Zeit-Frequenz Darstellungen und Deep Learning

Daniel Haider, Peter Balazs, Nicki Holighaus, Lorenz Gutscher
Institut für Schallforschung, 1040 Wien, Österreich

Einleitung

Maschinelles Lernen (ML) als Teilgebiet der künstlichen Intelligenz hat als generelles Ziel, einen Algorithmus bzw. eine Funktion im Sinne eines statistischen Modells zu finden, der Wissen aus Erfahrung generiert. In anderen Worten, man will eine Zuordnungsvorschrift finden, die eine bestimmte (Lern)aufgabe für gegebene Trainingsdaten so gut wie möglich löst und gut auf neue Daten angewandt werden kann. In den letzten Jahren hat sich eine simple aber besonders flexible Klasse an Modellen herauskristallisiert, mit denen man, theoretisch, beliebige Lernaufgaben für jede Art von Daten lösen kann - die (tiefen) neuronalen Netze (*deep neural networks - DNNs*). Begünstigt wurde diese Entwicklung durch den technologischen Fortschritt in Form größerer Rechenleistung, parallelisierter Verarbeitung, sowie der steigenden Anzahl an Datensätzen und frei zugänglichen Toolkits. Die Tiefe solch eines neuronalen Netzes bezieht sich dabei auf die mehrfache Aneinanderreihung von Schichten. Diese bestehen jeweils aus einer affin-linearen Abbildung, gefolgt von einer nicht-linearen Funktion. Während des Trainings werden die Gewichte der linearen Abbildungen (Einträge der Matrizen) so angepasst, dass die Trainingsdaten möglichst gut erfasst werden. Ein Spezialfall solcher Netze, der besonders für Anwendungen in der Audio- und Bildverarbeitung wichtig ist, sind faltungs-basierte neuronale Netze (*convolutional neural networks - CNNs*). Die linearen Abbildungen werden dabei durch Faltung mit Filterkernen realisiert. Wir wollen Deep Learning (DL) hier als ML Paradigma bezeichnen, welches DNNs als Lernmodelle verwendet. Im Allgemeinen erlaubt diese Modellklasse die Approximation beliebiger Funktionen, was aber nicht bedeutet, dass dies auf Basis eines gegebenen Datensatzes auch möglich ist. Im Regelfall benötigt das Training eines DNNs große Trainingsdatensätze, nicht zuletzt als Konsequenz dieser Flexibilität. DNNs sind somit nicht immer geeignet für eine gegebene Lernaufgabe. Für die meisten Anwendungen im Audiobereich basiert dennoch der derzeitige State-of-the-art zumindest zu gewissem Maß auf DL Methoden, was es eindeutig zum dominanten ML Paradigma macht.

Die Frage mit der wir uns hier beschäftigen ist nun, *wie* akustische Daten einem DNN übergeben werden. Dafür gibt es prinzipiell zwei verschiedene Herangehensweisen. Zum einen werden die Audiosignale direkt in abgetasteter Wellenform verwendet, was in der DL Community üblicherweise als *end-to-end* bezeichnet wird, weil das zu trainierende Modell von Anfang bis zum Ende alle Aufgaben selbst übernimmt. Im Gegensatz dazu können die Audiodaten, bevor sie dem Modell übergeben werden, mit Hilfe von Methoden aus der Signalverarbeitung sinnvoll aufbereitet werden um wichtige Informationen bereits aufzuschlüsseln. Man nennt dies einen *vorwissens-basierten* Zugang. Ein solcher Datenverarbeitungsprozess wird als *Feature Extraktion* bezeichnet und Eigenschaften und semantische Inhalte, die relevant für die Lernaufgabe sind, als *Features*. Da Zeit-Frequenz Darstellungen (ZFDen) die Grundlage für viele Methoden zur Analyse von Audio Informationen sind und dabei wichtige Relationen bereits aufschlüsseln, werden sie auch im DL standardmäßig als Eingangsdarstellung im Sinne von *Audio Features* von Audiosignalen verwendet. Der Prozess der Feature Extraktion findet aber auch im DNN selbst implizit statt. Bei end-to-end Modellen können diese *gelernten* oder auch *latenten Features* jedoch nur bedingt nachvollzogen werden und liegen somit außerhalb der Kontrolle des Benutzers. Es wird üblicherweise angenommen, dass dies in den ersten Schichten eines DNNs passiert und man kann sogar beobachten, dass hier unter Umständen eine Aufschlüsselung stattfinden kann, die man in weiterem Sinne als als Analogon einer ZFD interpretieren kann [8]. Diese generische Verbindung soll die Verwendung von ZFDen als Vorverarbeitung zusätzlich motivieren, nicht zuletzt um Kontrolle über die Audio Features zu haben. Zudem konnte gezeigt werden, dass damit im Vergleich zu einer end-to-end Variante ähnliche Erfolgsquoten bei geringerer Modellkomplexität und Menge an Trainingsdaten erzielt werden können [3, 6].

Zeit-Frequenz Darstellungen als Audio Features für Deep Learning

ZFDen bilden seit den 1960ern den Grundstein für digitale Audio-Signalverarbeitung und wurden seitdem stets weiterentwickelt und in zahllosen Anwendungen eingesetzt. Dabei werden die Frequenzinformationen eines Audiosignals lokal ausgelesen und mit der entsprechenden zeitlichen Abfolge dargestellt. Visualisiert wird eine solche Darstellung intuitiv in einem 2-dimensionalen Plot, in welchem mittels einer Farbkodierung die Intensität der jeweils auftretenden Frequenz angezeigt wird. Für uns Menschen ist die lokale Frequenzverteilung eine natürliche Visualisierung von akustischen Ereignissen, aus der wir mit etwas Übung die enthaltene Audio Information ablesen können. Dies scheint auch für DNNs zu gelten, da ZFDen nachweislich dazu beitragen die Lernaufgabe zu vereinfachen und somit die Optimierung effizienter zu gestalten. Wir geben nun ein Überblick darüber, welche ZFDen im DL häufig verwendet werden.

Spektrogramm

Das *Spektrogramm* ist die prominenteste aller ZFDen und in vielen Anwendungen die Standardwahl. Als Amplitude (manchmal auch quadriert) der Kurzzeit Fourier-Transformation (STFT) wird ein reguläres Gitter von Zeit-Frequenz Koeffizienten berechnet, die kleine Regionen auf der Zeit-Frequenz Ebene repräsentieren. Die Größe dieser Regionen hängt vom Fenster in der STFT

ab. Üblicherweise wird die STFT unterabgestastet, da die Anzahl der Datenpunkte dem Quadrat der des originalen Signals entspricht, was als *überaus redundant* erachtet wird. Dies wird in der STFT durch eine Sprungweite von mehr als einem Sample, sowie einer Beschränkung der Länge der schnellen Fourier-Transformation (FFT) bewerkstelligt. Die Skala, entlang der die Frequenz aufgetragen wird ist aufgrund der Natur der Fourier-Transformation linear, was nicht der Art und Weise entspricht, wie das menschliche Gehör zwischen Frequenzen unterscheidet.

Mel-Spektrogramm
Für das *Mel-Spektrogramm* wird eine logarithmische Frequenzskala durch gewichtete Frequenzmittelung des Spektrogramms konstruiert, sodass die Darstellung auf perzeptiv relevante Frequenzbänder komprimiert wird. Man kann zeigen, dass dadurch eine verbesserte Stabilität bei Frequenzverschiebungen und Deformationen gegeben ist, was das Mel-Spektrogramm zu einer sehr beliebten ZFD für Lernaufgaben macht, da gewisse Invarianzen bereits kodiert werden [1]. Oft wird sie mit einer zusätzlichen logarithmischen Transformation verwendet.

Mel-Frequency Cepstral Coefficients
Die vor allem bei Sprachverarbeitung oft verwendeten *Mel-Frequency Cepstral Coefficients* erhält man durch die Anwendung einer Kosinus-Transformation auf ein log-Mel-Spektrogramms in Frequenzrichtung. Die Darstellung wird dadurch zusätzlich komprimiert, wobei insbesondere die harmonische Struktur des Signals herausgearbeitet wird. Man kann so die spektralen Hüllkurven des Signals unabhängig von der Tonhöhe darstellen, was bei Sprache und Musik Hinweise auf die Formantenstruktur, bzw. die Klangfarbe liefert.

Skalogramm
Eine weitere bedeutende ZFD ist das *Skalogramm*, das als Amplitude der Wavelet Transformation entsteht. Die *Constant-Q Transformation* ist eine diskrete Realisierung der Wavelet Transformation und berechnet ein irreguläres Gitter von Zeit-Frequenz Koeffizienten mit einer angepassten Frequenzauflösung: Im niederfrequenten Bereich werden die harmonischen Strukturen besser aufgelöst und im hochfrequenten die zeitlichen. Man kann damit eine Darstellung konstruieren, die besonders interessant für musikalische Signale ist, indem man zu jedem Notenwert ein Fenster verwendet, dessen Länge invers proportional zur Frequenz des Notenwerts ist. Die resultierende Darstellung zeigt dann natürliche Distanzen zwischen den einzelnen Noten.
Es gibt noch viele weitere Arten von ZFDen, die hier genannten sind im DL jedoch die bei weitem Gängigsten.

Dass ZFDen in direkter Verbindung mit DNNs so besonders gut geeignet sind ist kein Zufall, denn die beiden Konzepte decken sich ganz allgemein in den zugrundeliegenden Rechenoperationen:

- Lineare Transformationen (Filterung)
- Sub-sampling (Abtastung)
- Pooling (Reduktion der Dimensionalität)
- Nicht-lineare Funktionen (punktweise).

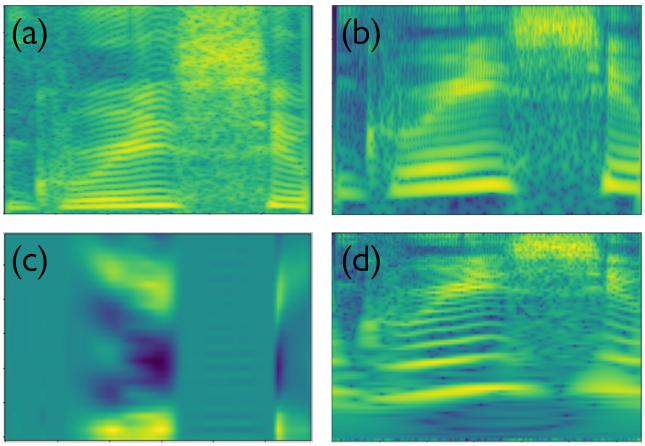

Abbildung 1: (a) Spektrogramm, (b) Mel-Spektrogramm, (c) MFCCs und (d) Skalogramm eines Sprachsignals.

Man kann das Spektrogramm beispielsweise mittels einer Filterbank realisieren und die Konstruktion der Mel-Skala als Pooling Operation interpretieren [3]. Diese intrinsische Verbindung macht faltungs-basierte neuronale Netze zu einer natürlichen Modellwahl für Audio Anwendungen, womit jede ZFD auch als eigenständiges flaches CNN mit festen Gewichten interpretiert werden kann [4]. Diese Sichtweise führt zu der Annahme, dass ZFDen als natürliche domänenrelevante Initialisierung eines CNNs erachtet werden können. Die Scattering Transformation stellt einen weiteren Schritt in dieser Sichtweise dar.

Scattering Transformation
Diese Transformation berechnet eine Kaskade von Filterungen und der Absolutsbetrags-Funktion (als nicht-lineare Funktion) über mehrere Schichten. Die entstehende Struktur gleicht vollständig jener eines tiefen CNNs, dessen Filter a priori festgelegt sind. Bei der Anwendung auf Audiosignalen kann man in den tieferen Schichten gut higher-level Features aus einer größeren Skala beobachten und analysieren [1]. Die zweite Schicht eines solchen Scattering Netzwerks wurde bereits erfolgreich als Input für DL Modelle verwendet, siehe e.g. [10]. Die natürliche

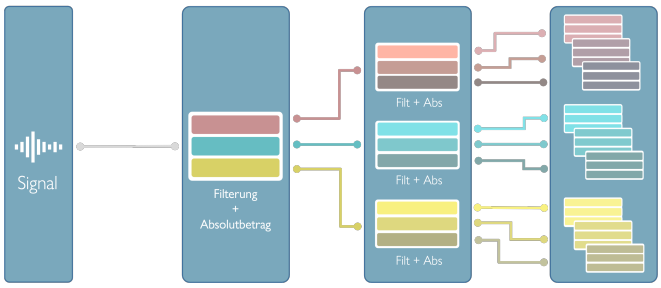

Abbildung 2: Schema der Scattering Transformation.

Verbindung von ZFDen und CNNs kann auch insofern ausgenutzt werden, dass man nur die prinzipielle Struktur einer Zeit-Frequenz Transformation vorgibt, die davon abhängenden Parameter aber vom Lernmodell optimiert. Vor allem Filterbank-Konstruktionen bieten sich dabei auf Grund ihrer großen Flexibilität an.

Filterbank-Lernen
Bei diesem Hybridansatz wird eine vorgegebene Filterbank (im Sinne einer ZFD zu verstehen) während des Trainings gemeinsam mit dem neuronalen Netz optimiert und so sukzessive aktualisiert. Dabei gibt es zwei verschiedene Zugänge. Beim *frei-geformten* Filterbank-Lernen wird nur die prinzipielle Form einer Filterbank fixiert und alle Gewichte der Filter gelernt [9]. Das entspricht genau genommen einer speziell geformten ersten Schicht eines CNN, kann aber auch als gelernte ZFD interpretiert werden. Beim zweiten Zugang wird eine parametrisierte Form einer Filterbank verwendet, bei der nur die freien Parameter (e.g. Bandbreite, zentrale Frequenzen der Bänder, Dilatationsfaktor, ...) dem Lernmodell als Input übergeben und somit gelernt. Dieser Ansatz wird *parametrisches* Filterbank-Lernen genannt [11]. Man kann so strukturierte Hybridsysteme mit beliebigen parametrisierten Transformationen konstruieren und so den end-to-end Zugang mit dem vorwissens-basierten verschmelzen.

Deep Learning mit Zeit-Frequenz Darstellungen - How To

Bei der Verwendung von ZFDen stellt die Wahl der passenden Parameter der zugrundeliegenden Transformation ein schwieriges Optimierungsproblem dar. Die Erfolgsquote des Lernmodells kann dadurch maßgeblich beeinflusst werden. Die große Kunst liegt demnach darin, den Feature Extraktions Prozess aufgabenspezifisch so zu entwerfen, dass relevante Features bestmöglich kodiert werden. Welche Art von ZFD mit welcher Parameterwahl die beste für bestimmte Lernaufgaben und Modelle sei, ist höchst aktuelles Forschungsgebiet, in dem laufend neue Ansätze publiziert werden. Im Folgenden geben wir einen groben Überblick darüber, worauf man bei der Verwendung von ZFDen achten sollte wenn man sie im Kontext von DL verwendet.

Einstellungen der ZFD
Die Wahl der Parameter der ZFD sollte sowohl an die Art der Signalklasse als auch an die Lernaufgabe selbst angepasst werden. Beim Spektrogramm werden dazu eine Fensterlänge und eine Zeitabtastungsrate festgelegt. Die Fensterlänge bestimmt dabei den Zeitraum, innerhalb dem Korrelationen erfasst werden sollen. Lange Fenster ermöglichen eine hohe Frequenzauflösung und stellen dadurch Grundfrequenz und harmonische Anteile gut trennbar dar, die transienten Anteile sind jedoch etwas verwaschen (Schmalband). Im Gegensatz dazu erhält man bei der Verwendung von kurzen Fenstern eine Darstellung mit einer hohen zeitlichen Auflösung und damit einem gegensätzlichen Effekt, die harmonischen Anteile sind verwaschen und schlechter trennbar, dafür werden etwa rhythmische Komponenten klarer dargestellt. Die Zeitabtastungsrate bestimmt die Überlappung der Fenster und somit die Redundanz der Darstellung. Tabelle 1 zeigt etablierte Einstellungen für diese beiden Parameter für Sprachen und Musik. Es gibt aber noch einen weiteren wichtigen, oft vernachlässigten Parameter der durch die Länge der FFT bestimmt wird: Die Anzahl der Frequenzbänder. Standardmäßig wird diese gleich der

	Fensterlänge	Zeitabtastung (Fensterüberlappung)
Sprache SB	40ms	2-8ms (80-95%)
Sprache BB	10-25ms	2-5ms (max. 80%)
Musik SB	60-100ms	3-20ms (80-95%)
Musik BB	40-60ms	2-12ms (80-95%)

Tabelle 1: Richtwerte für SSchmalband"(SB) und "Breitband"(BB) Spektrogramm.

Länge des Fensters gewählt, man kann die FFT Länge auch bewusst variieren, womit man neben neben der Fensterüberlappung noch eine weitere Möglichkeit erhält, die Redundanz der Darstellung zu steuern. Dies kann man sich bei vielen Anwendungen zunutze machen. Durch Reduktion der Anzahl der Frequenzbänder wird die Redundanz verringert, ohne Zeitauflösung zu vverlieren. Sollte man andererseits noch Kapazitäten für eine höhere Redundanz zur Verfügung haben, aber höhere Zeitabtastung bietet keine Verbesserung, so kann die FFT Länge erhöht werden und somit Separation harmonischer Signalanteile verbessert werden.
Darüber hinaus hat sich eine logarithmische Skalierung der Zeit-Frequenz Koeffizienten als nützlich erwiesen, was diese in eine (pseudo-)normalverteilte Form bringt. Ein zusätzlicher Normalisierungsschritt scheint jedoch nicht notwendig zu sein, da dieser von den meisten Modellen direkt vorgenommen wird [2].
Kenntnisse über die Charakteristiken der Audiosignale und darin enthaltene Zeit-Frequenz Informationen, die essentiell für die Lernaufgabe sind, sollten so in die Einstellungen der ZFD einfließen, um eine optimale Ausgangssituation für das Modell zu gewährleisten.

Modelldesign
Nach dem Erfolg der faltungs-basierten neuronalen Netze in der Bildverarbeitung, wurden sie auch direkt auf ZFDen angewandt. Die 2-dimensionalen Filterkerne, die mit den Bilder gefaltet werden, ermöglichen dem Modell lokale Korrelationen zu erfassen und weiter zu verarbeiten. Man muss hier jedoch beachten, dass ZFDen grundlegend andere Informationen enthalten als natürliche Bilder. Ein wesentlicher Unterschied sind die enthaltenen Invarianzen. Bildinformation ist für gewöhnlich invariant entlang beider Achsenrichtungen, wobei man Audioinformation auf der Zeit-Frequenz Ebene (in den meisten Anwendung) ausschließlich zeitinvariant annehmen kann - eine Translation in Frequenzrichtung erhält nicht notwendigerweise essentielle Informationen! In ZFDen sind zudem Zeit und Frequenz voneinander abhängig, anders als bei natürlichen Bildern. Man kann dies bei der Verarbeitung berücksichtigen, indem man Größe und Form der Filterkerne der ersten Schicht des CNNs an Zeit-Frequenz Information anpasst. Filter mit einer vertikalen Ausdehnung erfassen dabei vorwiegend Korrelationen in Frequenzrichtung, die beispielsweise Informationen über Klangfarbe und Obertöne beinhalten. Horizontale Filter erfassen Korrelationen in Zeitrichtung, die sich etwa auf Rhythmus und Verlauf der Grundfrequenz beziehen. In der Literatur wurden sowohl horizontale, vertikale, als auch quadratische Filter verwendet, eine

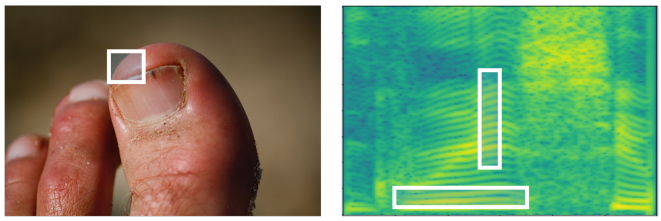

Abbildung 3: Natürliches Bild mit quadratischem Filter und ZFD mit rechteckigen Filtern.

besondere Tendenz geht jedoch zu 1-dimensionalen Filtern entlang der Zeitachse. [5]. Zudem sind ZFDen für gewöhnlich viel höher dimensional als Bilder. Deshalb ist es oft notwendig, das Netz tendenziell etwas tiefer zu gestalten als bei klassischer Bildverarbeitung, sodass auch Abhängigkeiten über weitere Bereiche erfasst und somit gelernt werden können. Das unterstützend wird oft eine dilatierte Version der Faltung verwendet, die ursprünglich für die diskrete Implementierung der Wavelet Transformation entwickelt wurde und der Faltungoperation eine zusätzliche Sprungweite verleiht um so weitere Abhängigkeiten mit derselben Filterlänge zu erfassen.

Ein weiterer Unterschied zwischen natürlichen Bildern und ZFDen bezieht sich auf die Lokalität und Überlagerung von Informationen. Ein Punkt in einem Bild repräsentiert üblicherweise nur Information zu einem einzigen Objekt, welches in der Regel gut von seiner Umgebung abgegrenzt werden kann. In einer ZFD verschwimmen die Grenzen der Zugehörigkeit oft, da die Darstellung aus einer Überlagerung von mehreren Zeitpunkten resultiert. Zusätzlich besteht ein "akustisches Objekt" auch immer aus mehreren voneinander separierten Regionen in der Zeit-Frequenz Ebene. Ein lokaler Filter, der für Bildinformation sinnvoll erscheint, ist für Zeit-Frequenz Information also im Allgemeinen eher suboptimal.

Auch beim Design des Lernmodells kann man also die Eigenschaften einer ZFD sinnvoll berücksichtigen. Um DL Modelle aber auch weiterhin effizienter, kontrollierbarer und somit nachhaltiger zu gestalten ist es wichtig Synergien zwischen konventionellen Modellen aus der Signalverarbeitung und modernen DL Modellen zu schaffen und auszunutzen und damit Domänewissen strukturiert einsetzen.

LTFAT in Python

Die Large Time-Frequency Analysis Toolbox (LTFAT) ist eine am Institut für Schallforschung entwickelte Toolbox für Matlab/Octave, die speziell für das Arbeiten mit Zeit-Frequenz Analyse und Synthese ausgelegt ist [7]. Sie enthält zahllose Funktionen und Transformationen rund um audiospezifische Signalverarbeitung und geht dabei weit über die Funktionalität der meisten anderen klassischen Signalverarbeitungstoolboxen hinaus. Derzeit ist ein Python Wrapper für LTFAT in Entwicklung, sodass diese Funktionalität in Zukunft auch leicht direkt in Python verfügbar ist und damit leichter in ein DL Projekt eingebunden werden kann. Wie sie derzeit, mithilfe einer Vorabversion des Python Wrappers, geladen und verwendet werden kann, wird in dem von uns vorbereiteten Google Colab File anhand einer Anwendung in der Audioklassifizierung demonstriert `https://colab.research.google.com/drive/1lL9OeEFhpGNtuxAFQSqAfSZIrca_JAqr?usp=sharing`.

Literatur

[1] J. Andén and S. Mallat. Deep scattering spectrum. *IEEE Transactions on Signal Processing*, 62(16):4114–4128, August 2014.

[2] K. Choi, G. Fazekas, and M. Sandler. A comparison of audio signal preprocessing methods for deep neural networks on music tagging. In *Proceedings of the 26th European Signal Processing Conference (EUSIPCO)*, 2018.

[3] M. Dörfler, T. Grill, R. Bammer, and A. Flexer. Basic filters for convolutional neural networks applied to music: Training or design? *Neural Computing and Applications*, 32:941–954, 2020.

[4] E. J. Humphrey, J. Bello, and Y. LeCun. Moving beyond feature design: Deep architectures and automatic feature learning in music informatics. In *Proceedings of the 13th ISMIR Conference*, 2012.

[5] G. Peeters and G. Richard. Deep learning for audio and music. In *Multi-faceted Deep Learning: Models and Data*. J. Benois-Pineau and A. Zemmari (Eds.), Springer, 2021.

[6] J. Pons, O. Nieto, M. Prockup, E. Schmidt, A. Ehmann, and X. Serra. End-to-end learning for music audio tagging at scale. In *Proceedings of the 19th ISMIR Conference*, 2018.

[7] Z. Průša, P. L. Søndergaard, N. Holighaus, C. Wiesmeyr, and P. Balazs. The Large Time-Frequency Analysis Toolbox 2.0. In M. Aramaki, O. Derrien, R. Kronland-Martinet, and S. Ystad, editors, *Sound, Music, and Motion*, Lecture Notes in Computer Science, pages 419–442. Springer International Publishing, 2014.

[8] H. Purwins, B. Li, T. Virtanen, J. Schlüter, S. Chang, and T. Sainath. Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219, 2019.

[9] T. N. Sainath, B. Kingsbury, A.-R. Mohamed, and B. Ramabhadran. Learning filter banks within a deep neural network framework. In *IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 297–302, 2013.

[10] B. Soro and C. Lee. A wavelet scattering feature extraction approach for deep neural network based indoor fingerprinting localization. *Sensors*, 19(8), 2019.

[11] T. Zhang and J. Wu. Discriminative frequency filter banks learning with neural networks. *EURASIP Journal on Audio, Speech, and Music Processing*, 2019:1, 2019.