidluckeallokering kan indata till algoritmen vara nodernas trafiklaster och utdata en fördelning av tidluckor.

I [46] presenteras ett problem i vilket en RL-agent lär sig hur ett ALOHA-baserat system fungerar. Q-learning, den enklaste formen av RL, används för att lära agenten när den ska sända. Agenten belönas om sändningen lyckas och straffas vid kollisioner.

En metod som presenteras i [47] bygger på att bryta ner några befintliga MAC[1]-protokoll i ett antal byggblock. En RL-agent tränas sedan i att kombinera ett eller flera byggblock till ett protokoll som passar det aktuella scenariot. Tekniken illustreras med ett exempel, i vilket byggblocken utgörs av funktioner i protokoll som baseras på att noder lyssnar på kanalen för att se om den är ledig för att sedan sända (exempel på sådana protokoll är CSMA[2] och ALOHA).

Ett arbete som ligger närmare problemet med fördelning av tidluckor presenteras

[1] medium access control
[2] carrier sense multiple access

i [48], i vilket kommunikation mellan mobila användare och en basstation undersöks. Förutsättningen är att tiden är uppdelad i tidluckor och användarna ska sända data till basstationen på så få tidluckor som möjligt. Djup förstärkt inlärning (eng. deep reinforcement learning, DRL) används, i vilken den tabell som används i så kallad Q-learning ersätts av ett neuralt nätverk. Användarna ersätts med agenter och tränas i att hitta en policy för att välja tidluckor samt utarbeta ett kommunikationsprotokoll användarna emellan för att undvika kollisioner på kanalen. I [48] utvärderas enbart ett scenario med två användare och en basstation medan det i [49] utökas till ett scenario med upp till fem användare. I fallet med fem användare noterar författarna problem med skalbarhet i det framtagna protokollet.

En liknande problemställning betraktas i [50], i vilket schemaläggaren i en 5G-basstation ersätts med en DRL-baserad agent. Indata till schemaläggaren är ett antal vektorer, med samma längd som antalet mobila användare, som beskriver bland annat användarnas paketbuffertlängder och kanalestimat. Schemaläggaren lär sig en policy för att tilldela ett resursblock till en viss användare baserat på indata. En intressant aspekt av denna metod är att schemaläggaren enbart betraktar ett resursblock åt gången vilket innebär att olika storlek på ramar av resursblock kan hanteras utan omträning. En storhet som inte är dynamisk är antalet användare, vilket förutsätts vara konstant. I [51] av samma författare anpassas nätverket till att innehålla ett så kallat *pointer network* för att hantera att antalet användare, och därmed dimensionen på indata, varierar. Resultaten visar att en schemaläggare tränad för ett givet antal användare presterar bra även i scenarier med ett annat antal användare.

Det finns ett stort överlapp mellan att schemalägga tidluckor i en ram och att schemalägga resursblock i nerlänken i 5G. Det är därför möjligt att den lösning som presenteras i [51] skulle kunna anpassas till ett taktiskt nätscenario för att få en lösning som dynamiskt kan hantera olika ramlängder och nodantal. En skillnad är dock att schemaläggning behöver ske distribuerat i ett taktiskt nät, jämfört med en central schemaläggare i ett cellulärt nät, vilket kan ge ytterligare utmaningar.

4.2 Utmaningar

Det finns flera utmaningar att lösa kopplat till att utveckla ML-baserade algoritmer för resursallokering för jämförelse med klassiska protokoll. Till att börja med existerar det få, om några, allmänt accepterade referensprotokoll för taktiska nät att jämföra framtagna ML-algoritmer med. En anledning att fördelning av tidluckor i en TDMA-ram valdes som specifikt problem är att det finns av FOI utvecklade protokoll som kan agera referens. Ett annat hinder är bristen på tillgängliga träningsdata. Till skillnad från andra områden, som exempelvis modulationsklassificering, finns inga befintliga träningsdatamängder att utgå från. Tidluckeallokering har dessutom en mer komplicerad relation mellan in- och utdata än modulationsklassificering.

Skalbarhet riskerar också att bli ett hinder. I [49] undersöks exempel som är kraftigt förenklade, med enbart ett fåtal noder, och författarna pekar på problem med att skala upp till scenarier med fler noder. Idealt vore det möjligt att träna fram en policy i ett

litet nät för att sedan använda den policyn i ett större nät. Om det istället krävs att policyn tränas med flera agenter samtidigt (eng. multi-agent RL) [48, 49] är det stor risk att beräkningskomplexiteten blir för stor.

Det är oklart hur generella de utvecklade protokollen blir och hur träningen bör utföras för att få ett så användbart protokoll som möjligt. Det finns flera frågeställningar som bör studeras vidare inom detta område, exempelvis:

- Hur beroende blir protokollet av antaganden på exempelvis kanalmodell och antal noder?
- Kan agenter som inte tränas tillsammans samarbeta och förstå varandras protokoll?
- Är den tränade modellen unik för varje nod, eller kan samma modell överföras mellan noder?
- Är det möjligt att få fram ett fungerande protokoll endast genom förträning eller måste protokollet uppdateras kontinuerligt?
- Hur kan ett utvecklat protokoll förklaras?
- Kan protokollet brytas ner till regler som är lätta att förstå?

5 Slutsatser

Maskininlärning tillämpat inom radiokommunikationsområdet är fortfarande att betrakta som ungt jämfört med andra områden, exempelvis bildigenkänning och språkbehandling. Möjligheterna och sårbarheterna med ML i radiosystem är idag svåra att förutse, men med tanke på utvecklingen av ML inom andra områden finns potentiellt en enorm möjlighet att exploatera. Mer forskning behöver bedrivas inom ML för radiokommunikation för att nyttja dess fulla potential, särskilt för att erhålla robusta radiosystem. I detta kapitel presenteras slutsatser relaterat till de tre frågeställningar som rapporten adresserar.

Hur bör träningsdata av tillräcklig kvalitet och kvantitet skapas för att garantera robust kommunikation? Träningsdata av tillräcklig mängd och kvalitet är avgörande för hur väl en ML-algoritm presterar på nya, okända data. Träningsdatasetet bör vara av tillräcklig storlek och variation för att undvika över- och underträning samt ha en tillräckligt balanserad mångfald av den typ av data som är relevant för applikationen.

Valet att skapa träningsdata genom mätningar, att syntetisera ett eget simulerat dataset eller att nyttja ett existerande, publikt dataset är inte självklart och medför både för- och nackdelar. Att mäta egna, verkliga data i den mängd som behövs är svårt, men har fördelen att fånga upp verkliga signalbeteenden. Syntetiska data är lättare att skapa en tillräcklig mängd av, men det är svårt att modellera komplicerade signal- och kanaleffekter. Både verkliga och syntetiska data riskerar att omfattas av sekretess om militära särkrav inkluderas.

Att nyttja existerande dataset sparar tid och arbete, och har fördelen att det är möjligt att jämföra resultat med övriga forskarvärlden som utvärderar modeller på samma dataset. Med publika dataset är det svårt att ha kontroll över hur data har konstruerats. De kan dessutom innehålla, avsiktliga eller oavsiktliga, felaktigheter.

I en rekommendation för att skapa träningsdata av tillräcklig kvalitet och kvantitet för robust radiokommunikation måste ett antal utmaningar tas hänsyn till. Träningsdatasetet måste representera den signalmiljö som modellen förväntas verka i. För att utöka datamängden då tillräckligt stor mängd saknas kan datautökning nyttjas, men det resulterande datasetet måste valideras med relevanta metoder. För att underlätta hantering, indexering och återskapande av ML-algoritmer bör generering av dataset följa gemensamma riktlinjer, exempelvis FAIR-systemet eller motsvarande verksamhetsspecifika riktlinjer.

Hur kan robusthet analyseras och förbättras för ML-baserade algoritmer avseende både traditionella störhot och ML-anpassade attacker? Algoritmer baserade på neurala nätverk har visats kunna utföra adaptiv demodulation i impulsrik och varierande interferensmiljö med god prestanda och lägre beräkningskomplexitet än traditionella algoritmer. Studien visade att neurala nätverk tar hänsyn till den underliggande strukturen hos en felrättande kod i varierande utsträckning beroende på deras nätarkitektur. En slutsats är att det är lättare att uppnå bra prestanda med neurala nätverk som utför endast demodulation, och därefter traditionell avkodning, än att utföra kombine-

rad demodulation och avkodning.

Genom att inkludera ML-attacker i träningen av neurala nätverk reduceras sårbarheterna. En mottagare som utför kanalestimering, kanalutjämning och demodulering i ett neuralt nätverk har gjorts mer robust mot störning genom denna princip. Robustheten i den neurala mottagaren påverkas av hur ML-attacker inkluderas i träningen samt effektförhållandet mellan signal och attack.

Sårbarheter i ML-baserade system kan innebära dels en sårbarhet för det egna systemet och dels utnyttjas mot en antagonist. För att undvika att modulationstypen hos den egna kommunikationssignalen klassificeras av en antagonist (t.ex. signalspaning) kan sårbarheter i en ML-baserad modulationsklassificerare utnyttjas. Den av sändaren applicerade attacken undertrycks av den egna mottagaren för att inte försämra bitfelshalten i mottagningen. Simuleringar visar att principen fungerar i avseendet att modulationsklassificeringen får försämrad noggrannhet samtidigt som den egna mottagaren får något bättre bitfelshalt genom undertryckning av den adderade attacken. Tekniken behöver dock vidareutvecklas för att få erforderlig prestanda.

Hur kan ML-teknik nyttjas för effektivare resursallokering i mobila radionät? Resursallokering i mobila radionät är ett område av problem som lämpar sig väl för ML-baserade lösningar, då de optimala lösningarna till problemen typiskt är av sådan komplexitet att heuristiska metoder krävs. Vi har fokuserat på radioresursallokering i form av tidlucketilldelning som ett specifikt problem. Ett flertal arbeten har publicerats på senare tid kring kanalaccess med huvudsakligt fokus på 5G och 6G. RL lyfts fram som en variant av ML som är väl lämpad för dessa frågeställningar och det finns lovande resultat publicerade för RL-baserade schemaläggare i 5G-nät. Den schemaläggning som sker i nedlänken i ett 5G- eller 6G-nät påminner i många avseenden om schemaläggning av tidluckor i ett mobilt radionät, dock sker beslutsfattandet centraliserat. Hur dessa tekniker kan översättas till mobila radionät, i vilka schemaläggning av resurser sker distribuerat, är ett arbete som påbörjats och kommer att fortsätta i kommande forskningsprojekt på FOI.

Referenser

[1] X. Yuan, P. He, Q. Zhu och X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, nr. 9, s. 2805–2824, 2019.

[2] D. Adesina, C.-C. Hsieh, Y. E. Sagduyu och L. Qian, "Adversarial machine learning in wireless communications using RF data: A review," *IEEE Commun. Surveys Tuts.*, vol. 25, nr. 1, s. 77–100, 2023.

[3] E. Axell, P. Eliardsson, K. Hägglund, P. Brännström och C. Svensson, "Overview of machine learning in communication systems," Totalförsvarets forskningsinstitut, FOI-R--5275--SE, dec. 2021.

[4] ——, "Adversarial machine learning in wireless communications - basics and two examples," Totalförsvarets forskningsinstitut, FOI-R--5427--SE, jan. 2023.

[5] G. James, D. Witten, T. Hastie och R. Tibshirani, *An Introduction To Statistical Learning with Applications in R*. Springer, 2022.

[6] T. Pollard och G. Peru, "Introduction to machine learning in Python (Carpentries Incubator)." [Online]. URL: https://github.com/carpentries-incubator/machine-learning-novice-python

[7] A. Lindholm, N. Wahlström, F. Lindsten och T. Schön, *Machine learning: a first course for engineers and scientists*. Cambridge, United Kingdom: Cambridge University Press, 2022.

[8] L. Huang, W. Pan, Y. Zhang, L. Qian, N. Gao och Y. Wu, "Data augmentation for deep learning-based radio modulation classification," 2019.

[9] J. Ding och X. Li, "An approach for validating quality of datasets for machine learning," *Proc. IEEE Int. Conf. on Big Data (Big Data)*, 2018, s. 2795–2803.

[10] A. Andersson, "Utilizing neural networks to adaptively demodulate and decode signals in an impulsive environment," Masters Thesis, Uppsala universitet, jun 2023.

[11] T. J. O'Shea, T. Roy och T. C. Clancy, "Over-the-air deep learning based radio signal classification," *IEEE J. Sel. Topics Signal Process.*, vol. 12, nr. 1, s. 168–179, 2018.

[12] A. Engelbrecht, *Computational Intelligence*. John Wiley and Sons Ltd, 2007.

[13] D. Gustafsson *et al.*, "Maskininlärning inom försvarorienterade tillämpningar - nya ansatser och experimentella resultat," Totalförsvarets forskningsinstitut, FOI-D--0999--SE, nov. 2020.

[14] F. Kamrani *et al.*, "Attacking and deceiving military systems," Totalförsvarets forskningsinstitut, FOI-R--5396--SE, 2023.

[15] Y. Si, W. Zhou och J. Gai, "Research and implementation of data extraction method based on NLP," *in Proc. IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, 2020, s. 11–15.

[16] L. Sixt, B. Wild och T. Landgraf, "RenderGAN: Generating realistic labeled data," 2017. [Online]. URL: http://arxiv.org/abs/1611.01331

[17] I. J. Goodfellow *et al.*, "Generative adversarial networks," 2014. [Online]. URL: https://arxiv.org/abs/1406.2661

[18] L. Oakden-Rayner, "Exploring large scale public medical image datasets," 2019. [Online]. URL: https://arxiv.org/abs/1907.12720

[19] J. Ding, J. Wang, X. Kang och X.-H. Hu, "Building an SVM classifier for automated selection of big data," *Proc. IEEE International Congress on Big Data (BigData Congress)*, 2017, s. 15–22.

[20] J. Ding, X. Kang, X.-H. Hu och V. Gudivada, "Building a deep learning classifier for enhancing a biomedical big data service," *Proc. IEEE International Conference on Services Computing (SCC)*, 2017, s. 140–147.

[21] J. Mange, "Effect of training data order for machine learning," *Proc. Int. Conf. on Computational Science and Computational Intelligence (CSCI)*, 2019, s. 406–407.

[22] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, s. 2825–2830, 2011.

[23] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Process. Mag.*, vol. 29, nr. 6, s. 141–142, 2012.

[24] M. Wilkinson *et al.*, "The FAIR guiding principles for scientific data management and stewardship," *Nature Genetics, Scientific Data*, vol. 48, nr. 4, april 2016.

[25] T. O'Shea och N. West, "Radio machine learning dataset generation with GNU radio," *Proc. GNU Radio Conference*, vol. 1, nr. 1, 2016. [Online]. URL: https://pubs.gnuradio.org/index.php/grcon/article/view/11

[26] T. A. Hall, R. Caromi, M. Souryal och A. Wunderlich, "Reference datasets for training and evaluating RF signal detection and classification models," *Proc. IEEE Globecom Workshops (GC Wkshps)*, 2019, s. 1–5.

[27] T. J. O'Shea, J. Corgan och T. C. Clancy, "Convolutional radio modulation recognition networks," *Proc. Int. Conf. Engineering Applications of Neural Networks*, 2016.

[28] C. d. Vrieze, L. Simic och P. Mahonen, “The importance of being earnest: Performance of modulation classification for real RF signals,” *Proc. IEEE Int. Dynamic Spectrum Access Networks (DySPAN) symposium*, 2018, s. 1–5.

[29] L. Chen, S. T. Jose, I. Nikoloska, S. Park, T. Chen och O. Simeone, “Learning with limited samples – meta-learning and applications to communication systems,” 2022.

[30] T. Hospedales, A. Antoniou, P. Micaelli och A. Storkey, “Meta-learning in neural networks: A survey,” 2020.

[31] C. T. Nguyen *et al.*, “Transfer learning for wireless networks: A comprehensive survey,” *Proceedings of the IEEE*, vol. 110, nr. 8, s. 1073–1115, 2022.

[32] U. Challita, H. Ryden och H. Tullberg, “When machine learning meets wireless cellular networks: Deployment, challenges, and applications,” *IEEE Commun. Mag.*, vol. 58, nr. 6, s. 12–18, 2020.

[33] M. Herlich och S. Farthofer, “Wireless communications data sets for machine learning,” Salzburg Research, 2020.

[34] T. Hossfeld och S. Wunderer, “White paper on crowdsourced network and QoE measurements – definitions, use cases and challenges,” Universität Würzburg, 2020. [Online]. URL: https://opus.bibliothek.uni-wuerzburg.de/20232

[35] Machine learning for communications emerging technologies initiative. [Online]. URL: https://mlc.committees.comsoc.org/datasets/

[36] L. J. Wong, S. McPherson och A. J. Michaels, “Assessing the value of transfer learning metrics for RF domain adaptation,” 2022. [Online]. URL: https://arxiv.org/abs/2206.08329

[37] DeepSig, Inc. RF datasets for machine learning. [Online]. URL: https://www.deepsig.ai/datasets

[38] K. Tekbiyik, O. Akbunar, A. R. Ekti, A. Görcin, G. K. Kurt och K. A. Qaraqe, “Spectrum sensing and signal identification with deep learning based on spectral correlation function,” *IEEE Trans. Veh. Technol.*, vol. 70, nr. 10, s. 10 514–10 527, 2021.

[39] K. Hägglund och E. Axell, “Adaptive demodulation in impulse noise channels,” *IEEE Trans. Veh. Technol.*, vol. 71, nr. 2, s. 1685–1698, 2022.

[40] A. Andersson, K. Hägglund och E. Axell, “Deep learning-based demodulation in impulse noise channels,” *Proc. IEEE Mil. Commun. Conf. MILCOM*, nov. 2023.

[41] C. Nikias och M. Shao, *Signal Processing with Alpha-stable Distributions and Applications*. New York, NY, USA: Wiley-Interscience, 1995.

[42] T. Gruber, S. Cammerer, J. Hoydis och S. ten Brink, "On deep learning-based channel decoding," *Proc. Annual Conference on Information Sciences and Systems (CISS)*, 2017, s. 1–6.

[43] A. Nicklasson Cedbro, "Robust neural receiver in wireless communication: Defense against adversarial attacks," examensarbete, Linköping University, Communication Systems, 2023.

[44] I. Goodfellow, J. Shlens och C. Szegedy, "Explaining and harnessing adversarial examples," *International Conference on Learning Representations*, 2015. [Online]. URL: http://arxiv.org/abs/1412.6572

[45] P. Eliardsson och E. Axell, "Adversarial machine learning to avoid modulation classification," Totalförsvarets forskningsinstitut, FOI-D--1237--SE, nov. 2023.

[46] Y. Chu, P. D. Mitchell och D. Grace, "ALOHA and Q-learning based medium access control for wireless sensor networks," *Proc. International Symposium on Wireless Communication Systems (ISWCS)*, 2012, s. 511–515.

[47] H. B. Pasandi och T. Nadeem, "Challenges and limitations in automating the design of MAC protocols using machine-learning," *Proc. Int. Conf. on Artificial Intelligence in Information and Communication (ICAIIC)*, 2019, s. 107–112.

[48] M. P. Mota, A. Valcarce, J.-M. Gorce och J. Hoydis, "The emergence of wireless MAC protocols with multi-agent reinforcement learning," *Proc. IEEE Globecom Workshops (GC Wkshps)*. IEEE, dec 2021.

[49] M. P. Mota, A. Valcarce och J.-M. Gorce, "Scalable joint learning of wireless multiple-access policies and their signaling," *Proc. IEEE Vehicular Technology Conf. (VTC2022-Spring)*, 2022.

[50] F. Al-Tam, N. Correia och J. Rodriguez, "Learn to schedule (LEASCH): A deep reinforcement learning approach for radio resource scheduling in the 5G MAC layer," *IEEE Access*, vol. 8, 2020.

[51] F. AL-Tam, A. Mazayev, N. Correia och J. Rodriguez, "Radio resource scheduling with deep pointer networks and reinforcement learning," *Proc. IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2020, s. 1–6.

[52] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. URL: http://download.tensorflow.org/paper/whitepaper2015.pdf

[53] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," *in Proc. International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[54] S. Chintala. PyTorch strengthens its governance by joining the Linux foundation. [Online]. URL: https://pytorch.org/blog/PyTorchfoundation/

[55] J. Bradbury *et al.* (2018) JAX: composable transformations of Python+NumPy programs. [Online]. URL: http://github.com/google/jax

[56] F. Chollet *et al.* (2015) Keras. [Online]. URL: https://keras.io

[57] Mathworks. Deep learning toolbox. [Online]. URL: https://se.mathworks.com/products/deep-learning.html

[58] J. Hoydis *et al.*, "Sionna: An open-source library for next-generation physical layer research," 2023. [Online]. URL: https://arxiv.org/abs/2203.11854

[59] M.-I. Nicolae *et al.*, "Adversarial robustness toolbox v1.0.0," 2018. [Online]. URL: https://arxiv.org/abs/1807.01069

[60] N. Papernot *et al.*, "Technical Report on the CleverHans v2.1.0 Adversarial Examples Library," 2018. [Online]. URL: https://arxiv.org/abs/1610.00768

A Maskininlärningsverktyg

Det finns ett stort antal verktyg och programvarubibliotek för maskininlärningstillämpningar. Flera av biblioteken är fritt tillgängliga (öppen källkod och tillåtande licenser) och ofta står mjukvarujättar som Facebook, Google eller Microsoft bakom projekten. Flera av verktygen erbjuder liknande funktionalitet och riktar sig mot samma programspråk, huvudsakligen Python och C++. I detta kapitel ges en översikt av de mest använda biblioteken.

A.1 Bibliotek för generell maskininlärning

Det finns flera mjukvarubibliotek som är utvecklade specifikt för utveckling av maskininlärningteknik. De vanligast använda verktygen beskrivs i detta avsnitt. Verktygen kan användas för att lösa flertalet maskininlärningsuppgifter, exempelvis

- att lösa bildigenkänningsproblem avseende att detektera objekt i bilder och filmer med hjälp av faltningsnätverk (eng. *convolutional neural network*, (CNNs)) och Tensorflows API för detektion av objekt
- språkbearbetningsproblem för att förstå och generera språk, exempelvis textklassificering, attitydanalys och maskinöversättning
- taligenkänning för att konvertera tal till text eller det omvända
- anomalidetektion av stora datamängder, exempelvis systemövervakning och bedrägerier
- förutspå tidsserier, exempelvis efterfrågan av produkter, aktiepriser, väderprognoser med hjälp av så kallade *long short-term memory* (LSTM) nätverk
- förstärkningsinlärning (eng. *reinforcement learning* (RL)) för att lära en dator att fatta en sekvens av beslut, exempelvis spela spel eller styra robotar
- generativa modeller för att skapa nya data, exempelvis bilder eller texter.

A.1.1 TensorFlow

TensorFlow är ett gratis mjukvarubibliotek för maskininlärning utvecklat av Google Brain [52]. Mjukvarubiblioteket är så kallad öppen källkod och licensierad under Apache License Version 2.0. TensorFlow har programmeringsgränssnitt (API) till ett flertal stora programspråk såsom Python, JavaScript, C++ och Java. Gränssnittet mot Python är det mest fullständiga och får de senaste uppdateringarna först. Bakåtkompatibilitet garanteras för Python men inte för de övriga språken.

I TensorFlow representeras maskininlärningsmodellen som matematiska grafer där noderna i grafen representerar matematiska operationer och bågarna representerar dataflöden. Detta gör arkitekturen flexibel och att beräkningar kan ske parallellt för flera

datorer i beräkningskluster och i en maskin med flera beräkningsenheter, exempelvis flerkärniga processorer (CPU), grafikkort (GPU) och integrerade kretsar (ASIC) kallade *tensor processing units* (TPU).

Mjukvarubiblioteket är fortfarande under utveckling. Mellan version 1 och 2 gjordes stora förändringar i ramverket vilket gör att det saknas kompatibilitet mellan dessa versioner.

A.1.2 PyTorch

PyTorch är ett annat kostnadsfritt mjukvarubibliotek för maskininlärning utvecklat av Facebook AI Research Lab (nu Meta AI) [53]. Även PyTorch består av öppen källkod och licensieras under BSD-3. Sedan september 2022 är PyTorch en del av Linux Foundation [54]. PyTorch tillhandahåller API för Python samt C++. PyTorch är designat för att vara likt Python och kompatibelt med övriga paket till Python. Mjukvarubiblioteket är under fortsatt utveckling.

A.1.3 JAX

JAX är ett bibliotek för Python som i sin tur är en sammanslagning av två tidigare separata bibliotek: Autograd som var ett bibliotek för automatisk differentiering och XLA som står för *accelerated linear algebra*. JAX utvecklas av Google, består av öppen källkod och licensieras under Apache License Version 2.0 [55]. Automatisk differentiering är en effektiv metod för att räkna ut gradienter av funktioner med bra numerisk precision. Eftersom användning av neurala nätverk i hög grad bygger på uträkningar av gradienter ger effektivare beräkningar av gradienter stora tidsvinster vid träning av neurala nätverk. Ett Python-bibliotek kallat Flax, som också underhålls av Google, bygger på JAX och tillhandahåller färdiga moduler för att bygga neurala nätverk, liknande PyTorch och Keras (se avsnitt A.1.4).

A.1.4 Keras

Keras är ett gränssnitt mot andra maskininlärningsbibliotek och bygger på öppen källkod som licensieras under Apache License Version 2.0 [56]. Beroende på programvaruversion stöds olika mjukvarubibiliotek. Upp till version 2.3 stöds TensorFlow, Microsoft Cognitive Toolkit, Theano och PlaidML. Med version 2.4 stöds endast TensorFlow. Med utvecklingen till 3.0 stöds mjukvarubiblioteken TensorFlow, JAX och PyTorch. Syftet med Keras är att göra tillvaron enklare för utvecklare genom att koden ska bli färre rader, mer lättöverskådlig och snabbare att felsöka. Kerasblock fungerar sömlöst ihop med Tensorflow-block. Keras är sedan version 2 av Tensorflow en integrerad del av Tensorflow.

A.1.5 Matlab

Till Mathworks Matlab finns en *Deep learning toolbox* (ett extra tillägg) som är en proprietär mjukvara och kräver en betald licens [57]. Med toolboxen är det i Matlab möjligt att lösa liknade maskininlärningsproblem som kan lösas med t.ex. TensorFlow och PyTorch. Det finns möjligheter att importera modeller från andra verktyg, exempelvis TensorFlow och PyTorch, eftersom toolboxen stödjer Open Neural Network Exchange (ONNX), vilket är ett standardiserat format för neurala nätverk. För att dra nytta av beräkningskraften från GPU:er och parallell programmering krävs en toolbox för parallellberäkningar, vilken också är proprietär mjukvara som kräver en betald licens.

A.2 Bibliotek för specifik ML-tillämpning

Förutom de generella ML-biblioteken finns det dessutom mjukvarubibliotek som är utvecklade för utveckling av maskininlärningteknik inom specifika delområden. Detta avsnitt beskriver bibliotek som är relevant för studier av robusthet och sårbarhet inom ML-baserad radiokommunikationsteknik.

A.2.1 Sionna

Sionna är ett mjukvarubibliotek baserat på öppen källkod från NVIDIA, licensierad under Apache License Version 2.0 [58]. Sionna är avsett för simuleringar av fysiska lager i trådlösa kommunikationssystem. Mjukvarubiblioteket har många moderna kommunikationsalgoritmer implementerade för jämförelse med maskininlärda alternativ. Syftet med utvecklingen av Sionna var att tillhandahålla ett verktyg för forskning och utveckling av maskininlärda algoritmer till 5G och 6G. De implementerade algoritmerna i Sionna är implementerade i Keras-lager, vilket gör Sionna anpassningsbart och underlättar integrering med exempelvis Tensorflow.

A.2.2 Adversarial Robustness Toolbox (ART)

Adversarial Robustness Toolbox är ett Python-baserat bibliotek för *adversarial machine learning* (AML), dvs. för att skapa ML-attacker (eng. adversarial attacks eller adversarial examples) och försvar mot sådana attacker [59]. Biblioteket innehåller ett stort antal attacker av olika typ och flertalet försvarsalgoritmer. ART finns tillgänglig gratis som öppen källkod[1] under MIT-licens och har stöd för flera av de vanliga ML-biblioteken, t.ex. TensorFlow (v1 och v2), PyTorch och Keras.

[1] https://github.com/Trusted-AI/adversarial-robustness-toolbox

A.2.3 CleverHans

CleverHans är ytterligare ett Python-baserat bibliotek för AML och innehåller både attacker och försvarsmetoder [60]. CleverHans stödjer, sedan version 4.0.0, de tre ramverken TensorFlow version 2, PyTorch och JAX. Källkoden för biblioteket finns tillgänglig under MIT-licens[2]. CleverHans användes tillsammans med Sionna i [43] för att utvärdera AML för en mottagare baserad på neuralt nätverk.

[2] https://github.com/cleverhans-lab/cleverhans

FOI är en huvudsakligen uppdragsfinansierad myndighet under Försvarsdepartementet. Kärnverksamheten är forskning, metod- och teknikutveckling till nytta för försvar och säkerhet. Organisationen har cirka 1000 anställda varav ungefär 800 är forskare. Detta gör organisationen till Sveriges största forskningsinstitut. FOI ger kunderna tillgång till ledande expertis inom ett stort antal tillämpningsområden såsom säkerhetspolitiska studier och analyser inom försvar och säkerhet, bedömning av olika typer av hot, system för ledning och hantering av kriser, skydd mot och hantering av farliga ämnen, IT-säkerhet och nya sensorers möjligheter.

FOI
Totalförsvarets forskningsinstitut
164 90 Stockholm

Tel: 08-55 50 30 00
Fax: 08-55 50 31 00

www.foi.se