# Musikgenerierung mittels KI

Michael Wagner

E-Mail: mw201@hdm-stuttgart.de  Matrikelnummer: 35118

## Einleitung

Schon immer strahlt Musik, die nicht direkt aus Menschenhand stammt eine gewisse Faszination aus. Die ersten Gehversuche wie etwa Windspiele oder Äolsharfen folgten keinen musiktheoretischen Regeln und waren ein reines Zufallsprodukt. Wolfgang Amadeus Mozart erzeugte gegen Ende des 18. Jahrhunderts einen Algorithmus, mit dem Zufall und menschliche Komposition verbunden wurden. Das musikalische Würfelspiel wurde seinerzeit zum Trend.

Da Musik sehr gut mathematisch beschreibbar ist, überrascht es kaum, dass bereits in den jungen Jahren der Informatik an Computerprogrammen getüftelt wurde, die Musik generieren sollten. Diese Faszination ist auch heute noch ungebremst, wenngleich sich der Trend weg von regelbasierten, hin zu Machine-Learning-Ansätzen verschiebt.

## Regelbasierte Systeme

In regelbasierten Ansätzen werden Musiktheorie und Kompositionsregeln in Programmcode übersetzt. Mit der Authentizität des Ergebnisses nimmt die Komplexität des Codes zu. Die Skalierbarkeit wird nach oben hin durch den Faktor Mensch gebremst.


*Abbildung 1: (Seychell, 2018)*

## Lernbasierte Systeme

Dem gegenüber stehen Machine-Learning-Modelle, meist künstliche *neuronale Netze* (NN). Hier wird ein System anhand von Beispielmaterial trainiert. Es erkennt Muster und Regeln eigenständig.

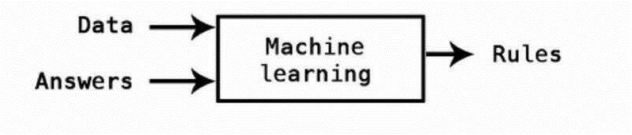


*Abbildung 2: (Seychell, 2018)*

Die Genauigkeit beziehungsweise musikalische Authentizität eines solchen Modells steigt mit der Skalierung des Systems. Das bedeutet vereinfacht gesagt: Je größer der Datensatz und das Netz, desto besser das Ergebnis.

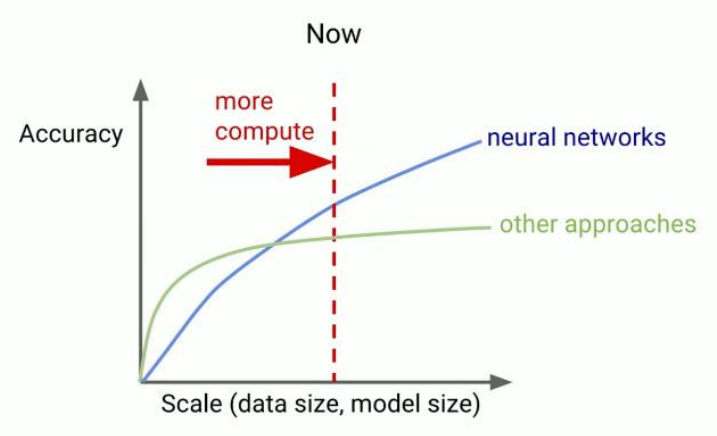


*Abbildung 3: (Google Developers, 2017)*

## Musikformate

Da nicht jedes Netzwerk für jedes Formate gleich gut geeignet ist, muss dieses im Vorfeld definiert werden. Die Spannweite reicht dabei von repräsentativen Formaten, die möglichst nahe an ein natürliches Klangereignis herankommen, wie zum Beispiel digitale Rohaufnahmen (.wav-Dateien), hin zu einer rein semantischen Darstellung der Musik (also ohne jegliches Audiosignal), wie beispielsweise eine Partitur.

Alle digitalen Formate lassen sich hierzwischen einordnen. In der Regel sind Input- und Output-Format identisch.

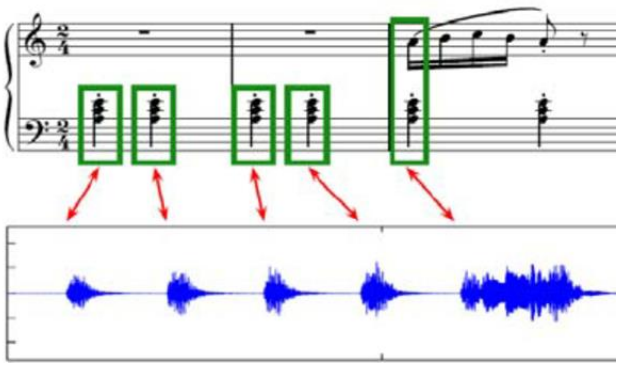

*Abbildung 4: (Fremerey, Müller, & Clausen, 2009)*

## Semantische Musikkomposition

Für eine semantische Musikkomposition wird nur selten eine Partitur verwendet. Wesentlich handlicher sind zum Beispiel Midi-Dateien. In ihnen ist nahezu die volle Semantik enthalten und das ohne Vorzeichen, Notenschlüsseln, etc. In Midi wird festgehalten, welche Note von wann bis wann gespielt wird. Dabei ist der *Beat* (=Taktschlag) die grundlegende Zeiteinheit. Das *Tempo* ergibt sich aus den Beats pro Minute (BPM). Die einzelnen Noten eines Klaviers sind von tief nach hoch durchnummeriert. So entspricht beispielsweise die Midi-Note 60 dem C5, welches wiederum ca. 261Hz zugeordnet ist. Die kleinste Auflösungsstufe eines Midi-Tracks ist der *Tick*. (Auflösung = Ticks pro Beat).

Um einen Machine-Learning-Algorithmus mit Midi-Dateien zu füttern, werden sie zunächst in eine Programmiersprache übersetzt. Python ist in diesem Bereich sehr beliebt. Der Datensatz wird anschließend in Spaltenvektoren überführt. Diese Vektoren geben Auskunft darüber, welche Noten zu einem konkreten Zeitpunkt aktiv sind. Da Midi-Ticks in relativer Zeit gespeichert sind, muss an dieser Stelle in absolute Zeit konvertiert werden (Auflösung · Tempo / 60 = Ticks pro Sekunde). Das Ergebnis ist eine 2D-Matrix, die als sogenannte Pianorolle visualisiert werden kann.

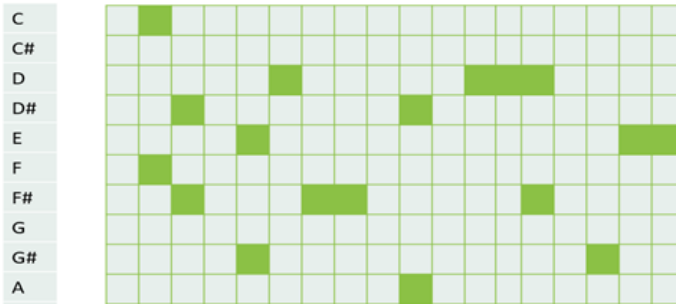


*Abbildung 5: (Microsoft (ML Blog Team), 2017)*

Als nächstes stellt sich die Frage nach einem geeigneten Modell. Frühe Modelle basierten auf einer sogenannten Markow-Kette. Diese sagt die jeweils folgende Note ausschließlich auf Grundlage der aktuellen voraus. Sie verfügt über kein ‚Gedächtnis'. Dasselbe gilt auch bei Feed-Foreward Neural Networks (FNN). Da Musik zeitlich gesehen sowohl eine Mikro- als auch Makrostruktur aufweist, kann dieses fehlende Gedächtnis problematisch sein. Denn erst im zeitlichen Kontext ergeben sich beispielsweise Harmonien und Akkorde, entsteht Konsonanz oder Dissonanz, etc. Dieser zeitliche Kontext trägt mehr Bedeutung als ein alleinstehender Momentanwert.

Wesentlich besser geeignet sind *Recurrent Neural Networks* (RNN). Durch ihre Rückkopplung können bereits vergangene Daten Einfluss auf einen aktuellen Output nehmen; das Netzwerk erhält ein ‚Gedächtnis'.

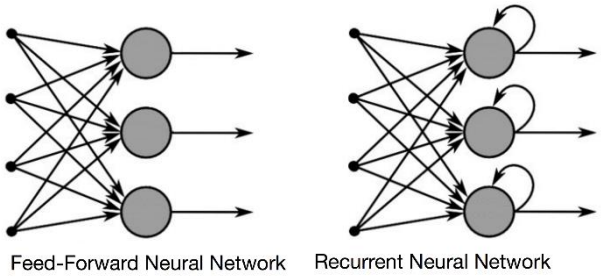


*Abbildung 6: (Donges, 2018)*

Viele Modelle zur Musikkomposition greifen auf ein erweitertes RNN zurück; das sogenannte *Long short term memory* (LSTM). Eine LSTM Zelle wird um drei Gates erweitert, die beeinflussen wie Information in dieser Zelle gespeichert wird. Das Input-Gate reguliert die Stärke, mit der ein neuer Input den Zellzustand beeinflusst. Das Forget-Gate gibt irrelevante Information in der Zelle zur Löschung frei. Und das Output-Gate reguliert wie viel vom Input direkt an das nächste Modul weitergeleitet wird. Durch diese Speicherfunktion wird das ‚Gedächtnis' gegenüber herkömmlichen RNNs verlängert, weshalb sie auch besser für Signale aus längeren Sequenzen geeignet sind.

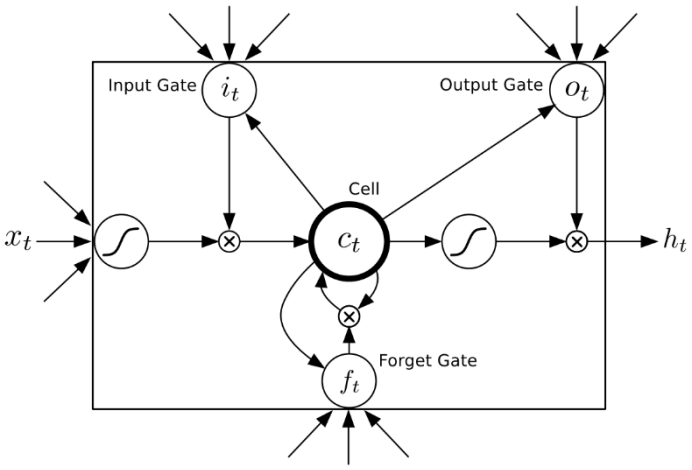


*Abbildung 7: (Ogunmolu, Gu, Gans, & Jiang, 2016)*

Wurde ein geeignetes Modell gewählt, so kann dessen Training beginnen. Hierfür wird der Algorithmus mit einem Datensatz an Musik gefüttert. Ziel ist es, die Musik möglichst genau durch den Algorithmus zu replizieren. In Phase 1, der *Foreward Propagation* generiert der Algorithmus aus einem Input-Fragment einen eigenen Output. Das Ziel hierbei ist es, das Original zu rekonstruieren. In Phase 2 wird dieser Output mit dem Original verglichen und ein globaler Systemfehler ermittelt. Eine Optimierung findet mittels Backpropagation in Phase 3 statt: Für jedes Neuron wird schichtweise, von hinten nach vorne in jedem *Hidden-Layer* der Fehler bestimmt. Anschließend werden die Gewichtungen entsprechend angepasst. In- und Output nähern sich an.

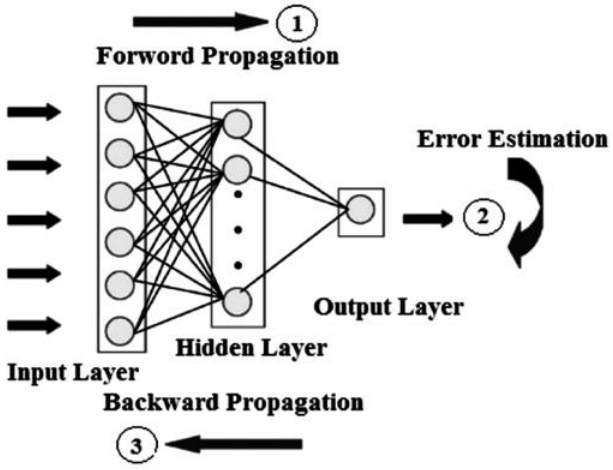


*Abbildung 8: (Donges, 2018)*

Zur Generierung neuer Musik greifen die meisten Modelle auf einen *Primer* zurück. Dies ist eine kurze Notenfolge, die als Initiator dient. Der generierte Output wird anschließend wieder als Input dem System zugeführt. Dadurch führt das System eigenständig Melodie oder Harmonie weiter.

Anwendungen wie beispielsweise *Deep Bach*, *Bach Bot* und *Bot Dylan* beweisen, dass einzelne Stile und Komponisten sich recht gut imitieren lassen und Googles *Performance RNN* zeigt, dass KI auch expressive Musik komponieren kann. Insgesamt bewegen sich die KI-Kompositionen jedoch auffällig stark im Jazz und Klassik. Eine mögliche Erklärung hierfür ist, dass solche Machine-Learning Modelle noch Probleme mit allzu starren, repetitiven Strukturen haben, wie sie etwa in der Popmusik üblich sind.

Ein wichtiger Vorteil ist, dass der Output bei Bedarf sehr einfach händisch korrigiert oder bearbeitet werden kann. Zudem sind semantische Modelle verhältnismäßig leicht und ressourcenschonend trainierbar.

## Klangsynthese

Erzeugt ein Algorithmus hingegen ein künstliches Audiosignal, so spricht man von Klangsynthese. Durch die Verwendung von Audiodateien ergeben sich im Vergleich zur rein semantischen Komposition kritischere Problemstellungen. Bereits kleine Veränderungen auf Samplebasis führen zu einer Veränderung der Wellenform. Schlimmstenfalls wird das Audiomaterial hierdurch völlig unbrauchbar. Die Samples können analog zu den Midi-Ticks betrachtet werden. Letztere bewegen sich typischerweise in Größenordnungen von ca. 1000 Ticks pro Sekunde. Das digitale Audiosignal schwingt hingegen mit einer weitaus höheren Geschwindigkeit. Ab 44,1 kHz wird von CD Qualität gesprochen. Aus Leistungsgründen wird die Synthese mit MAchine-Learning-Algorithmen heute noch auf 16 kHz beschränkt. Da das Nyquist-Shannon-Theorem nicht erfüllt ist kommt es zu tiefen Alias-Frequenzen. Das Signal klingt insgesamt ‚rau'. Es wird auch deutlich, dass ein RNN weitaus mehr Speichervolumen mitbringen muss, um den zeitlichen Kontextes von Musik abbilden zu können. Tatsächlich ist der Aufwand derart hoch, dass sie für KI-gesteuerte Klangsynthese selten das Mittel der Wahl sind.

2016 stellte Deep Mind mit *WaveNet* eine neue Architektur zur Sprachsynthese vor, mit der eine höhere Abstraktionsebene von Audio auf Samplebasis

möglich wurde. Dies diente beispielsweise als Grundlage für Googles *NSynth*. Ein kurzes Audio-Snippet, z.B. eine Klaviernote wird in eine Spektraldarstellung überführt. Diese Darstellung kann wie ein Bild behandelt und weiterverarbeitet werden. Hierfür eignen sich *Convolution Neuronal Networks* (CNN).

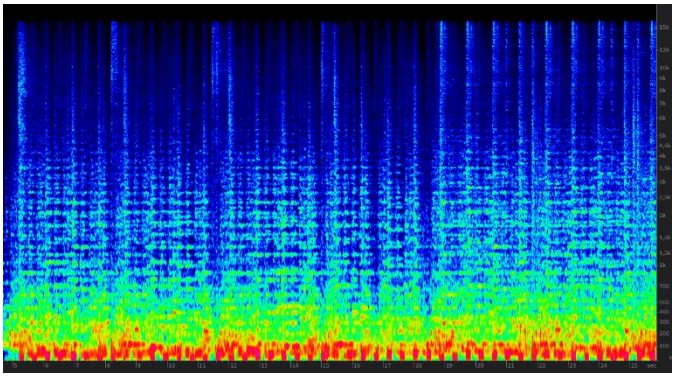

*Abbildung 9: Eigene Grafik*

Wie andere Filter auf Faltungsbasis modelliert NSynth Sounds Sample für Sample. Dazu wird ein modifizierter WaveNet Encoder im Eingang genutzt. Dieser wurde so umgewandelt, dass er stets den kompletten Kontext eines Input-Snippets sieht. Dieser Kontext ist momentan noch auf ca. eine halbe Sekunde beschränkt. Signale die darüber hinausgehen benötigen ein externes Steuersignal. Im Ausgang findet sich ein herkömmlicher WaveNet Decoder.

Der Encoder findet nun für einen Input eine komprimierte Z-Repräsentation. In dieser befindet sich die volle spektrale Semantik oder anders ausgedrückt, die frequenzabhängigen Charakteristika eines Sounds. Das System wird anschließend so trainiert, dass der Output den Input möglichst gut annähert.

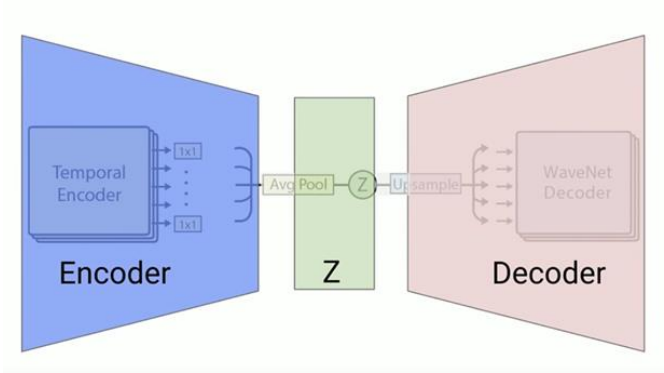


*Abbildung 10 (Google Developers, 2017)*

NSynth hat nicht die Absicht komplette Musikstücke zu generieren. Da der WaveNet Encoder nicht mit Notenwechseln im Sinn entwickelt wurde, kann das Modell diese nicht ohne Artefakte und Glitches reproduzieren. Vielmehr soll der NSynth Musikern Tür und Tor zu neue Klangwelten öffnen. Dies geschieht indem Sounds miteinander gemischt werden. Während beim herkömmlichen Mischen in der Audiotechnik zwei Signale linear addiert werden, können hier durch die Addition der Z-Transformierten neue einzigartige Sounds kreiert werden.

## Fazit

Während die rein semantischen Kompositionen bereits beeindruckende musikalische Authentizität erreichen, sind sie immer noch auf eine menschliche Performance oder mindestens sehr gute virtuelle Instrumente gebunden, um ein befriedigendes Audiosignal zu erreichen. Auf der anderen Seite befinden sich Klangsynthesemodelle, die zwar Audiomaterial generieren, jedoch extrem aufwändig sind (komplexer Code, viel Rechenleistung und –zeit) und deshalb noch nicht in der Lage sind ganze (und hochqualitative) Musikstücke zu erzeugen. Wie immer liegt die Wahrheit in der goldenen Mitte: Wenn beide Technologien weiter fortschreiten und kombiniert werden, ein Modell also musikalische Semantik erlernt und durch Klangsynthese glaubhaft umsetzt, dann ist die Musik einer KI vermutlich nichtmehr von menschgemachter zu unterscheiden. Bis dahin ist der Faktor Mensch noch notwendig um ansprechende Musik zu schaffen. Doch die Zukunft liegt bereits in Sichtweite.

# Verweise


(GitHub), l. (29. Januar 2017). *Python MIDI*. Abgerufen am 15. Dezember 2018 von https://github.com/louisabraham/python3-midi

Babuschkin, I. (6. April 2018). *A TensorFlow implementation of DeepMind's WaveNet paper*. Abgerufen am 12. Dezember 2018 von https://github.com/ibab/tensorflow-wavenet

carykh. (4. Juli 2017). *AI evolves to compose 3 hours of jazz!* Abgerufen am 14. Dezember 2018 von https://www.youtube.com/watch?v=nA3YOFUCn4U

CodeParade. (24. Juli 2018). *Generating Songs With Neural Networks (Neural Composer)*. Abgerufen am 16. Dezember 2018 von https://www.youtube.com/watch?v=UWxfnNXlVy8

Donges, N. (25. Februar 2018). *Recurrent Neural Networks and LSTM*. Abgerufen am 10. Dezember 2018 von https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5

Fremerey, C., Müller, M., & Clausen, M. (2009). Towards Bridging the Gap between Sheet Music and Audio. *Knowledge Representation for Intelligent Music Processing.* Abgerufen am 12. Dezember 2018 von https://www.researchgate.net/profile/Michael_Clausen/publication/221025901/figure/fig1/AS:305653949190146@1449884988930/Illustration-of-the-sheet-music-audio-synchronization-by-means-of-the-first-four-measures.png

Google Brain; DeepMind. (6. April 2017). *NSynth: Neural Audio Synthesis*. Abgerufen am 12. Dezember 2018 von https://magenta.tensorflow.org/nsynth

Google Developers. (19. Mai 2017). *Project Magenta: Music and Art with Machine Learning (YouTube)*. Abgerufen am 27. Februar 2019 von https://www.youtube.com/watch?v=2FAjQ6R_bf0

Hawthorne, C., Roberts, A., & Roberts, A. (30. Oktober 2018). *The MAESTRO Dataset and Wave2Midi2Wave*. Abgerufen am 15. Dezember 2018 von https://magenta.tensorflow.org/maestro-wave2midi2wave

Mann, Y. (Mai 2017). *NSynth: Sound Maker*. Abgerufen am 16. Dezember 2018 von https://experiments.withgoogle.com/sound-maker

Microsoft (ML Blog Team). (6. Dezember 2017). *Music Generation with Azure Machine Learning*. Abgerufen am 15. Dezember 2018 von https://blogs.technet.microsoft.com/machinelearning/2017/12/06/music-generation-with-azure-machine-learning/

Microsoft Developer. (8. Februar 2018). *Deep Learning for Music Generation*. Abgerufen am 12. Dezember 2018 von https://www.youtube.com/watch?v=4bCrNl4Bx1M

Ogunmolu, O., Gu, X., Gans, N., & Jiang, S. (Oktober 2016). *Nonlinear Systems Identification Using Deep Dynamic Neural Networks*. Abgerufen am 16. Dezember 2018 von https://www.researchgate.net/publication/308896333_Nonlinear_Systems_Identification_Using_Deep_Dynamic_Neural_Networks

Olah, C. (27. August 2015). *Understanding LSTM Networks*. Abgerufen am 13. Dezember 2018 von http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Raval, S. (23. September 2016). *Generate Music in TensorFlow*. Abgerufen am 12. Dezember 2018 von https://www.youtube.com/watch?v=ZE7qWXX05T0

Raval, S. (10. März 2017). *How to Generate Music - Intro to Deep Learning #9*. Abgerufen am 15. Dezember 2018 von https://www.youtube.com/watch?v=4DMm5Lhey1U

Raval, S. (15. März 2017). *How to Generate Music with Tensorflow (LIVE)*. Abgerufen am 26. Februar 2019 von https://www.youtube.com/watch?v=pg9apmwf7og

Raval, S. (25. Mai 2018). *AI for Music Composition*. Abgerufen am 15. Dezember 2018 von https://www.youtube.com/watch?v=NS2eqVsnJKo

Sandred, Ö., Laurson, M., & Kuuskankare, M. (kein Datum). Revisiting the Illiac Suite – a rule based approachto stochastic processes. Abgerufen am 27. Februar 2019 von http://sandred.com/texts/Revisiting_the_Illiac_Suite.pdf

Seychell, D. (22. September 2018). *Getting started in AI: 2018*. Abgerufen am 27. Februar 2019 von https://hitsingularitydotcom1.files.wordpress.com/2017/10/img_2076.jpg?w=800&h=366

Simon, I., & Oore, S. (29. Juni 2017). *Performance RNN: Generating Music with Expressive Timing and Dynamics*. Abgerufen am 14. Dezember 2018 von https://magenta.tensorflow.org/performance-rnn

Wren, C. (21. Mai 2017). *Visualising MIDI files with Python*. Abgerufen am 15. Dezember 2018 von https://medium.com/@colinwren/visualising-midi-files-with-python-b221feacd762