Tämja hagelbössan: En domänspecifik optimeringspipeline för generering av spelbar gitarrtabulatur

Ross Ottosson & Ludvig Ölund Danielsson

Lunds Tekniska Högskola

Datum: 18 maj 2026
Handledare: Ester Daniel Ytterbrink
Examinator: Mattias Nordahl
Löpnummer: LU-CS-EX: 2026-17

Abstract

Automatic Music Transcription (AMT) has made significant advancements but transcribing guitar remains a substantial challenge due to the acoustic complexity of the instrument, including overtones, sympathetic resonance, and mechanical transients. This study investigates how a general-purpose AMT model (Spotify's Basic Pitch) can be adapted to automatically generate physically playable guitar tablature. This is done by lowering the model's minimum note length threshold so that fast musical passages are successfully captured, albeit at the cost of massive over-detection of false ghost notes (a "shotgun" effect).

To mitigate this, a domain-specific filtering pipeline was developed and optimized to eliminate unwanted noise based on the physical and acoustic properties of the guitar. The purified notes were subsequently processed using a statistical algorithm to calculate biomechanically ergonomic fingerings.

The results indicate that the customized pipeline performs better than the standard version of Basic Pitch by capturing rapid guitar notes that Basic Pitch typically misses, and by cleaning up a majority of the additional false positive notes that occur as a result of the lower threshold. Furthermore, the study demonstrates that the algorithm for calculating fingerings performs significantly better after many of the false positive notes have been cleaned up. The findings conclude that a strategy of intentional over-detection paired with domain-specific filtering is an effective method for automatic guitar transcription.

Sammanfattning

Automatisk musiktranskribering (AMT) har gjort stora framsteg men att transkribera gitarr är fortfarande en betydande utmaning på grund av instrumentets akustiska komplexitet, inklusive övertoner, sympatisk resonans och mekaniska transienter. Denna studie undersöker hur en generell AMT-modell (Spotifys Basic Pitch) kan anpassas för att automatiskt generera fysiskt spelbar gitarrtabulatur. Detta görs genom att sänka modellens tröskelvärde för minsta notlängd så att snabba musikaliska passager fångas upp till priset av en massiv överdetektering av falska spöknoter (en så kallad "hagelbösse-effekt").

För att hantera detta utvecklades och optimerades en domänspecifik filtreringspipeline för att eliminera oönskat brus baserat på gitarrens fysiska och akustiska egenskaper. De renade noterna bearbetades därefter med en statistisk algoritm för att beräkna biomekaniskt ergonomiska fingersättningar.

Resultaten indikerar att den anpassade pipelinen presterar bättre än standardversionen av Basic Pitch genom att fånga upp snabba gitarrnoter som Basic Pitch i vanliga fall missar, samt genom att städa upp en majoritet av de extra falska positiva noter som tillkommer som följd av det lägre tröskelvärdet. Vidare visar studien att algoritmen för beräkning av fingersättningar presterar avsevärt bättre efter att många av de falska positiva noterna har rensats bort. Slutsatsen är att en strategi med medveten överdetektering parat med domänspecifik filtrering är en effektiv metod för automatisk gitarrtranskribering.

Innehåll

1 Inledning

Att översätta inspelad musik till noter eller tabulatur är en tidskrävande och komplex process som är starkt beroende av en musikers vältränade öra. Med framväxten av maskininlärning och modeller för automatisk musiktranskribering (AMT) öppnas nya dörrar och möjligheten att omedelbart omvandla ljud till spelbara noter blir allt mer en verklighet. Modeller som Spotifys Basic Pitch har visat goda och imponerande resultat för generell musikanalys och kan t.ex plocka ut en pianoslinga med god träffsäkerhet. Men gitarren är ett fysiskt och akustiskt komplext instrument som präglas av övertoner, mekaniska transienter och resonans som gör det mycket svårare för AMT-system att lista ut vilka noter som spelas. Denna studie syftar till att undersöka glappet mellan generell AI-transkribering och gitarrens domänspecifika krav, samt att utveckla en anpassad filtrerings- och optimeringspipeline (inklusive Viterbi algoritm för att beräkna optimal fingersättning) för att automatiskt generera en musikalisk och fysiskt spelbar gitarrtabulatur.

2 Bakgrund

Detta kapitel presenterar den teoretiska bakgrunden och de tekniska verktyg som ligger till grund för studien. Här introduceras bland annat modeller för automatisk musiktranskribering, källseparering och parameteroptimering, samt principerna bakom den Viterbi algoritm som möjliggör den slutgiltiga tabulaturgenereringen.

2.1 Automatisk musiktranskription (AMT) och Basic Pitch

Basic Pitch är ett verktyg för automatisk musiktranskribering (AMT) utvecklat av forskare på Spotify. Modellen använder maskininlärning och neurala nätverk för att analysera ljudsignaler och översätta dessa till MIDI-data. [1] MIDI är ett digitalt protokoll som lagrar musikaliska instruktioner såsom tonhöjd, varaktighet och anslagsstyrka snarare än själva ljudvågen. Genom att representera noter som strukturerad data utgör MIDI ett lämpligt format för att möjliggöra filtrering av felaktiga noter.

2.2 Utvärdering av AMT och mir_eval

Ramverket mir_eval är ett öppet källkods verktyg för att jämföra och utvärdera automatisk musiktranskribering. Detta görs genom att jämföra ett systems utdata på notnivå mot ett facit genom att beräkna överlappning i tid och tonhöjd. Prestandan mäts därefter ofta genom tre mått; Precision, Recall och F1-poäng. Precision anger andelen av de transkriberade noterna från modellen som faktiskt är korrekta. Hög precision innebär att systemet genererar få falska positiva. Recall anger andelen av noterna i facit som systemet lyckades identifiera. Hög recall innebär att systemet missar väldigt få av de faktiskt spelade noterna. F1-poäng utgör det harmoniska medelvärdet av precision och recall. Då en algoritms prioritering av det ena måttet ofta sker på bekostnad av det andra används F1-poängen som ett sammanvägt mått för att beskriva systemets totala noggrannhet. [2]

2.3 Källseparering med HT-Demucs

HT-Demucs är ett verktyg för källseparering som kan dela upp en färdig ljudmix i dess enskilda instrument [3]. I denna studie används det som ett förbehandlingssteg

för att isolera gitarren från övriga instrument i en låt. Genom att separera ut gitarren får AMT-modellen en renare signal att arbeta med vilket minskar risken för att ljud från trummor eller bas felaktigt tolkas som gitarrnoter. Specifikt används modellen htdemucs_6s eftersom det är den version som har stöd för att identifiera och extrahera just gitarr [4].

Demucs används i denna studie som en förtränad komponent utan vidare optimering då fokus ligger på den efterföljande filtreringen och tabulaturgenereringen.

2.4 Optuna - Optimering av parametrar

Optuna är ett generellt ramverk med öppen källkod för hyperparameteroptimering vilket innebär att det kan användas för att optimera parametrar i allt från maskininlärning till matematiska funktioner. Verktyget försöker automatiskt hitta de bästa inställningarna för en algoritm genom att minimera eller maximera en målfunktion. Optuna kan hantera olika typer av parametrar, såsom kontinuerliga flyttal (t.ex. tröskelvärden för volym), heltal (t.ex. fönsterstorlekar) eller kategoriska val (t.ex. val av filtertyp). För denna rapport är en enkel förståelse av hur Optuna fungerar tillräcklig. Optimeringen sker genom en process som kallas för en studie vilken består av en sekvens av enskilda försök. I varje försök tilldelar Optuna en ny kombination av parametrar som sedan utvärderas mot målfunktionen. Optuna bygger upp sökrymden dynamiskt under körtid och har en tidseffektiv implementering. [5]

2.5 Datamängden GuitarSet

GuitarSet är en datamängd som utvecklats för att erbjuda en standardiserad resurs för utvärdering av algoritmer och system för automatisk musiktranskribering. Datamängden består av 360 inspelningar uppdelade i 180 inspelningar av solospel och 180 inspelningar av ackordspel. Dessa respektive 180 inspelningar är sedan indelade i 5 olika genrer. Utöver inspelningarna har GuitarSet även detaljerade annoteringar som för varje ljudfil beskriver noters starttid, slut och tonhöjd. Annoteringen innefattar även information om vilken sträng och vilket band gitarristen använde för att spela just den tonen. [6]

Tack vare detta kan GuitarSet fungera som ett absolut facit. Detta gör det möjligt att matematiskt utvärdera hur väl en modell som Basic Pitch detekterar rätt toner, hur väl efterbehandling kan förbättra resultatet, samt hur väl en efterföljande algoritm (som den algorimten som används i denna studie) lyckas bestämma den faktiska fingersättningen.

2.6 Viterbi algoritmen

Viterbi algoritmen är en metod för att hitta den mest sannolika sekvensen av händelser i system där vi inte kan se allt som händer. Inom signalbehandling används den för att estimera dolda tillstånd baserat på det vi faktiskt kan observera [7]. I denna studie används algoritmen för att översätta MIDI noter till den mest ergonomiska vägen över greppbrädan.

Algoritmen bygger på konceptet Hidden Markov Models (HMM). En grundläggande princip här är Markov-antagandet (se ekvation 1), vilket innebär att nästa steg i en sekvens endast beror på var man befinner sig just nu [8]:

$$P(q_i|q_1...q_{i-1}) = P(q_i|q_{i-1}) \tag{1}$$

Där q_{i-1} är det valda greppet vid en viss tidpunkt. För en gitarrist betyder detta att valet av nästa position på halsen matematiskt sett bara påverkas av handens nuvarande position.

Inom teorin för HMM kallas de faktiska bakomliggande händelserna för "dolda tillstånd", medan det vi kan mäta kallas för "observationer"[8]. I denna studie utgörs de dolda tillstånden av gitarrens sträng- och bandval medan observationerna består av MIDI noter. Greppen betraktas som dolda eftersom en specifik ton ofta kan spelas på flera olika platser på greppbrädan, information som inte framgår av själva MIDI signalen. För att hitta den mest spelbara vägen genom alla noterna stegar Viterbi algoritmen framåt ett tidssteg i taget (vilket kan vara en enskild not eller ett ackord) och beräknar:

1. Övergångskostnad: Hur jobbigt det är att flytta handen från föregående grepp till nästa.

2. Observationsmatchning: Att det valda greppet faktiskt motsvarar de MIDI noter som observeras i det aktuella tidssteget.

Genom att välja den väg som har högst total sannolikhet vilket i praktiken innebär den lägsta biomekaniska ansträngningen genererar algoritmen en färdig tabulatur. I vår implementation är systemet väldefinierat genom att låta den första noten i sekvensen kunna starta på vilket giltigt band som helst med samma sannolikhet.

3 Problembeskrivning: Gitarrens utmaningar för AMT

Detta kapitel redogör för de specifika utmaningar som gitarren som instrument medför vid automatisk musiktranskribering. Fokus ligger på instrumentets fysiska och akustiska egenskaper, samt varför fenomen som mekaniska transienter och övertoner utgör ett betydande problem för AI-modeller som Basic Pitch.

3.1 Gitarrens fysik

Denna rapport utgår från att gitarrinstrumentet har sex strängar och 22 band. Varje sträng namnges efter den ton som spelas då man spelar en öppen sträng vilket betyder att inget grepp är taget på den strängen. Strängarna är följande: E2 $(82,41Hz)$, A2 $(110,00Hz)$, D3 $(146,83Hz)$, G3 $(196,00Hz)$, B3 $(246,94Hz)$, E4$(329,63Hz)$ [9]. Varje band på gitarrhalsen representerar en ökning av en halvton vilket betyder att noten som spelas på G3 om man greppar det första bandet är G3, det andra bandet är A3 och så vidare. Strängarnas tonomfång överlappar även vilket innebär att exakt samma tonhöjd ofta kan spelas på flera olika platser på greppbrädan. Till exempel kan noten A3 spelas genom att trycka ned det andra bandet på G3-strängen, men exakt samma ton kan också spelas på det sjunde bandet på D3-strängen eller det tolfte bandet på A2-strängen. Gitarrens max frekvens fås vid spelning på den sträng med högsta not på högsta bandet vilket ger not D6 $(1174,66Hz)$ och minsta frekvensen fås via E2-strängen $(82,41Hz)$ [9].

3.2 Akustiska fenomen

Ljudet från en gitarrsträng består av en grundfrekvens och en serie harmoniska övertoner, vilka är svagare medklingande frekvenser som uppstår vid heltalsmultiplar av

grundtonen och ger gitarren dess unika klang. De mest framträdande övertonerna återfinns vid intervall som motsvarar en oktav (där frekvensen dubbleras), en kvint (en frekvensökning med faktorn 1,5) och två oktaver ovanför grundtonen [10].

Gitarren påverkas även av sympatisk resonans vilket innebär att strängar som inte aktivt spelas börjar vibrera genom energiöverföring från andra spelade strängar som delar gemensamma frekvenser. Detta kan skapa en oavsiktlig bakgrundsringning av en ton man inte spelat. [10].

Benetos et al. [11] förklarar att dessa harmoniska serier ofta överlappar vid polyfont spel, det vill säga när flera toner ljuder samtidigt (till exempel vid ackordspel). Detta gör källsepareringen underbestämd vilket för ett AMT-system (som Basic Pitch) innebär att energin från en grundtons starka övertoner riskerar att felaktigt registreras som självständiga toner. Detta kan skapa systematiska fel där övertonerna är som starkast. Vidare beskriver författarna att detektering av ett not-avslut är en av de allra svåraste uppgifterna för ett AMT-system. Den sympatiska resonansen bidrar till denna svårighet då, som Benetos et al. [11] beskriver, modellen får svårt att avgöra när en aktivt spelad ton slutar och när den oavsiktliga bakgrundsringningen tar vid.

3.3 Mekaniska transienter

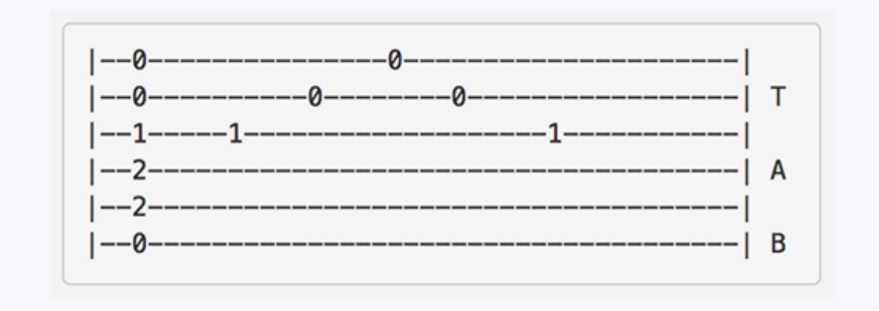
Vid gitarrspel uppkommer även annat ljud än bara de spelade noterna på gitarren. Detta kan t.ex vara anslaget av ett plektrum, fingrar som glider över strängar eller greppbrädan och en hand som slår mot gitarrens kropp. Dessa ljud kallas transienter och karaktäriseras av en plötslig och stark ökning av energi i ljudvågen. [12]

För ett AMT-system utgör dessa transienter en utmaning. Benetos et al [11] beskriver att automatisk transkribering försvåras när ljudsignalen innehåller andra inslag utöver de riktiga noterna. Eftersom mekaniska transienter innebär en plötslig och stark ökning av energi riskerar systemets algoritm att felaktigt tolka dessa ljud som starten på en ny ton. Konsekvensen av detta blir att systemet genererar felaktiga "spöknoter" i utdatan när den hör transienter [11].

3.4 En not - flera möjliga sätt att spela den

I gitarrvärlden används ofta tabulatur för att förmedla musikalisk information på (se figur 1), men som beskrivits i avsnitt 3.1 innebär gitarrens konstruktion att en och samma ton ofta kan spelas på flera olika platser på greppbrädan. Till följd av detta finns det inte en tabulatur för en given MIDI fil, utan en och samma MIDI fil kan översättas till en tab på en mängd olika sätt.

För att översätta en MIDI fil till läsbar och praktiskt spelbar gitarrtabulatur krävs därför en ytterligare algoritmisk bedömning. Ett sätt att lösa detta är att formulera fingersättningsprocessen som ett sökproblem där målet är att minimera en ackumulerad kostnadsfunktion med hjälp av dynamisk programmering [13]. I detta sammanhang innebär dynamisk programmering att algoritmen bryter ner problemet i mindre, sekventiella steg. Enligt denna modell tilldelas en fysisk övergångskostnad för varje tänkbar förflyttning mellan olika fingerpositioner. Kostnaden baseras på biomekaniska begränsningar och tar hänsyn till faktorer som handens rörelse längs banden och över strängarna. Genom att beräkna den väg genom låten som resulterar i den lägsta totala kostnaden kan systemet automatiskt välja den mest ergonomiska fingersättningen och generera en logisk tabulatur [13].



Figur 1: Exempel på gitarrtabulatur. De sex horisontella linjerna representerar gitarrens strängar (där den översta linjen är den ljusaste/tunnaste strängen). Siffrorna anger vilket band på greppbrädan som ska tryckas ner, där 0 innebär att strängen spelas öppen (lös). Tiden läses från vänster till höger.

3.5 Utmaningar med Basic Pitch

Bittner et al. [1] skriver att Basic Pitch är designad för att fungera för alla möjliga sorters instrumenttyper och för att stödja polyfoni (flera toner låter samtidigt). En central utmaningen vid utveckling av en sådan universell modell är avvägningen mellan att identifiera samtliga spelade noter och att ha så lågt antal falska positiva noter som möjligt.

Vidare berättar författarna att en ytterligare egenskap hos modellen är att den inte tillämpar några monofoniska begränsningar på utdatan. Modellens arkitektur har visserligen inbyggda funktioner för brusreducering som ska filtrera bort transienter och oönskade övertoner. Problemet är dock att Basic Pitch saknar monofoniska begränsningar och dessutom är mycket känslig för nya notstarter. För ett instrument som gitarren innebär detta att dess starka övertoner och mekaniska transienter ändå riskerar att felaktigt registreras som riktiga noter. Detta bekräftas av författarna själva, som beskriver sin process för notgenerering som heuristisk. De drar slutsatsen att det förmodligen krävs mer specifika modeller för att kunna höja precisionen ytterligare. [1]

3.6 Notfragmentering

Ett återkommande problem för automatisk musiktranskribering är att systemets utdata innehåller felaktigheter i form av fragmenterade noter. Benetos et al [11] påpekar just detta och identifierar “merged or fragmented notes” som en svårighet för AMT modeller. Fragmentering av en ton innebär att den delas upp i flera kortare MIDI-händelser. Författarna beskriver att för att hantera detta krävs ofta efterbehandlingsmetoder som grupperar rådata till diskreta nothändelser.

3.7 Slutgiltig problemformulering

Basic Pitch är en generell modell som ämnar att fungera för så många instrument som möjligt och har en relativt hög gräns för minsta tillåtna notlängd (ca 128 ms). För gitarrspel utgör detta ett problem då gitarr ofta spelas med snabbare passager (t.ex 16-dels not i 120 bpm varar i 125 ms och en 32-dels not varar i 62,5 ms). Om standardgränsen behålls finns det en stor risk att Basic Pitch filtrerar bort dessa snabbare spelade noter innan de ens hunnit registreras.

För att inte förlora dessa noter måste gränsvärdet sänkas drastiskt (i denna studie till 40 ms). Konsekvensen av detta blir att systemet kommer kunna fånga upp mer

av alla noter som faktiskt spelades men med nackdelen att känsligheten för alla ovannämnda problem ökar. Systemet kommer få in många fler spöknoter. För att lösa denna problematik och därmed göra en generell AMT modell (Basic Pitch) ännu bättre och genuint användbar för gitarr föreslås i denna studie en metod i tre steg:

1. Ett sänkt tröskelvärde för notlängd hos Basic Pitch för att säkerställa att inga faktiskt spelade noter missas.

2. Utveckling av domänspecifik filtrering baserat på gitarrens fysik och harmonik för att i efterhand städa upp alla spöknoter som Basic Pitch genererar.

3. Applicering av dynamisk programmering via Viterbialgoritmen för att slutligen översätta den filtrerade MIDI datan till en ergonomisk och fysiskt spelbar gitarrtabulatur.

4 Forskningsfrågor

Med bakgrund av den problematik som beskrivits i tidigare avsnitt är syftet med detta arbete att utvärdera hur effektivt en domänspecifik filtreringspipeline och en ergonomisk Viterbi optimering kan omvandla rå och överkänslig AMT-data full av spöknoter till en korrekt och spelbar gitarrtabulatur. För att uppnå detta syfte och utvärdera systemets effektivitet ämnar rapporten besvara följande forskningsfrågor:

- **RQ1:** Hur väl presterar den skräddarsydda filtreringspipelinen när det gäller att reducera andelen spöknoter (öka Precision) från den överkänsliga AMT baslinjen (40 ms), utan att kompromissa med täckningsgraden (Recall)?

- **RQ2:** Hur beroende är det avslutande MIDI-till-Tab-steget av korrekt MIDI-data, och i vilken utsträckning kan Viterbialgoritmen identifiera och hantera fysiskt omöjliga ackord som genererats av AMT modellen?

- **RQ3:** Hur väl generaliseras Tab-genereringsprocessen tillsammans med Demucs (separerar gitarren från en ljudmix) på ljudfiler som innehåller mer än bara gitarr? Med andra ord, hur väl generaliserar pipelinen på riktiga låtar?

5 Metod

Nedan följer en beskrivning över hur alla delar i pipelinen togs fram och hur de optimerades. De ingående delarna är följande:

1. Rå ljudfil utsätts för ett förbehandlingssteg som inkluderar ett högpassfilter och källseparering på instrument.

2. Resultatet skickas sedan till Basic Pitch som svarar med korresponderande MIDI fil.

3. MIDI filen utsätts därefter av ett efterbehandlingssteg som innefattar filter som beskrivs i underkapitel *Filter*.

4. Den efterbehandlade MIDI-filen översätts sedan till tabulatur.

Inledningsvis så testades Basic Pitch med det föreslagna lägre tröskelvärdet för minsta notlängd (från och med nu kallat för baseline) på olika låtar för att bilda en uppfattning av hur verktyget fungerade. Det visade sig tidigt att baseline har brister när det gäller gitarrtranskribering till MIDI filer. Ett övergripande tema var att baseline alltid gissar för mycket, likt en hagelbössa. Detta innebär att den ofta hör den grundläggande musiken och lyckas sätta ut melodin/ackorden, men att det även tillkommer väldigt många så kallade "spöknoter". Från detta drogs slutsatsen att filter måste skapas och tillämpas för att förbättra resultatet. Följande filter togs fram:

5.1 Filter

Efter att ljudfilen processats av Basic Pitch (med det sänkta tröskelvärdet på 40 ms), genererades en rå MIDI-fil. Denna fil fångade upp snabba passager väl men innehöll samtidigt en betydande mängd felaktiga spöknoter. För att rena denna data designades och programmerades en skräddarsydd filtreringspipeline.

Pipelinen fungerade genom att listan av genererade noter skickades genom en sekvens av egenutvecklade filter. Varje filter bestod av regler baserade på gitarrens fysiska och akustiska begränsningar. Genom att iterera genom datan utvärderades noternas variabler (tonhöjd (pitch), anslagsstyrka (velocity) och varaktighet (duration)). Om en not bröt mot ett filters regler modifierades eller raderades den. De noter som godkändes skickades sedan vidare som indata till nästa steg i pipelinen, tills en renad och spelbar MIDI fil slutligen erhållits.

5.1.1 Pre-processering, frekvensbaserad eliminering

Innan ljudfilen skickas till Basic Pitch appliceras ett högpassfilter vars syfte är att eliminera lågfrekvens buller och ljud som inte kommer från det riktiga gitarrspelet utan som uppstår på grund av den fysiska kontakten med gitarren. Sådana ljudkällor kan t.ex. vara plektrumet som slår an strängen eller gitarristens hand som slår mot gitarrens kropp. Gitarrens lägsta grundton (den öppna E-strängen) är ca 82 Hz så detta utgjorde ett högsta tak för detta filters värde.

5.1.2 Filtrering av mekaniska transienter och anslagsstyrka

Då ett högpassfilter endast kan eliminera allt ljud som är under sin brytfrekvens kan det finnas transienter som kvarblir med högre frekvenser. Detta filter applicerades iterativt på MIDI datan genom att varje enskild nothändelse utvärderades mot tre sekventiella villkor:

1. **Anslagsstyrka (Velocity):** Tonens anslagsstyrka kontrollerades mot ett tröskelvärde. Toner som understeg detta värde raderades direkt då de bedömdes vara resultatet av bakgrundsbrus.

2. **Transientskydd (Thump-filter):** Mekaniska "dunsar" identifierades genom att kombinera tonhöjd och varaktighet. Om en not har en mycket låg tonhöjd kombinerat med en extremt kort varaktighet klassificerades den som mekaniskt brus och sorterades bort.

3. **Fysiskt tonomfång:** Säkerställ att noten är fysiskt spelbar på gitarr genom att kolla om tonhöjden ligger inom intervallet för toner som faktiskt kan spelas på en gitarr.

5.1.3 Not-sammanslagning (merge)

När en gitarrist spelar en ton och håller den länge kan den gå upp och ner i styrka (velocity). Risken finns då att Basic Pitch tolkar detta som separata anslag och därmed separata toner. Resultatet blir då att en lång ton tolkas som flera korta toner med samma tonhöjd. För att motarbeta detta slår filtret ihop två efterföljande toner med samma tonhöjd som spelas med mycket kort mellanrum (mindre än 50 ms mellan de två tonerna).

5.1.4 Harmonisk och oktavbaserad rensning

När en gitarrsträng slås an så uppstår ofta övertoner på grund av strängens naturliga vibrationsmönster då den vibrerar i flera olika delar samtidigt. Som tidigare nämnt riskerar dessa övertoner att registreras som egna nothändelser även fast de aldrig egentligen spelats. För att identifiera och rensa bort dessa oönskade övertoner analyserades huruvida en not låg exakt 12, 19 eller 24 halvtonssteg över en samtida grundton (vilket motsvarar gitarrens tre starkaste övertoner: en oktav, en oktav plus en kvint, samt två oktaver). Vidare utvärderades noternas anslagsstyrka (velocity). Om den högre notens anslagsstyrka var lägre än grundtonens, eller översteg den med högst en specifik toleransmarginal, klassificerades den som en överton och raderades. Denna marginal finns för att kompensera för eventuella estimeringsfel hos Basic Pitch, då modellen ibland riskerar att överskatta anslagsstyrkan hos medklingande övertoner i en komplex ljudbild.

5.1.5 Hantering av polyfoni

På en gitarr ringer ofta flera toner samtidigt och det kan göra det svårt för modellen att förstå vad som faktiskt spelas just nu och vad som är gamla kvarvarande ringande toner. För att hantera denna problematik utvecklades ett polyfoni filter. Detta filter säkerställer inledningsvis att ingen enskild not får ringa längre än ett förutbestämt gränsvärde, toner som ringer längre än detta får sitt slut flyttat till gränsvärdet. Sedan hanteras kollisioner mellan två samma tonhöjder genom att stänga av en pågående ton i samma ögonblick som en ny ton med samma tonhöjd startar vilket speglar det faktum att en gitarrsträng endast kan vibrera med en grundton i taget. Vidare kontrollerar även filtret om en ton klingat i bakgrunden längre än en viss tidsgräns och om en ny ton påbörjas som har en tillräckligt hög anslagsstyrka i förhållande till den ljudande tonen. Om så är fallet tvingas den äldre tonen att avslutas. Denna relativa jämförelse säkerställer att nya dominanta anslag tillåts maskera ut gamla svagare toner som ringer kvar.

5.1.6 Adaptiv durationsfiltrering baserad på notdensitet - Shred filter

Gitarrister kan inte spela hur fort som helst men ju snabbare de spelar desto svårare blir det för modellen att uppfatta alla noter. Ett statiskt filter med en fast tidsgräns riskerar att radera avsiktliga noter i snabba partier om dessa understiger gränsvärdet. För att försöka lösa detta problem så väl som möjligt appliceras ett så kallat "Shred filter". Filtret analyserar hur många noter som spelas i ett visst tidsfönster kring varje enskild not. Genom att räkna antal noter i tidsfönstret kring noten kan filtret avgöra om detta är ett passage där gitarristen spelar snabbt eller inte. Om antalet noter är över ett visst tröskelvärde anses det vara ett "Shred passage" och då sänks kraven på notens varaktighet så att extremt korta noter kan passera, om det inte anses vara ett Shred passage återgår kraven till en högre standardgräns. När det inte är ett Shred

passage appliceras en statisk tidsgräns och alla toner som har en lägre varaktighet än tröskelvärdet raderas.

Det är viktigt att särskilja filtrets tidsgränser från det tröskelvärde på 40 ms som tidigare satts för Basic Pitch. Värdet på 40 ms är den kortaste möjliga ton som modellen överhuvudtaget tilläts generera i rådatan. Det adaptiva shred-filtret verkar istället i efterhand på denna genererade data. Syftet med filtret var att kunna använda en högre tidsgräns vid långsamt spel för att aggressivt filtrera bort korta spöknoter och brus för att sedan dynamiskt sänka gränsen vid snabba partier så att de korta, faktiskt spelade noterna som Basic Pitch lyckats fånga upp inte raderades av misstag.

5.2 Experimentell uppsättning och optimeringsstrategi

För att utvärdera dessa filter krävs en tydlig referenspunkt (baseline) och en metod för att hitta de mest effektiva inställningarna för samtliga filter. Istället för att förlita sig på manuella gissningar tillämpades en strukturerad experimentell uppsättning. I detta avsnitt presenteras först den baseline som filtren utvärderades mot. Därefter redogörs för hanteringen av optimeringsstrategin med Optuna som hittar bästa möjliga inställningar för alla filter. Slutligen beskrivs de utvärderingsmetoder och mätetal som användes för att kvantifiera systemets prestanda i relation till studiens referensdata.

5.2.1 Baseline

Som referenspunkt för studien används Basic Pitch i sin grundform utan någon före- eller efterföljande bearbetning eller gitarrspecifik filtrering. Syftet med detta är att representera hur en generell modern AI-modell presterar på gitarrspel utan speciella förändringar och justeringar av processen. Parametrarna för Onset Treshhold (0.5), vilket bestämmer känsligheten för att registrera en ny notstart och Frame Treshhold (0.3), vilket anger den lägsta sannolikhetsnivån för att en ton ska anses vara aktiv över tid behölls enligt Spotifys standardvärden [1]. Den enda ändringen som gjordes från standardinställningarna var att sänka minsta tillåtna notlängden från Spotifys standard 127,7 ms till 40 ms. Som tidigare nämnts gjordes detta för att förhindra att modellen automatiskt filtrerar bort snabba passager, vilket säkerställde att snabba anslag ner till en trettioandradelsnot hade möjlighet att registreras i rådatan. Så baseline syftar alltså till Basic Pitch i sitt standardutförande med den enda ändringen som är sänkningen av den minsta tillåtna notlängden.

5.2.2 Val av parametrar och begränsningar - Hårdkodade värden

Då systemet innehåller ett stort antal konfigurerbara parametrar bedömdes en avgränsning av optimeringsrymden vara nödvändig. Om samtliga variabler hade inkluderats i optimeringen hade beräkningskomplexiteten ökat avsevärt, vilket hade försvårat algoritmens förmåga att hitta de mest effektiva inställningarna för att maximera systemets F1 score. På grund av detta så valdes det att vissa värden för filterinställningar skulle hållas konstanta genom hela studien då de representerar fysiska eller musikteoretiska begränsningar. Värdena som hålls konstanta är:

Tonomfång: Värde 40-88 (MIDI notnummer). Detta motsvarar de noter som är spelbara på en standardgitarr.

Tidsföntster för att klassas som övertoner: Värde 0,05 s. Övertoner uppstår i princip samtidigt som grundtonen. Ett fönster på 50 ms bedöms vara tillräckligt för att fånga upp tidsförskjutningar i modellen utan att riskera att radera nästa avsiktliga ton.

Maximal tidslucka för sammanslagning (Merge): Värde 0,05 s. Värdet valdes med stöd i Haas effekten som påstår att om fördröjningen mellan två ljud är mer än 50 ms så uppfattas det som två olika ljudkällor [14].

Maximal notlängd: Värde 1,5 s. Detta valdes för att agera som en nödspärr ifall att modellen misslyckas med att detektera en nots avslut. Bittner et al (2022) beskriver att definitionen av offsets (notavslut) är mindre objektiv än onsets (notstart) på grund av faktorer som efterklang och sustain [1]. Värdet valdes utifrån en bedömning av hur länge en ton på en gitarr brukar spelas baserat på studieförfattarnas erfarenhet.

Maximal tid för bakgrundsringning: Värde 0,4 s. Detta valdes för att balansera hanteringen av oavsiktlig resonans mot avsiktliga korta toner som ska behållas. Eftersom modellen skapar noter genom att spåra sannolikheter framåt i tiden tills värdena faller under ett tröskelvärde finns en risk att svaga vibrationer håller kvar en not i bakgrunden felaktigt länge. Värdet 0,4 s bedöms vara en rimlig heuristisk gräns för att rensa bort sådana artefakter utan att klippa av det avsiktliga spelet.

Det är nödvändigt att skilja på den totala maximala notlängden och varaktigheten för bakgrundsringning då de hanterar olika typer av detekteringsfel. Den maximala notlängden på 1,5 s fungerar som en nödspärr mot kritiska fel i modellens offset detektering, medan gränsen för bakgrundsringning på 0,4 s syftar till att begränsa oavsiktlig ackumulerad sympatisk resonans och efterklang som stör den polyfona tydligheten. Bittner et al. (2022) betonar att just detektering av notavslut är en av de svåraste utmaningarna inom musiktranskribering vilket motiverar användandet av flera heuristiska filter med olika tidsspann för att korrigera modellens utdata.

Gränsvärde för Shred-filter: Värde 9 noter per sekund. Detta fixerades som en fast definition för vad som utgör en tekniskt snabb passage i denna studie. Tröskelvärdet är baserat på att en serie sextondelsnoter spelade i 120 BPM motsvarar 8 noter per sekund vilket av författarna bedöms som "Shred" spel. Genom att sätta gränsen till 9 säkerställs det att den minsta tillåtna notvaraktigheten endast sänks om det faktiskt rör sig om en musikalisk passage som spelas snabbare än konventionell rytmik.

5.2.3 Val av parametrar - Optimerade värden

För resterande parametrar bestämdes deras värde med hjälp av Optuna. Optimeringen utfördes som en flermålsoptimering där både precision (reducering av falska noter) och recall (detektering av samtliga spelade noter) maximeras samtidigt. Sökrymden för varje parameter definierades baserat på tekniska begränsningar och kännedom om instrumentets akustik för att styra algoritmen mot realistiska lösningar. Nedan beskrivs de parametrar som togs med i optimeringsprocessen och deras respektive sökintervall.

Två modellspecifika tröskelvärden för Basic Pitch används: Onset Treshold och Frame Treshhold. Modellens övriga konfigurerbara variabler (såsom minsta tillåtna notlängd och frekvensomfång) hade redan fixerats direkt eller indirekt i studiens baseline och hårdkodade filter, varför endast dessa två sannolikhetsbaserade tröskelvärden lämnades över till Optunas sökrymd.

Starttröskel (Onset Treshold) (0.1-0.9): Bestämmer hur kraftig en energitopp i ljudsignalen behöver vara för att modellen ska registrera att en ny ton startas. Ett högre värde minskar risken för falska positiva toner men med ett för högt värde riskerar man att missa mjuka anslag.

Ramtröskel (Frame Treshold) (0.1-0.9): styr den lägsta sannolikhetsnivån för att en not ska anses vara aktiv över tid, det vill säga hur modellen skiljde uthållig klang från tystnad.

De resterande parametrarna är:

Brytfrekvens (Cutoff Frequency) $(20,0-82,0Hz)$: Brytfrekvensen för högpassfiltret som rensar bort allting under tröskelvärdet Det övre värdet 82 Hz valdes eftersom det motsvarar den lägsta grundfrekvensen på en gitarr i normal stämning ($E282,41Hz$).

Minsta varaktighet (Minimum Duration) (0,01–0,15 s): Definierar den kortaste tillåtna noten (i normalfallet där shred filtret inte är igång). Intervallsgränserna valdes för att ge Optuna möjlighet att optimera balansen mellan att filtrera bort brus och att bevara faktiska noter.

Minsta anslagsstyrka (Minimum Velocity) (0–40): Ett tröskelvärde för anslagsstyrka enligt MIDI standarden (0–127). Detta används för att kasta bort extremt svaga noter som sannolikt är resultatet av bakgrundsbrus eller oavsiktlig strängkontakt.

Duns Filter (Thump Filter) (30–45 MIDI pitch, 0,01–0,15 s): Detta används för att identifiera mekaniska dunsar (plektrumanslag, stötar mot gitarren osv) vid anslag. Optuna tilläts söka efter den kombination av tonhöjd och varaktighet som bäst isolerar dessa dunsar från faktiska toner.

Harmonisk styrkemarginal (Harmonic Velocity Margin) (0–127): Marginalen för när en ton klassas som en överton till en grundton och därmed raderas. Genom att tillåta sökrymden att sträcka sig till 127 kan Optuna i praktiken neutralisera filtret i de fall en tillämpning av filtret visade sig försämra F1 score.

Polyfonisk elimineringsmarginal (Polyphony Kill Margin) (0–127): Bestämmer hur mycket starkare en ny ton måste vara för att tvinga fram ett avslut på en ljudande bakgrundston. Även för denna parameter innebar den breda sökrymden upp till 127 att optimeringsalgoritmen kunde utvärdera om filtret förbättrade systemets prestanda eller om den optimala lösningen var att lämna det inaktivt.

5.2.4 Utvärderingsmetoder och metriker

För att mäta hur bra alla filter presterar jämförs MIDI-filen som producerats med studiens Ground Truth som utgörs av datasetet GuitarSet. Utvärderingen gjordes med hjälp av biblioteket mir_eval med standard inställningar vilket innebär att en detekterad not klassas som korrekt om dess startpunkt ligger inom ± 50 ms från referensnoten och dess tonhöjd ligger inom en kvartston från facit. Dessa toleransnivåer utgör etablerade standarder inom området för att hantera mänsklig tidsperception och naturliga frekvensvariationer. Prestandan utvärderas genom följande metriker: 1. Precision (Träffsäkerhet): Andelen detekterade noter som faktiskt finns i referensfilen. Mäts för att se hur väl systemet städar bort spöknoter. 2. Recall (Täckningsgrad): Andelen av referensfilens noter som systemet lyckades identifiera. Mäts för att säkerställa att inga avsiktliga noter raderas. 3. F1-poäng: Det harmoniska medelvärdet mellan Precision och Recall vilket ger ett samlat mått på total prestanda.

5.3 Genomförande av parameteroptimering för filtren

När utvärderingsmetrikerna var definierade och samtliga filter implementerade så genomfördes optimeringen. Processen exekverades separat över GuitarSets samtliga genrer (BN, Funk, Jazz, Rock, SS), uppdelat på Solo- respektive Comp-spår. Den totala beräkningstiden för optimeringsalgoritmen uppgick till 12 timmar för solospåren och 12 timmar för compspåren.

För att validera de enskilda filtrens bidrag till systemets totala prestanda genomfördes därefter en ablationsstudie. Detta är en analytisk metod där specifika komponenter, i detta fall de enskilda filtren, inaktiveras systematiskt ett i taget för att isolera och mäta deras individuella påverkan på slutresultatet. Utifrån de insikter som detta gav

kring vilka filter som gynnade respektive missgynnade prestandan, utfördes slutligen en andra och slutgiltig optimeringsrunda med Optuna på samma datamängd.

5.4 Tabulaturens generingsprocess

Att översätta MIDI data till gitarrtabulatur är ett svårt problem då en given tonhöjd kan spelas på flera olika strängar på en gitarr. För att lösa detta problem använder pipelinen Viterbi algoritmen kombinerat med heuristiska regler.

5.4.1 Ackordgruppering och tillståndsrymd

Till att börja med grupperas noter som startar inom +-50 ms från varandra till ackord (solo noter blir därför definierade som ackord bestående av endast en ton). För varje enskild not i detta ackord identifieras samtliga möjliga positioner (alla möjliga sträng och band kombinationer) på greppbrädan. Sen genereras en tillståndsrymd av möjliga fingersättningar för hela ackordet genom att ta en kartesisk produkt av noternas positioner. För att begränsa sökrymden appliceras sedan två regler baserat på studieförfattarnas gitarrkunskaper: 1. Varje sträng kan endast spela en not åt gången. 2. Skillnaden mellan det högsta och lägsta bandet i en fingersättning får inte vara mer än 4 band.

5.4.2 Viterbialgoritmen och kostnadsfunktionen

För att försöka hitta den bästa fingersättningen för hela låten används Viterbialgoritmen som minimerar en ackumulerad kostnadsfunktion. Denna kostnadsfunktions syfte är att simulera biomekanisk ansträngning för att spela efterföljande noter eller ackord på gitarr. Viterbialgoritmen hittar sedan den billigaste vägen att fingersätta hela låten och detta blir den resulterande tabulaturen. Kostnaden att gå från en fingersättning till den nästa definierades som:

$$K_{\text{transition}} = (w_{\text{fret}} \cdot \Delta B) + (w_{\text{string}} \cdot \Delta S) - (p_{\text{open}} \cdot N_{\text{open}}) \tag{2}$$

Där variablerna definieras som:

- ΔB: Den horisontella förflyttningen av handens masscentrum längs banden.
- ΔS: Den vertikala förflyttningen av handens masscentrum över strängarna.
- N_{open}: Antalet öppna strängar i det nuvarande greppet, vilket ger en sänkt kostnad (belöning) då dessa inte kräver fingerplacering.
- $w_{\text{fret}}, w_{\text{string}}, p_{\text{open}}$: Vikter som styr prioriteringen mellan de olika rörelserna.

5.4.3 Optimering av ergonomiska vikter

För att bestämma de vikter som innebar det mest optimerade resultatet i förhållande till facit (Guitar Set) användes *Optuna*. Optimeringen genomfördes genom att jämföra algoritmens val av strängar mot valet av strängar i facit. Målet för optimeringen var att maximera strängträffsäkerheten, det vill säga andelen noter där algoritmen valde samma sträng som i facit. Två separata körningar med Optuna genomfördes, där varje körning pågick under sex timmar, för att hitta optimala vikter för solo- respektive ackordspel.

5.5 Hela pipelinen kopplas ihop

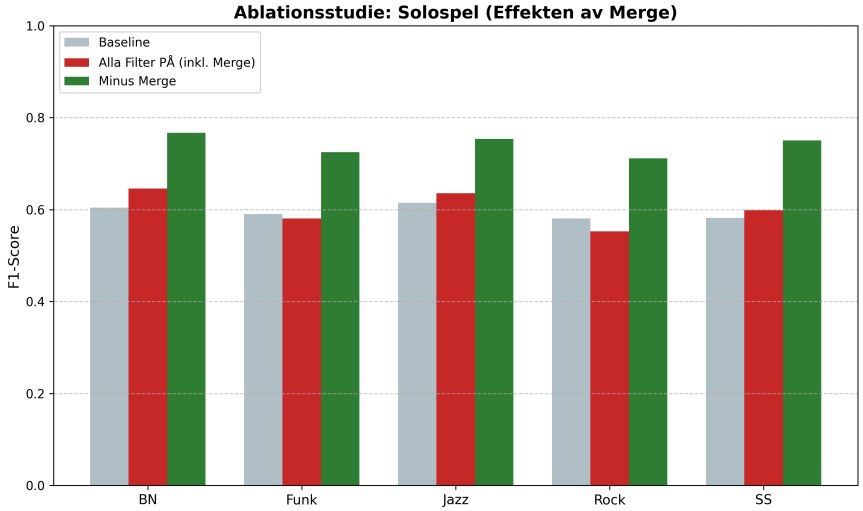
Till slut när alla filter var på plats med värdena från Optuna optimeringen och vikterna för Viterbi algoritmen var på plats kopplades hela pipelinen ihop med Demucs som försteg. I denna studie användes Demucs för att separera gitarr från en ljudfil och resultatet blir en ny ljudfil med bara gitarren i sig. Denna skickas sen först till högpassfiltret, sen till Basic Pitch (med 40ms tröskelvärde), sen appliceras resterande filter och till slut skapas en tab från noterna som är kvar efter all filtrering.

6 Resultat

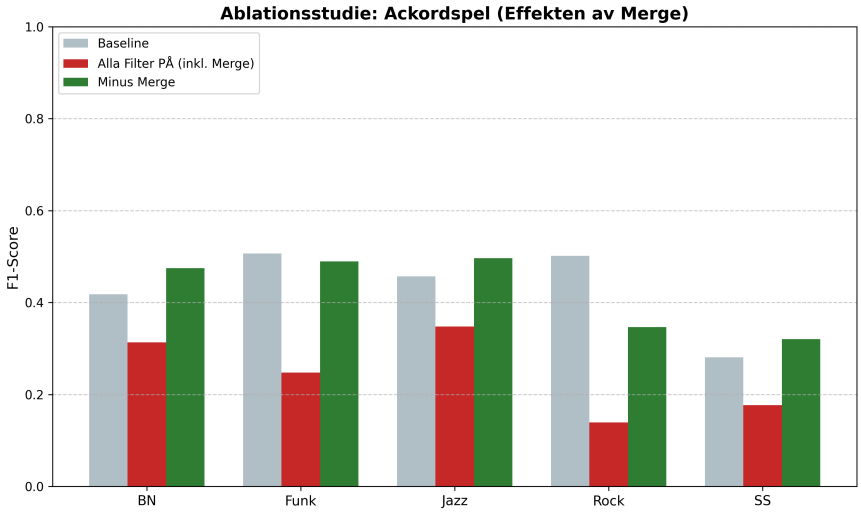
I detta kapitel presenteras resultaten från utvärderingen av den utvecklade filtreringspipelinen.

6.1 Påverkan av merge

Ablationsstudien visade att alla filter hade en positiv eller försumbar påverkan på F1-poängen förutom merge filtret. Merge filtret gjorde alltid mer skada än nytta i samtliga kategorier.



Figur 2: Ablationsstudie för solospel (melodi). Grafen visar hur F1-poängen påverkas när Merge filtret inaktiveras.



Figur 3: Ablationsstudie för ackordspel (comp). Grafen visar hur F1-poängen påverkas när Merge filtret inaktiveras.

6.2 Parametervärden för filter

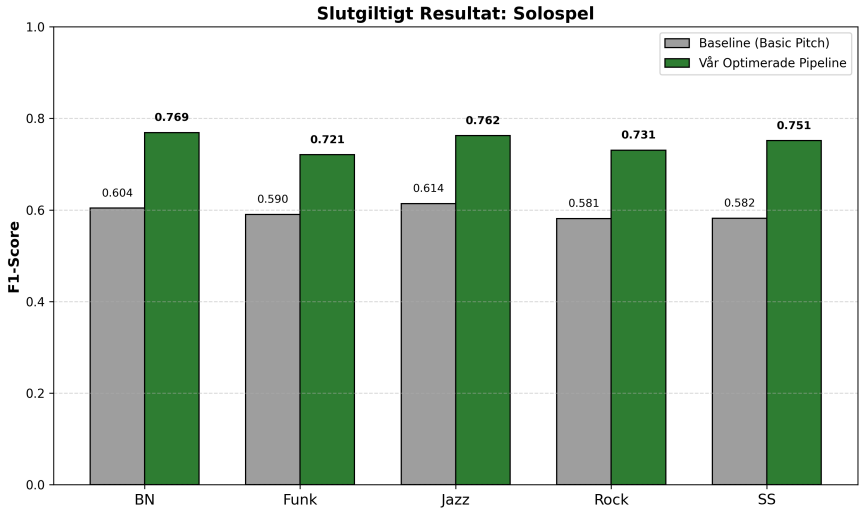
Följande värden är resultatet från optimering med Optuna med merge filtret avstängt.

Läge	Genre	Onset	Frame	Högpass (Hz)	Min Varakt. (s)	Min Anslag	Trans. Pitch	Trans. Tid (s)	Övertonsmarg.	Polyfonimarg.	F1-poäng
Solo	BN	0.82	0.41	45.5	0.093	25	44	0.145	121	3	0.77
Solo	Funk	0.82	0.41	70.7	0.088	38	35	0.143	121	43	0.72
Solo	Jazz	0.79	0.35	49.0	0.088	28	40	0.107	66	11	0.76
Solo	Rock	0.82	0.41	70.7	0.088	38	35	0.143	121	43	0.73
Solo	SS	0.79	0.46	34.2	0.073	35	35	0.026	69	40	0.75
Comp	BN	0.65	0.23	76.5	0.125	38	41	0.096	53	119	0.47
Comp	Funk	0.50	0.25	54.1	0.092	36	37	0.026	19	75	0.53
Comp	Jazz	0.53	0.25	54.1	0.108	36	40	0.109	19	33	0.49
Comp	Rock	0.50	0.25	54.1	0.092	36	37	0.026	19	75	0.49
Comp	SS	0.82	0.35	21.1	0.087	28	39	0.055	120	2	0.37

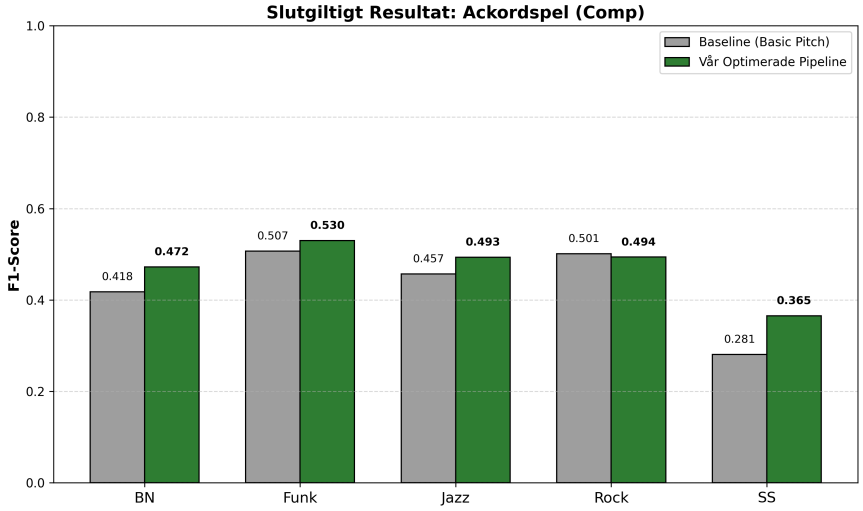
Tabell 1: De optimala parametervärdena genererade av Optuna, samt resulterande F1-poäng för varje genre. Kolumnerna representerar: Basic Pitchs interna tröskelvärden (Onset/Frame), eq-filtrering (Högpass), minsta notvaraktighet (Min Varakt), minsta tillåtna volym (Min Anslag), maxgränser för radering av mekaniska dunsar (Trans. Pitch/Tid), marginal för radering av naturliga övertoner (Övertonsmarg.), samt känsligheten för att stänga av ringande bakgrundstoner (Polyfonimarg.).

6.3 Slutgiltiga F1-poäng

Ovan parametrar gav upphov till följande slutgiltiga F1-poäng.



Figur 4: Slutgiltig transkriberingsprestanda (F1-poäng) för solospel över de fem undersökta genrerna.



Figur 5: Slutgiltig transkriberingsprestanda (F1-poäng) för ackordspel (comp) över de fem undersökta genrerna. Ackordspel uppvisade generellt en lägre träffsäkerhet än solospel.

6.4 Not statistik

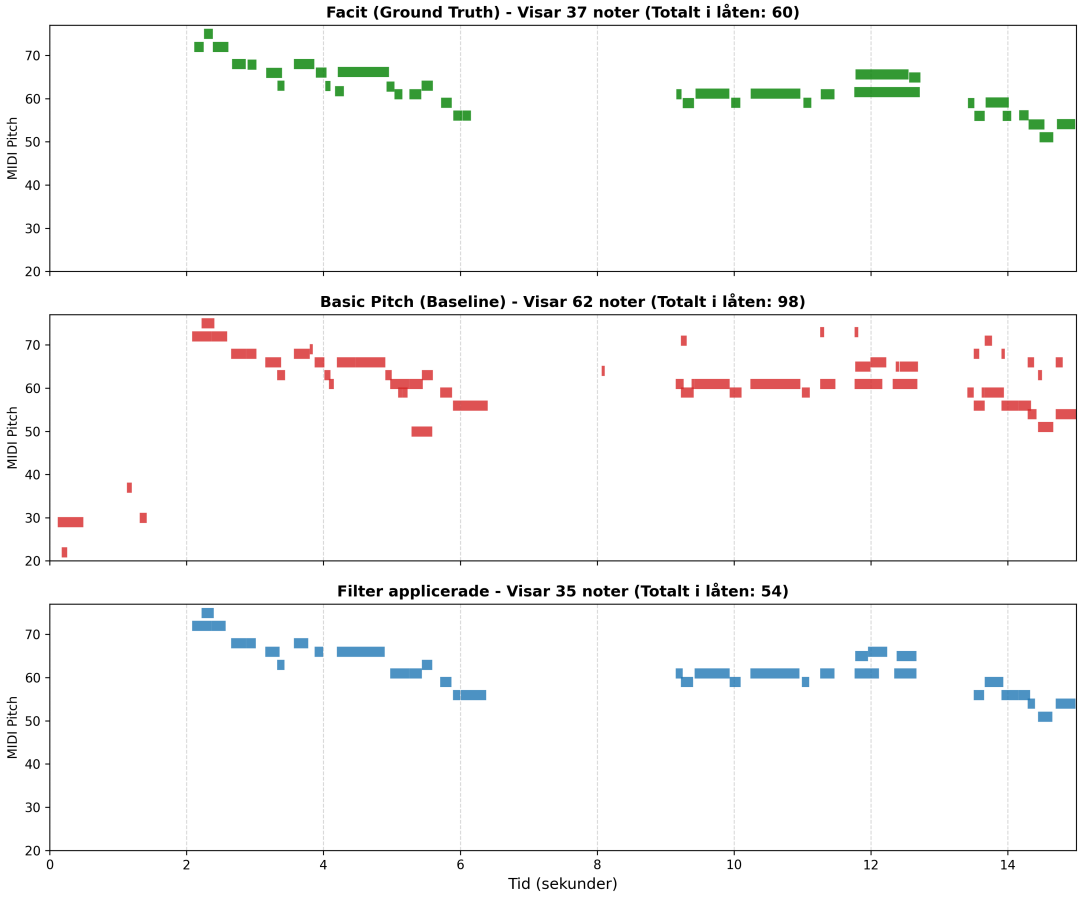
I tabell 2 framgår hur många noter baseline genererade per kategori jämfört med den optimerade MIDI filen med alla filter påslagna (förutom merge).

Läge	Genre	Facit	Baseline	Optimerad	Felmarginal (Base)	Felmarginal (Opt)
Solo	BN	71.8	116.9	72.2	+62.9 %	+0.6 %
Solo	Funk	107.1	163.1	102.1	+52.2 %	-4.7 %
Solo	Jazz	78.1	121.9	79.1	+56.2 %	+1.4 %
Solo	Rock	99.8	160.0	98.8	+60.3 %	-1.0 %
Solo	SS	111.6	183.5	114.4	+64.4 %	+2.5 %
Comp	BN	185.1	271.4	163.1	+46.6 %	-11.9 %
Comp	Funk	282.8	372.4	299.8	+31.7 %	+6.0 %
Comp	Jazz	180.8	248.4	180.6	+37.3 %	-0.1 %
Comp	Rock	342.2	433.2	292.3	+26.6 %	-14.6 %
Comp	SS	276.2	536.1	209.9	+94.1 %	-24.0 %

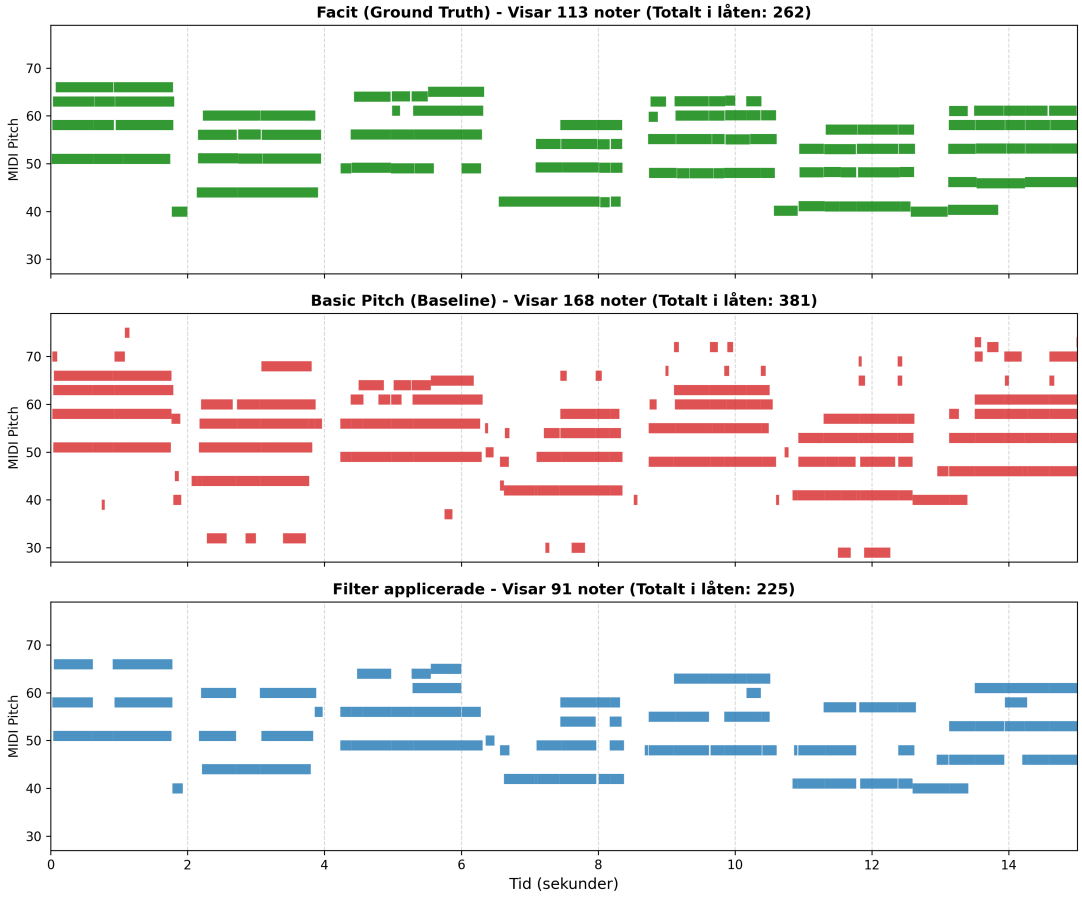
Tabell 2: Sammanställning av antalet genererade noter (genomsnitt per låt) jämfört med annoterad referensdata. Felmarginal indikerar procentuell över/under detektion (spöknoter).

6.5 Visuell utvärdering (Pianorullar)

Figur 6 och 7 är en visualisering av hur filterna höjer F1-poängen genom att filtrera bort spöknoter som Basic Pitch har genererat fel.



Figur 6: Visuell representation av transkriberingen för solospel under ett 15 sekunder långt tidsfönster. Den understa grafen visar hur den optimerade pipelinen framgångsrikt har filtrerat bort en betydande del av det brus (spöknoter) som syns i den oprocessade baslinjen (mitten).



Figur 7: Visuell representation av transkriberingen för ackordspel Jazz.

6.6 Vilka filter gjorde vad i slutändan?

För att utvärdera vilka komponenter i filtreringen som var mest gynnsamma i det slutgiltiga resultatet genomfördes ännu en ablationsstudie där ett filter stängdes av i taget och påverkan på F1-poängen undersöktes. De olika stegen i studien definieras enligt följande:

Minus Högpassfilter: Innebär att ljudfilen skickas direkt till modellen utan att lågfrekvent buller har filtrerats bort i förväg.

Minus Anslagsstyrka: Innebär att tröskelvärdet för MIDI velocity sätts till noll. Systemet godkänner därmed alla noter oavsett hur svagt de spelades vilket inkluderar extremt tyst bakgrundsbrus.

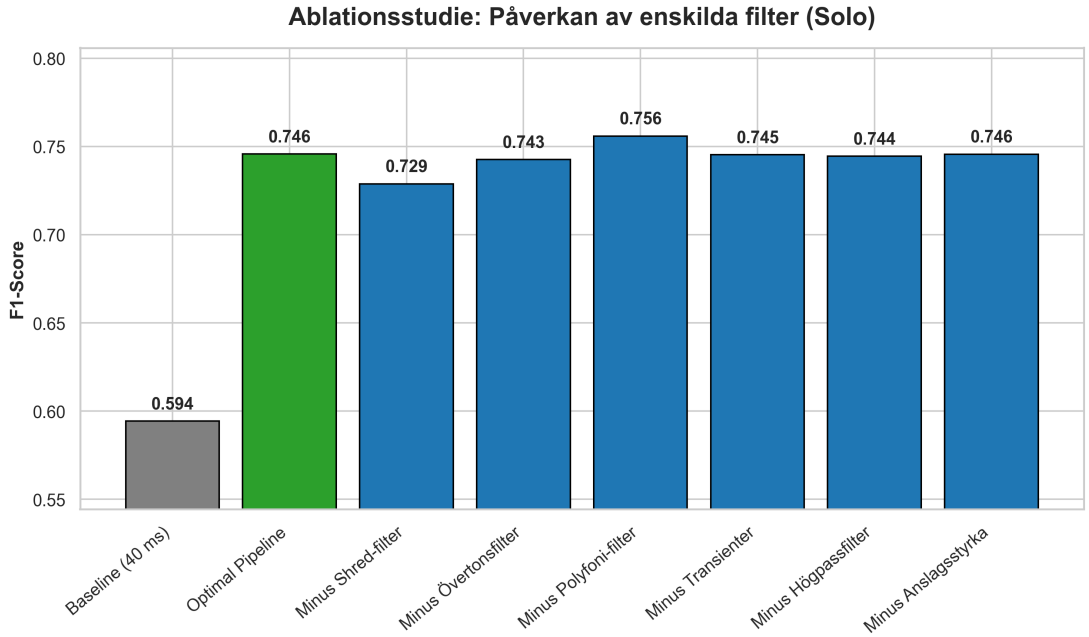
Minus Transienter: Innebär att det filter som letar efter och raderar extremt djupa och korta thumps stängs av.

Minus Övertonsfilter: Innebär att algoritmen inte längre analyserar harmoniska samband. Modellen slutar alltså att radera spöknoter som uppstår på grund av strängarnas naturliga övertoner (oktaver och kvinter).

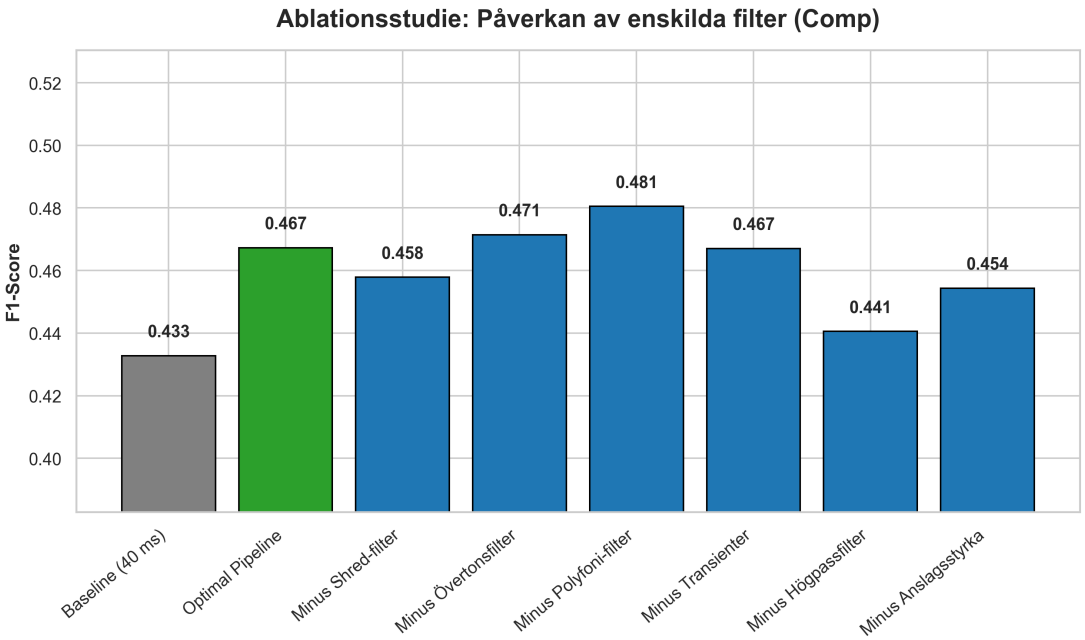
Minus Polyfoni-filter: Innebär att systemets regler för att hantera polyfonisk överlappning inaktiveras. Gamla, ringande bakgrundstoner tillåts därmed klinga kvar utan att stängas av när nya dominanta toner spelas.

Minus Shred-filter (Durationsfilter): Innebär att spärren för minsta tillåtna notlängd stängs av helt (sätts till 1 ms). Systemet släpper därmed igenom alla noter oavsett hur

korta de är vilket stänger av både den statiska minimigränsen och det adaptiva shred undantaget.



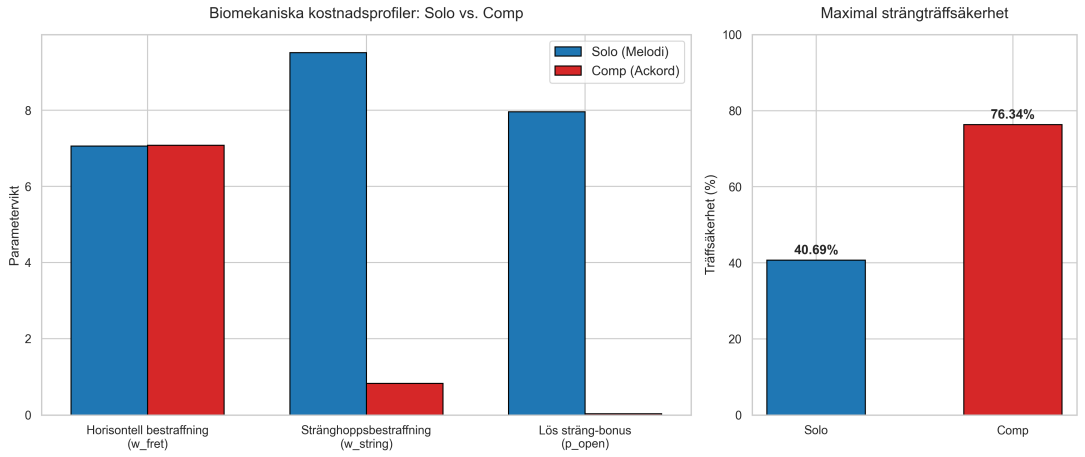
Figur 8: Ablationsstudie för Solo-läget. Grafen visar hur den totala F1-poängen påverkas när enskilda filterkomponenter inaktiveras från den fullständiga pipelinen, det vill säga systemet där samtliga filter är aktiverade med de parametervärden som Optuna tagit fram för att maximera prestandan



Figur 9: Ablationsstudie för Comp-läget (ackordspel). Grafen visar hur den totala F1-poängen påverkas när enskilda filterkomponenter inaktiveras från den fullständiga pipelinen, det vill säga systemet där samtliga filter är aktiverade med de parametervärden som Optuna tagit fram för att maximera prestandan

6.7 Greppbrädesoptimering och spelbarhet

Nedan syns resultatet från Optuna-körningen för Viterbi algoritmen och den slutgiltiga strängträffsäkerheten.



Figur 10: De optimala biomekaniska kostnadsprofilerna för Viterbi-algoritmen. Vänstra grafen visar hur algoritmen prioriterar annorlunda mellan horisontella förflyttningar (band) och vertikala förflyttningar (strängar) beroende på spelstil. Högra grafen visar algoritmens resulterande strängträffsäkerhet mot facit.

Här är en sammanfattning av de bästa värdena från optimeringen:

Spelläge	Träffsäkerhet	w_{fret}	w_{string}	p_{open}
Solo	40,69 %	7,06	9,51	7,96
Komposition	76,34 %	7,08	0,83	0,03

Tabell 3: Optimerade vikter för de två spellägena. Värdena representerar de inställningar som bäst efterliknade de mänskliga strängvalen i GuitarSet.

6.8 Mätning av beroendeförhållande (RQ3)

För att besvara forskningsfråga 3, gällande beroendeförhållandet mellan transkribering och tabulaturgenerering, mättes hur ofta Viterbi algoritmen misslyckades med att finna en möjlig fingersättning. En "krasch" definieras här som en sekvens där transkriberingen genererat ett ackord med ett handspann som är större än fyra eller om ett ackord har bestått av fler noter än som är möjligt att spela samtidigt på en gitarr (om t.ex MIDI-filen har haft 7 toner samtidigt)

Läge	Genre	Baseline krasch	Optimerad krasch	Förbättring
Solo	BN	1,4 %	0,0 %	1,4 %-enheter
Solo	Funk	1,2 %	0,0 %	1,2 %-enheter
Solo	Jazz	1,1 %	0,0 %	1,1 %-enheter
Solo	Rock	1,8 %	0,0 %	1,8 %-enheter
Solo	SS	1,5 %	0,0 %	1,5 %-enheter
Comp	BN	9,3 %	0,7 %	8,6 %-enheter
Comp	Funk	7,1 %	0,9 %	6,2 %-enheter
Comp	Jazz	11,4 %	1,3 %	10,1 %-enheter
Comp	Rock	14,7 %	1,6 %	13,1 %-enheter
Comp	SS	12,2 %	0,1 %	12,1 %-enheter
Totalt	**Solo**	**1,41 %**	**0,01 %**	**1,40 %-enheter**
Totalt	**Comp**	**11,13 %**	**0,88 %**	**10,25 %-enheter**

Tabell 4: Fullständig sammanställning av kraschtest för Viterbi algoritmen per genre och läge. Siffrorna visar andelen genererade ackordsekvenser där algoritmen inte kunde finna en möjlig fingersättning.

6.9 Studiens Baseline vs Spotifys Baseline

Spelläge	Modell	Precision	Recall	F1-poäng	Snitt Noter	Facit
Solo	Spotify Standard (128ms)	0.647	0.705	0.671	103	94
Solo	Studie Baslinje (40ms)	0.493	0.760	0.594	149	94
Comp	Spotify Standard (128ms)	0.447	0.464	0.449	272	253
Comp	Studie Baslinje (40ms)	0.374	0.528	0.433	372	253

Tabell 5: Jämförelse mellan Spotifys standardinställning för Basic Pitch (128 ms) och den anpassade baslinjen (40 ms) som användes i denna studie.

7 Diskussion av resultat

I detta kapitel analyseras och tolkas de resultat som presenterades i föregående kapitel.

7.1 Ablationsstudien – Bra i teorin men sämre i praktiken

För att förstå vilka filter som faktiskt bidrog till förbättringen genomfördes som nämnt en ablationsstudie. Det mest anmärkningsvärda resultatet från detta var effekten av merge filtret som konsekvent försämrade F1-poängen i samtliga genrer för både Solo och Comp spel (se figur 2 och 3).

Som diskuterats i teoriavsnittet är notfragmentering en känd svårighet för AMT-system. Merge filtret designades med den teoretiska utgångspunkten att åtgärda denna fragmentering genom att tvinga ihop intilliggande noter av samma tonhöjd. Att filtret ändå försämrade helhetsresultatet i praktiken kan bero på en konflikt mellan hur man spelar gitarren och utvärderingsramverket mir_eval.

Inom gitarrspel är det vanligt med snabba upprepade anslag på samma sträng. När algoritmen sveper över datan och slår ihop noter som ligger nära varandra i tid riskerar den att radera korrekta notstarter från dessa snabba passager och istället baka ihop dem till en enda lång MIDI-not. Eftersom utvärderingsramverket mir_eval lägger en kritisk vikt vid just starttider för att godkänna en not (inom ± 50 ms) innebär en sammanslagen not att en av de faktiskt spelade noterna bedöms som missad (en falsk negativ). Detta leder direkt till en sänkt recall. Den logiska slutsatsen är därmed att försöket att rädda ett fåtal fragmenterade långa toner skapade mer skada i snabba rytmiska passager än vad det gjorde nytta för helheten vilket är en trolig förklaring till den sjunkande F1-poängenn.

7.2 Den anpassade baselinen - Hagelbössan

Vid analys av resultatet kan en tydlig trend utläsas. Den anpassade baselinen (Basic Pitch med gränsvärdet sänkt till 40 ms) agerar som en hagelbössa och gissar konsekvent alldeles för många noter för en given ljudfil. Detta syns tydligt i tabell 2 där baseline konsekvent har en felmarginal mellan 52,2% och 64,6% för Solo låtar och en felmarginal mellan 26,6% hela vägen upp till 94,1% för Comp låtar. Med felmarginal avses i detta sammanhang systemets procentuella över- eller underdetektering av noter. Baselinen kan alltså liknas vid en hagelbössa som skjuter massa skott på en gång och även om det leder till att den träffar rätt på många av noterna så introduceras en väldigt stor mängd spöknoter till följd av detta. Om man sedan ser till tabell 2 igen men tittar på felmarginalen efter alla filter slagits på sjunker den markant. För Solo låtar ligger felmarginalen under 5% för samtliga låtar och så lågt som +0,6% för BN genren. Liknande kan det ses att felmarginalen för Comp låtar sjunker drastiskt när alla filter sätts på. Felmarginalen ligger under 24% för alla genrer men är så låg som 0,1% för Jazz låtar.

Denna trend är även tydlig i figur 6 och 7. Baselines gissning är full av extra skräp som städas upp när filterna på. T.ex i figur 6 ses ett antal noter i början av transkriberingen som inte finns med i facit. Detta är troligen någon form av ljud som uppstått på grund av gitarristens interaktion med gitarren. Det kan möjligtvis vara ljudet av gitarren som läggs i knät eller andra former av lågfrekvent brus. Dessa toner identifieras korrekt som spöknoter av filterna och slängs bort i den slutgiltiga MIDI-filen. Vidare ser man i figur 6 och 7 att baseline konsekvent lägger till extra gissade noter som inte finns med i facit (kan t.ex vara övertoner eller sympatisk resonans) som filterna korrekt identifierar som skräp och raderar.

I termer av utvärderingsmått innebär detta att vår anpassade baslinje uppnår en hög recall (få missade noter) men till priset av en mycket låg precision (hög andel spöknoter). Detta belyser grundproblematiken med att applicera en generell modell som Basic Pitch på snabbt gitarrspel. För att modellen inte ska filtrera bort de korta snabba noterna

måste den tvingas att gissa brett, vilket oundvikligen skapar brus. Resultaten indikerar att vår optimerade pipeline kan filtrera bort denna hagelskur av spöknoter vilket är en förutsättning för att kunna utnyttja den höga täckningsgraden.

7.3 Skillnaden mellan Solo och Comp

När man jämför resultaten för solospel och ackordspel syns det tydligt att systemet har lättare för den ena kategorin än den andra. I figur 4 kan det utläsas att F1-poäng för sololåtar landar på en hög och relativt stabil nivå mellan 0,72 och 0,77 efter optimering medan i figur 5 syns motsvarade F1-poängen på värden mellan 0,37 och 0,53. Trots att filterna lyckas identifiera och filtrera bort spöknoter för båda kategorierna så har systemet mycket svårare för ackord musik och förklaringen till detta är förväntad och grundar sig i de akustiska fenomen som polyfoni (när flera toner ljuder samtidigt) medför.

Som tidigare nämnt i kapitel 3 så överlappar harmoniska serier ofta vid polyfont spel. För Basic Pitch som saknar monofoniska begränsningar i sin utdata blir detta problematiskt. När flera toner i ett gitarrackord klingar samtidigt så innebär det att energin från grundtonernas starka övertoner löper en ännu större risk att felaktigt registreras som självständiga noter. Modellen gör därför systematiska fel eftersom den inte kan avgöra huruvida energin tillhör en ny not i ackordet eller om det enbart är en överton från en annan sträng.

En annan trolig förklaring till att comp har lägre F1 poäng kan vara att alla problem som modellen har när bara en sträng spelas (som övertoner, transienter, sympatisk resonans osv) blir multiplicerade med sex (förutsatt att alla sex strängar spelas) när gitarristen istället börjar spela ackord.

7.4 Genrespecifika utmaningar - Spelteknik kontra ackordstruktur

När man granskar F1-poängen över de olika genrerna framträder en intressant paradox mellan solo och comp låtarna. För sololåtar har BN (Bossa Nova) högst F1-poäng medan Funk och Rock har lägst men för complåtar är det tvärtom där Funk och Rock presterar högst medan BN är näst sämst bland genrerna. Detta gör att det är svårt att dra slutsatser om en genre skulle vara lättare än den andra i generella drag. Dock kan det finnas möjliga förklaringarna till att det blev såhär i denna studie. Utifrån författarnas musikteoretiska och praktiska domänkunskap är Bossa Nova sololåtar ofta spelade med fingrarna på en nylonsträngad gitarr och mjuka långa anslag utan snabba byten mellan toner, alltså en väldigt fördelaktig miljö för en modell som Basic Pitch att fånga upp alla toner. Å andra sidan spelas Rock och Funk solospår ofta med plektrum på en stålsträngad gitarr med snabba rytmiska inslag och snabba byten mellan toner varför det blir svårt för Basic Pitch att fånga alla toner vid solospel. Sen vad gäller comp spel så spelar ofta Rock och Funk med enklare ackord (mycket triader och power chords där inte alla strängar spelas samtidigt), alltså en mer fördelaktig miljö för Basic Pitch, medan Bossa Nova ofta innehåller mer komplexa ackord med fler toner vilket gör det svårare för Basic Pitch att fånga alla noterna.

7.5 Vilka filter gjorde vad och Optunas val av parametrar

7.5.1 Basic Pitch - Onset och Frame Treshold

Vid granskning av de optimerade värdena för filterna som Optuna genererade kan vissa trender utläsas i hur systemet skiljer på solo och ackordspel. För Basic Pitchs inbyggda

tröskelvärden (Onset och Frame) ser vi att solospel generellt kräver högre värden för båda (Onset runt 0.80 och Frame runt 0.40) jämfört med ackordspel (Onset ofta runt 0.55 och Frame ofta runt 0.24). Detta indikerar att systemet har råd att vara striktare och kräva en starkare och tydligare signal vid solospel medan vid ackordspel måste den sänka kraven för att godkänna en not. Precis som tidigare diskuterats så är ackordspel mer komplext (mer övertoner, mer akustiska problem) så modellen måste sänka tröskeln för att inte missa några faktiska noter. Men ju mer man sänker tröskeln, desto mer osäker blir även modellen på varje gissad not och det skulle kunna förklara varför ackordspel generellt har lägre F1-poäng.

7.5.2 Shred och notlängds filtret

En annan trend kan ses i gränsvärdet för den kortaste tillåtna noten. I solospel ligger snittet lägre (mellan 0,073 och 0,093 sekunder) jämfört med ackordspel (mellan 0,087 och 0,125 sekunder). Detta kan ses som en matematist reflektion av verkligt gitarrspel där solospel ofta har fler kortare noter spelade snabbt efter varandra medan ackordspel har längre toner som ringer ut.

Vikten av att bevara dessa snabbare passager bekräftas av ablationsstudien för både solo och comp (se figur 8 och 9). När durationsfiltret inaktiverades (Minus Shred Filter) ledde det till ett signifikant fall i F1-poäng för båda spelstilarna. Detta beror på att systemet i detta fall släpper igenom alla noter oavsett hur korta de är. Konsekvensen blir att även om modellen fångar upp alla snabba passager, så översvämmas utdatan av en enorm mängd extremt korta spöknoter och brus. Detta sänker systemets Precision drastiskt, vilket i sin tur drar ner F1-poängen.

7.5.3 Högpassfilter (EQ)

Även om brytfrekvensen för högpassfiltret inte uppvisar någon uppenbar trend i Tabell 1 så visar ablationsstudien för ackordspel (Figur 9) att filtret är den viktigaste komponenten för polyfonisk transkribering. När filtret inaktiveras (Minus Högpassfilter) sjunker F1-poängen för Comp markant.

Detta tyder på att förekomsten av frekvensfiltrering är viktigare än dess exakta brytfrekvens. Så länge det lågfrekventa bruset och resonansen elimineras innan Basic Pitch analyserar ljudet ökar chansen för en korrekt tolkning av inspelningen.

7.5.4 Övertonsfilter och harmonisk rensning

Optunas val av den harmoniska styrkemarginalen uppvisar en paradox där för solospel väljs ofta värden nära 127 (vilket i praktiken innebär att filtret inaktiveras) men för ackordspel väljs däremot ofta mycket lägre värden vilket innebär en aggressiv filtrering. Det lägre värdet tyder på att optimeringsalgoritmen försöker använda en aggressiv filtrering för att hantera den komplexa ljudbilden som uppstår när flera strängar ringer samtidigt.

Ablationsstudien visar dock på ett mer nyanserat resultat gällande filtrets faktiska nytta. För solospel sjunker F1-poängen endast marginellt när filtret inaktiveras (Minus Övertonsfilter), vilket bekräftar att övertoner utgör ett mindre problem vid monofont spel. För ackordspel (Comp) observeras däremot att F1-poängen faktiskt ökar något när övertonsfiltret tas bort. Detta beror troligen på att filtret visserligen raderar oönskade spöknoter orsakade av övertoner, men att det samtidigt felaktigt raderar faktiska noter i ackorden som råkar ligga på de intervall (oktaver eller kvinten) som filtret letar efter.

7.5.5 Mekaniska transienter (Thump-filter) och Anslagsstyrka

För dunsfiltret (transienter) väljer Optuna generellt en längre varaktighet vid solospel än vid ackordspel. Detta kan förklaras av att enskilda anslag i en melodi (som vid solospel) ofta är mer aggressiva och skapar längre mekaniska vibrationer i gitarrkroppen än de svepande rörelserna vid ackordspel. Men som det visar sig i Figur 8 och 9 verkar inte transientfiltret ha en märkbar påverkan på F1-poängen. En anledning till detta kan vara att många av transienterna försvinner från tidigare filtrering som t.ex högpassfiltret som appliceras innan detta filter ens får en chans att göra nåt.

Vikten av tröskelvärdet för anslagsstyrka är däremot mer märkbar i alla fall vid ackordspel. Här sjunker F1-poängen från 0,467 till 0,454 när filtret inaktiveras. Vid solospel har detta filter ingen påverkan på slutgiltiga F1-poängen.

7.5.6 Polyfoni filtret

Ett anmärkningsvärt resultat i ablationsstudien var att inaktiveringen av polyfonifiltret (Minus Polyfoni-filter) ledde till en ökning av F1-poäng för både solo och comp. Detta tyder på att filtrets regler för att stänga av ringande toner är för strikta och därmed raderar legitima noter, vilket skadar modellens Recall. Optuna har valt spridda värden här och troligen har optimeringsalgoritmen försökt minimera filtrets negativa inverkan.

7.6 Generering av tabulatur: Viterbialgoritmen och biomekanik

Ett initialt förvånande resultat från Optuna-optimeringen (Tabell 3) var den stora skillnaden i träffsäkerhet. För ackordspel träffade algoritmen rätt i 76,34% av fallen medan motsvarande siffra för solospel endast var 40,69%. Denna skillnad grundar sig troligen i greppbrädans uppbyggnad och begränsningar i denna studies algoritm. Som tidigare nämnt kan en given ton tas på flera olika ställen på gitarrens greppbräda men när ett ackord spelas så begränsas antalet möjliga fingersättningar av algoritmens lagar (två toner kan inte spelas på en sträng och handspannet får inte vara större än fyra band). Detta tvingar fram logiska ackordformer vilket förenklar för algoritmen att välja samma fingersättning som original gitarristen gjorde. Vid solospel å andra sidan finns inte samma begränsningar då solospel bara spelar en sträng i taget så ett handspann kan t.ex aldrig vara större än 4 då bara en ton spelas i taget. En träffsäkerhet på 41% betyder dock inte att algoritmen är dålig, bara att den inte väljer samma fingersättning som gitarristen i inspelningen gjorde. Det kan även diskuteras huruvida strikt strängmatchning är en lämplig utvärderingsmetrik eftersom två helt olika fingersättningar kan vara precis lika korrekta och spelbara i praktiken.

Det är även intressant att studera de optimala vikterna som Optuna bestämde då dessa belyser skillnaden i biomekanik mellan de två spellägena. För solospel prioriterades en hög bestraffning för vertikala stränghopp (w_string = 9.51) och en hög belöning för öppna strängar (p_open = 7,93) medan ackordspel har nästan ingen bestraffning för sträng hopp (w_string = 0,83) och väldigt låg bonus för öppna strängar (p_open = 0,03).

Detta kan tolkas som en matematisk reflektion av hur gitarr spelas i verkligheten. Den höga bestraffningen för stränghopp för solo indikerar att gitarrister vill minimera antalet gånger både deras höger och vänsterhand (vänster på greppbrädan och höger med plektrum/finger) måste byta sträng och istället rör sig upp och ner längs med greppbrädan. Den stora belöningen för öppna strängar stämmer också in på detta mönster då öppna strängar är lättare att spela och ger gitarristen mer tid att flytta fingrar till ett nytt läge.

När ett gitarrackord spelas innebär det att ett flertal strängar ljuder samtidigt. Att stränghoppskostnaden sjunker till nästan noll tyder på att algoritmen har identifierat att det (i princip) inte kräver något extra fysisk ansträngning att slå över många strängar med ett svepande plektrumdrag jämfört med att spela en ensam sträng. På liknande vis ser vi den låga öppna strängbelöningen för ackordspel som kan förklaras av att ackord tas i låsta handpositioner (t.ex barreackord är en vanlig sådan handposition) som inte innehåller öppna strängar.

Intressant att notera är att den horisontella bestraffningen (bestraffningen att flytta upp och ner längs med banden) är liknande för båda spelstilarna. Detta kan förklaras av att en horisontell förflyttning kräver att hela armen och handen skiftar position på gitarrhalsen. Denna rörelse utgör en konstant fysisk tröskel som tar ungefär lika mycket tid och ansträngning oavsett om gitarristen därefter ska spela en enskild not eller placera fingrarna för ett helt ackord. Att Optuna tilldelar detta en likvärdig kostnad för båda spelstilarna visar att den har fångat en grundläggande fysisk begränsning hos gitarren som gäller oavsett spelstil.

7.7 Fingersättning och beroende av korrekt indata

I tabell 4 kan en tydlig skillnad ses i kraschfrekvensen mellan baseline och den optimerade pipelinen. En “krasch” definieras här som att transkriberingen genererat MIDI data som är fysiskt omöjlig att spela, antingen genom att innehålla fler än sex samtidiga noter eller ett handspann över greppbrädan som överstiger fyra band. Baseline kraschar i 11,13% för ackordfall men i 0,88% av fallen efter att filterna slås på, en förbättring på 10,25 procentenheter. För solo kan en liknande trend ses där kraschfrekvensen minskar med 1,40%. Detta visar att baseline har så mycket spöknoter och felaktigheter i sig att den matematiska modellen för fingersättning som använts i denna studie inte fungerar. Genom att applicera de optimerade filtren kan dock MIDI-datan städas så pass effektivt att kraschfrekvensen nästan helt elimineras vilket gör att MIDI-till-Tab-steget fungerar igen. Från detta blir det tydligt att Audio-till-MIDI steget och MIDI-till-Tab steget är beroende av varandra då tabulaturprocessen kräver ren indata utan massa spöknoter och liknande för att kunna ge en tillfredställande resulterande tabulatur.

7.8 Självförvållat brus eller nödvändig överdetektering?

För att kunna utvärdera om denna studies skräddarsydda pipeline faktiskt utgör en förbättring av Basic Pitch för gitarrspel måste inledningsvis konsekvenserna av att ändra Spotifys rekommenderade inställning för minsta tillåtna notländ på ca 128 ms till 40 ms analyseras. Som tidigare nämnt riskerar denna gräns att filtrera bort snabbare toner vilket är vanligt för gitarrspel.

Konsekvensen av denna sänkning illustreras tydligt i Tabell 5. Spotifys standardinställning gav ett F1-poäng på 0,671 för solospel, medan den råa 40 ms-baslinje sjönk till 0,594. Genom att sänka tröskeln skapades oundvikligen det som ovan liknats vid en "hagelbössa". Modellen började överdetektera och registrerade minsta lilla mekaniska skrap, överton och sympatiska resonans som legitima noter. Majoriteten av de spöknoter som filterna sedan fick kämpa emot var alltså artefakter som introducerades när modellen öppnades för snabbare spel.

Frågan blir då: var det värt att försämra modellens grundprestanda för att kunna fånga snabba noter? Resultaten indikerar ett tydligt ja. Spotifys originalmodell missade en betydande mängd korrekt spelade noter på grund av sin höga tröskel och dessa fångades upp av Baseline (se Tabell 5, Baseline har konsekvent högre Recall värde). Genom

att sänka gränsen till 40 ms och sedan applicera Optuna-optimerade domänspecifika filter kunde en slutgiltig förbättring i F1-poäng observeras när solo steg från 0.671 (för standard Basic Pitch) till ca 0.75 (för den optimerade pipelinen) och för comp steg F1-poängen från 0.449 (för standard Basic Pitch) till 0.47 (för den optimerade pipelinen).

En tolkning av detta är att när det gäller automatisk transkribering av gitarr är det en mer framgångsrik strategi att medvetet framkalla överdetektering (falska positiver) som sedan kan filtreras bort med fysik- och DSP-filter än att acceptera att modellen helt missar viktiga noter från början.

7.9 Hur väl generaliserde det? - En subjektiv bedömning

För att undersöka detta testades ett urval av låtar genom hela pipelinen. Dessa låtar var Back In Black av AC/DC, Long Train Running av Doobie Brothers, Jag Kommer av Veronica Maggio och en akustisk fingerstyle variant av låten My Heart Will Go On av Celine Dion. Då det inte finns någon exakt Ground Truth vi kan jämföra mot här skedde denna bedömning subjektivt med studieförfattarnas kunskaper inom musik och gitarr som grund. Jämförelsen gjordes genom att ställa utdatan från Spotifys standardversion av Basic Pitch mot resultatet från den optimerade pipelinen i denna studie. Analysen utfördes genom visuell inspektion och lyssning av MIDI data i Logic (ett program som kan spela upp MIDI data) samt en granskning av den genererade tabulaturens ergonomi.

7.9.1 Hur blev resulterande MIDI-datan?

För fingerstyle varianten av My Heart Will Go On presterade pipelinen bättre än standard Basic Pitch. Resultat från pipelinen hade färre spöknoter och noter som inte hörde hemma i låten och melodin lät mer lik originalet. För Back in Black och Long Train Running är bilden mer nyanserad där pipelinen är bättre på vissa sätt och Spotify standard Basic Pitch är bättre på andra. Spotify standard Basic Pitch har som förväntat mindre skräpnoter med men missar vissa av noterna i ackorden som fångas upp av den optimerade pipelinen. Den optimerade pipelinen låter bättre och mer likt originalet men har en del spöknoter som överlevde filtreringen. För Jag Kommer presterar pipelinen sämre än Spotify standard Basic Pitch på grund av mycket spöktoner och dylikt.

Anledningen till dessa skillnader grundar sig troligen i pipelinens design och källsepareringens begränsningar. För akustisk fingerstyle är det ingen överraskning att pipelinen presterar bra då det är ren isolerad gitarrsignal utan effekter (som reverb och delay) vilket med andra ord är en väldigt fördelaktig miljö för Basic Pitch och de domänspecifika filtren kunde effektivt lösa de akustiska problem som ändå uppstod. För rock och funklåtarna verkar det som att behovet av 40 ms gränsen bekräftas då resultatet från pipelinen lyckas fånga upp snabba rytmiska passager som Spotify standard Basic Pitch missade. Att detta genererar fler spöknoter, varav vissa överlever filtreringen, är ett direkt resultat av den avsiktliga överdetektering (hagelbössan) som krävdes för att bevara låtarnas rytmik och toner. Att systemet presterade sämst på Jag kommer beror med största sannolikhet på begränsningar vid källsepareringen med HT-Demucs. I en tät popmix läcker oundvikligen frekvenser från synthesizers och bas in i det isolerade gitarrspåret. När detta brus möter pipelinens överkänsliga gräns på 40 ms så leder det till en massiv mängd falska noter som filterna inte lyckas radera då de primärt är designade för att hantera gitarrspecifika problem.

Sammanfattningsvis belyser dessa observationer de grundläggande utmaningarna med att uppnå ett helt generaliserbart resultat för musik “in the wild”. Verkliga gitarrinspelningar i moderna produktioner är sällan helt rena utan manipuleras ofta kraftigt

med effekter som distorsion, delay och reverb. Sådana effekter försvårar avsevärt modellens förmåga att avgöra exakta notavslut och separation mellan toner. Till detta tillkommer att källseparering med verktyg som Demucs ännu inte är helt perfekt. När pipelinen tvingas arbeta med ett isolerat gitarrspår som bär på både komplexa ljud effekter och oundviklig frekvensblödning från en helhetsmix, begränsas det nuvarande systemets effektivitet märkbart.

7.9.2 Hur blev taben?

Ju bättre MIDI-datan är desto bättre kommer den resulterande taben att bli och därav följer det att de låtar där pipelinen presterade väl som t.ex My Heart Will Go On hade den bästa taben. Men generellt sett så blev de genererade tabsen nästan aldrig likadana som tabs andra musiker har skrivit för dessa låtar. Alla tabs blir spelbara och är legitima sätt att spela låtarna på men är ofta inte det enklaste eller vanligaste sättet att spela en given låt på.

Att tabulaturerna blir fysiskt spelbara men ofta skiljer sig från hur musiker vanligtvis spelar låtarna förklaras troligen av Viterbi algoritmens begränsningar. Algoritmen minimerar en rent biomekanisk kostnadsfunktion som enbart tar hänsyn till handens horisontella och vertikala förflyttning samt användandet av öppna strängar. Den saknar dock helt musikalisk kontext. En mänsklig gitarrist väljer ofta fingersättningar baserat på bekväma skalboxar (standardiserade geometriska mönster över ett fåtal band), genrespecifika ackordformer (exempelvis powerchords i rock) eller mönster som är visuellt och kognitivt enkla att memorera. Eftersom algoritmen saknar kunskap om dessa mänskliga traditioner letar den enbart efter den matematiskt kortaste vägen över greppbrädan, vilket resulterar i fingersättningar som är effektiva i teorin men opraktiska i verkligheten.

7.10 Svar på frågeställningar

RQ1: Hur väl presterar den skräddarsydda filtreringspipelinen när det gäller att reducera andelen spöknoter (öka Precision) från den överkänsliga AMT-baslinjen (40 ms), utan att kompromissa med täckningsgraden (Recall)? Filtreringspipelinen presterar väl. Den överkänsliga AMT-baslinjen (40 ms) var nödvändig för att fånga upp korta snabba noter (vilket gav hög Recall), men den genererade oundvikligen en oacceptabel mängd spöknoter. De skräddarsydda domänspecifika filterna lyckades därefter effektivt rensa bort detta mekaniska och harmoniska brus vilket sänkte antalet felaktigt gissade noter markant. Detta gav i slutändan en högre F1-poäng för låtarna i GuitarSet jämfört med Spotifys standardmodell.

RQ2: Hur beroende är det avslutande MIDI-till-Tab-steget, och i vilken utsträckning kan Viterbialgoritmen identifiera och hantera fysiskt omöjliga ackord som genererats av AMT-modellen? Beroendeförhållandet är kritiskt. Viterbialgoritmen identifierar effektivt fysiskt omöjliga ackord men när den matades med den råa 40 ms datan kraschade systemet i över 11% av fallen för ackordspel. När MIDI-datan istället hade städats av filtreringspipelinen sjönk denna kraschfrekvens till under 1%. Resultaten bevisar att Viterbialgoritmen framgångsrikt kan beräkna ergonomiska fingersättningar men att processen är absolut beroende av filtrerad och logisk MIDI indata för att fungera.

RQ3: Hur väl generaliseras Tab-genereringsprocessen tillsammans med Demucs som försteg på ljudmixar, där gitarr inte är den enda ljudkällan?

Systemet verkar generalisera väl på gitarrdriven musik, som rock, funk och akustisk fingerstyle men har tydliga begränsningar. I genrer med tydlig gitarrmix lyckades källsepareringen kombinerat med pipelinen fånga majoriteten av tonerna som spelades både i ackord och snabba solo passager som missades av standard Basic Pitch. Systemets generaliserbarhet begränsas dock kraftigt av att både Basic Pitch och studiens framtagna filter har svårt att hantera vanliga gitarreffekter. Dessutom uppstår problem med frekvensblödning från andra instrument på grund av brister i källsepareringen. Detta skapar överdetektering från 40 ms-gränsen och resultatet blir en stor mängd spöktoner varav en del överlever filtreringen.

7.11 Svagheter, förbättringar och vidare forskning

De parameterval och den optimering som gjorts med hjälp av Optuna har potentiella nackdelar. Den data som användes för att generera de optimerade parametrarna användes också för att testa resultaten. Detta riskerar en icke-generaliserad lösning som är exakt specialiserade på just det datasetet som användes för att optimera parametrarna. I detta fall hade studien behövt mer data för att finna mer generaliserande lösningar och minska risken för att lösningen är specialiserad. En annan lösning hade kunnat vara att dela upp datan i en testmängd och en träningsmängd men då får man ännu mindre data att träna på.

En till potentiell svaghet i metodiken är att den genomförda optimeringen med Optuna kan ha konvergerat mot lokala maxima istället för att identifiera det sanna globala maximet. På grund av den stora sökrymden kan det inte garanteras att de framtagna parametervärdena utgör systemets absolut bästa teoretiska konfiguration. Istället bör resultaten betraktas som en mycket effektiv och fungerande approximation för denna studie, snarare än en absolut matematisk sanning.

Ett annat problem handlar om antagandet att nästa fingersättning endast beror på den nuvarande. Det antagandet gör att Viterbi algoritmen blir ett passande val för att finna nästa grepp, men detta antagande behöver stämma i verkligheten. Många låtar visar upprepande mönster av fingersättningar. Därför bör vidare forskning prova att använda ett större kontext och använda alla tidigare tillstånd (fingersättningar) för att finna mönster i spelandet. Mönster kan t.ex. vara upprepande ackord (samma fingersättning) eller delar i en låt (intro, vers eller refräng) då dessa brukar ha återupprepande ackord och melodier, bland annat. I detta arbete fungerade även Viterbi algoritmen endast som ett sista steg som var helt beroende av filtrerad indata. Ett mycket intressant område för framtida forskning vore att integrera Viterbi algoritmens biomekaniska kostnadsfunktion direkt i filtreringssteget. Genom att låta systemet värdera sannolikheten för en gissad spöknot baserat på hur svår den är att spela i kontexten av föregående noter, skulle Viterbi algoritmen aktivt kunna användas för att städa upp MIDI-datan istället för att enbart reagera på den.

Ett ytterligare område att undersöka är om det vore fördelaktigt att introducera en form av "ackord-ordbok" (exempelvis baserad på gitarrens traditionella CAGED-system) som mall för vanliga fingersättningar. På detta sätt hade tabulaturen kunnat tvingas följa en striktare konvention vilket många gitarrister är vana vid att spela.

En ytterligare begränsning är att studiens hårdkodade, heuristiska värden för fitlrena inte har utvärderats empiriskt. Att fixera dessa utifrån domänkunskap var ett nödvändigt metodval för att undvika en ohanterligt stor sökrymd för Optuna men det medför ett oanalyserat moment i pipelinen. För framtida forskning rekommenderas därför att utvärdera även dessa parametrar, exempelvis genom separata delstudier eller flerstegsoptimeringar som kan hantera fler variabler utan att beräkningskostnaden blir orimlig.

En ytterligare faktor som bedöms ha påverkat systemets prestanda är det inledande källsepareringssteget med Demucs. Under den tidigare nämnda granskningen av pipelinens generaliserbarhet observerades att separeringsmodellen hade svårt att helt isolera gitarren i komplexa ljudbilder utan att förlora viss musikalisk information. Detta fenomen var framförallt påtagligt i arrangemang med kraftiga gitarreffekter, dubblerade gitarrspår, eller där synthinstrument delade frekvensomfång med gitarren. För att minimera denna felkälla i framtida AMT-system rekommenderas vidare forskning att utveckla eller finjustera källsepareringsmodeller som är specifikt tränade för polyfont gitarrspel i täta mixar.

Vidare åtgärder som hade förbättrat den automatiserade tabulatur-genereringen är ett utökat dataset tillsammans med en förbättrad Audio-to-MIDI-modell då en mer träffsäker grundmodell än Basic Pitch hade genererat renare utdata från start.

Slutligen bör valet av utvärderingsmetrik specifikt F1-poängen via mir_eval, kritiseras ur ett musikaliskt perspektiv. Metriken kräver att en detekterad not ligger inom ett strikt fönster på ± 50 ms från referensnoten för att godkännas som korrekt. Detta bestraffar små rytmiska förskjutningar hårt även om de i praktiken låter musikaliskt acceptabla för det mänskliga örat. Dessutom är F1-poängen värdeneutral i sin bedömning av toner. Algoritmen straffas lika hårt för att missa en livsviktig grundton i ett ackord som för att missa en svag övergångston. Framtida forskning inom musikalisk transkribering bör sträva efter att utveckla utvärderingsmetriker som i högre grad betonar musikalisk kontext snarare än bara strikt matematisk överlappning.

7.12 Etik och samhällspåverkan

I och med att AMT-system blir bättre och bättre uppstår etiska och samhällsrelevanta frågor. Verktyg som denna pipeline har potential att förenkla musikinlärning genom att möjliggöra att vem som helst kan få tillgång till spelbara tabulaturer för sina favoritlåtar utan att behöva ha domänspecifik kunskap eller betala en stor summa pengar. Samtidigt så riskerar sådana här system att störa arbetsmarknaden negativt för musiker och pedagoger som arbetar med att transkribera och sälja tabulatur.

Ur ett dataperspektiv förlitar sig denna studie på förtränade modeller och öppet tillängliga dataset för utvärdering vilket säkerställer att inget direkt intrång av upphovsrättsskyddat material har använts under studiens gång. Dock bör det tilläggas att det inte finns några begränsningar för andra att använda sådana här modeller eller liknande pipelines för att generera transkriberingar av upphovsrättsskyddat material.

Slutligen bör miljöaspekten av maskininlärning beaktas. Denna studie demonstrerar att det är möjligt att förbättra transkriberingsprestandan hos en befintlig modell genom beräkningseffektiva, algoritmiska efterbehandlingsfilter. Genom att optimera dessa filter undveks den energikrävande processen att samla in ny data och träna en komplex AI-modell från grunden. För att möta framtidens behov på ett ansvarsfullt sätt rekommenderas att vidare forskning inom området fortsätter att utforska liknande resurssnåla metoder där befintliga modellers kapacitet maximeras med minsta möjliga beräkningskostnad.

Referenser

[1] R. M. Bittner, J. J. Bosch, D. Rubin m. fl., "A Lightweight Instrument-Agnostic Pitch Tracking Model", i *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2022, s. 311–315.

[2] C. Raffel m. fl., "mir_eval: A transparent implementation of common MIR metrics", i *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, ISMIR, 2014.

[3] A. Défossez, N. Usunier, L. Bottou och F. Bach, "Music Source Separation in the Waveform Domain", *arXiv preprint arXiv:1911.13254*, 2019.

[4] A. Défossez m. fl., *Demucs: Deep Extractor for Music Sources*, `https://github.com/facebookresearch/demucs`, Hämtad: 2026-04-11, 2022.

[5] T. Akiba, S. Sano, T. Yanase, T. Ohta och M. Koyama, "Optuna: A Next-generation Hyperparameter Optimization Framework", i *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19, New York, NY, USA: Association for Computing Machinery, 2019, s. 2623–2631. DOI: `10.1145/3292500.3330701`.

[6] Q. Xi, R. M. Bittner, J. Pauwels, X. Ye och J. P. Bello, "GuitarSet: A Dataset for Guitar Transcription", i *19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[7] H. Lou, "Implementations of the Viterbi algorithm", *IEEE Signal Processing Magazine*, årg. 12, nr 5, s. 42–52, sept. 1995. DOI: `10.1109/79.410439`. URL: `https://aiichironakano.github.io/phys516/Lou-Viterbi-SPM95.pdf`.

[8] D. Jurafsky och J. H. Martin, "Hidden Markov Models", i *Speech and Language Processing*, 3rd ed. draft. Stanford University, 2026, kap. Appendix A, Draft of January 6, 2026. URL: `https://web.stanford.edu/~jurafsky/slp3/A.pdf`.

[9] I. S. University, "Technical Reference: Piano Key Frequencies", Department of Electrical och Computer Engineering, Course Material (ECE 575), 2016, Hämtad: 2026-04-10. URL: `https://www.ece.iastate.edu/~alexs/classes/2016_Spring_575/HW/HW5/files/piano-key-freq-wikipedia.pdf`.

[10] N. Andreasson, *Övertoner och resonans*, Hämtad: 2026-04-10, 2009. URL: `http://www.niklasandreasson.se/html/overtone.html`.

[11] E. Benetos, S. Dixon, Z. Duan och S. Ewert, "Automatic music transcription: An overview", *IEEE Signal Processing Magazine*, årg. 36, nr 1, s. 20–30, 2019.

[12] Joey Sturgis Tones. "What Are Transients & Why Do They Matter to Your Mix?" Hämtad: 2026-04-11. URL: `https://joeysturgistones.com/blogs/learn/what-are-transients-why-do-they-matter-to-your-mix`.

[13] D. P. Radicioni och V. Lombardo, "Guitar Fingering for Music Performance", i *Proceedings of the International Computer Music Conference (ICMC)*, Barcelona, Spain, 2005, s. 527–530.

[14] Wikipedia-bidragsgivare, *Haas-effekten*, `https://sv.wikipedia.org/wiki/Haas-effekten`, [Hämtad; 2026-04-11], 2020.