Neuronale Netze in der Videoproduktion

Laura Anger
Technische Hochschule Köln
Institut für Medien- und Phototechnik
laura.anger@th-koeln.de

Vera Brockmeyer
Technische Hochschule Köln
Institut für Medien- und Phototechnik
vera.brockmeyer@smail.th-koeln.de

Zusammenfassung

Vera

Auf Grund des hohem Zeit- und Kostenaufwand der Videoproduktion ist es notwendig einzelne Teilprozesse zu vereinfachen und zu automatisieren. Gerade in den Teilbereichen Produktion und Postproduktion gibt es enormes Potential für eine Automatisierung. Diese beschäftigen sich überwiegend mit der Erstellung und Bearbeitung von Bild- und Audiodaten. In den letzten Jahren konnten enorme Erfolge in der Bild- und Audiodatenverarbeitung mit Neuronalen Netzen (NN) erzielt werden. Im besonderen erwiesen sich Faltungsnetze (CNN) als zuverlässige und robuste Automatisierungsalgorithmen. Diese CNN können sowohl zur Klassifikation als auch zur Videodatengenerierung genutzt werden. Verfahren die Deep Dream und Deep Style können die Stilsemantik eines Bildes auf ein weiteres unabhängiges Bild übertragen. Während Deep Style viele Kriterien der Videoproduktion abgesehen vom hohen Rechenaufwand erfüllt, ist das Deep Dream Verfahren durch mangelnde Reproduzierbarkeit und Einflussnahme ungeeignet. Neben der Stilsnythese können auch Szenendynamiken aus einem Einzelbild produziert werden für die Generierung von kurzen Sequenzen. Diese benannten Verfahren sind nicht für die Videoproduktion entwickelt worden und werden detailliert beschrieben und auf ihre Tauglichkeit für die professionelle Produktion.

Schlüsselwörter

Faltungsnetze, Videoproduktion, Stilsynthese, Bewegtbildgenerierung, Deep Writing

1. Einleitung

Vera

Die Videoproduktion konnte sich im Zuge der Digitalisierung im letzten Jahrzehnt enorm qualitativ verbessern. Jeder der vier Produktionsabschnitte (siehe Abbildung 1) konnte verbessert und nachhaltig erleichtert werden. In der Konzeptionsphase konnten die Arbeitsprozesse mit Hilfe des Internets erleichtert werden durch den beschleunigten weltweiten Austausch von Skripten oder den standort-unabhängigen Zugriff auf cloudbasierte Projektmanagement Systeme. Während der Produktion des Videomaterials unterstützen moderne digitale Kamera- und Produktionssysteme aktiv den Kameramann. Die darauffolgenden Arbeitsprozesse, wie das Schneiden und Editieren des Videomaterials währen der Postproduktionsphase, wurden in den letzten Jahren vereinfacht oder können teilweise durch zuverlässige Algorithmen automatisch durchgeführt werden. Mittlerweile können durch *Computer Generated Imaging* (CGI) Produktionen fast ausschließlich im Studio zu produziert und selbst aufwendige Fantasywelten oder aufwendige Stunts mit geringeren Kosten umgesetzt werden.

Doch gerade qualitativ hochwertige Videoproduktionen erfordern immer noch einen sehr hohen Arbeitsaufwand mit einer große Anzahl an professionellen Mitarbeitern und kostenaufwändigen Materialien. Einen großen Anteil daran hat die Postproduktion in der jede Szene separat editiert und an das Gesamtbild angepasst werden muss. Dieses Gesamtbild muss vorab genau festgelegt werden und erlaubt keine Experimente in der Stilfindung. Aber auch die Generierung von Bildmaterialien für kurze Schnittszenen oder Webvideos ist sehr zeitaufwändig und kostenintensiv. Ein mehrköpfiges Team mit dem umfangreichen Equipment muss zum Drehort transportiert und untergebracht werden. Auch äußere Einflüsse, wie zum Beispiel das Wetter, können den Zeitplan verzögern und zusätzliche Kosten verursachen. Für Studioaufnahmen müssen in der Regel entsprechende Ressourcen angemietet oder in Stand gehalten werden.

In der Zukunft gilt es diesen Arbeits- und Kostenaufwand weiter zu reduzieren indem die einzelnen Arbeitsschritte automatisiert oder teil-automatisiert werden. Eine andere Vision ist es kurze Videoszenen für Webvideos oder Schnittszenen künstlich auf Basis von einzelnen Photographien am Computer zu erstellen. Dies erfordert Ansätze die komplexe Zusammenhänge und Erfahrungen

ähnlich wie das menschliche Gehirn leisten können. Sie sollten kreativ sein, Bewegungen und Abläufe voraussagen, bekannte Eigenschaften sinnvoll kombinieren oder erlernte Informationen übertragen und anwenden können. Diese Anforderungen erfüllen künstlichen Intelligenzen, wie NN (siehe Abschnitt 2.3), die jenen des menschlichen Gehirn nachempfunden sind [14]. In den letzten Jahren wurden sie stetig weiterentwickelt und es konnten vor allem im Bereich der Medienproduktion vielversprechende Erfolge erzielt werden. Die größten Erfolge konnten mit CNN erzielt werden. Diese (siehe Abschnitt 2.3) ermöglichen orts- und skalierungs-unabhängige Operationen und somit ideal für mehrdimensionale digitale Signale geeignet.

Zu Beginn wurden CNN zur Objektklassifizierung eingesetzt um unter anderem automatisch Metadaten von Bild- oder Videodaten zu generieren und in Datenbanken oder Suchmaschinen einzupflegen. In den letzten Jahren wurden sie auch verstärkt für die Generierung oder Fortsetzung von bekannten Daten oder Signalen eingesetzt. Es konnten klassische Musikstücke sinnvoll beliebig verlängert werden [27] oder bewegte Sequenzen aus einzelnen Bildern generiert werden [29]. Auch in der Postproduktion konnten Bildern einer Stil aufgeprägt werden [15].

In den folgenden Kapiteln wird in den Grundlagen (siehe Kapitel 2) auf den allgemeine Ablauf in der Videoproduktion sowie detailliert die Funktionsweise der NN und CNN beschrieben. Im Anschluss werden in den folgenden Kapiteln verschiedene Entwicklungen von Videoproduktionsmittel vorgestellt, welche verschiedene Formen von NN und im besonderen von CNN nutzen. Drei Ansätze werden detailliert beschrieben und bewertet. Der erste Ansatz [29] beschreibt in Anschnitt 4.2 die Generierung von eine bewegten Bildsequenz aus einem Einzelbild. Die anderen Ansätze [15] [7] erläutert die Übertragung eines Bildstils auf eine andere Videosequenz.

2. Grundlagen

Laura

Um gezielter Ansatzpunkte für den Einsatz von NNs in der Videoproduktion aufzeigen zu können, wird diese in Kapitel 2.1 kurz beschrieben. Der Schwerpunkt dieses Kapitels liegt jedoch auf den Grundlagen von NNs und insbesondere CNNs, sowie deren Training.

2.1. Videoproduktion

Laura

Mit der Videoproduktion oder auch Filmproduktion wird die Herstellung sowohl von Kino- als auch von Werbe- und

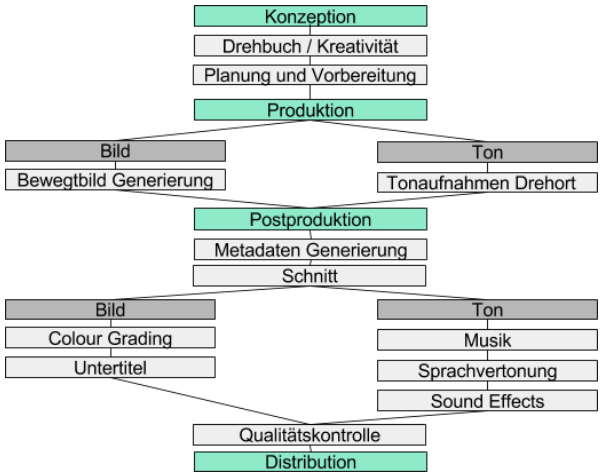

Abbildung 1: Grundlegende Arbeitsschritte einer Videoproduktion.

Fernsehfilmen zusammengefasst. In Abbildung 1 ist ein Ablaufplan einer typischen Videoproduktion zu sehen. Da es alleine in Deutschland über 850 Produktionsformen gibt (Stand 2014) [22], kann der Ablaufplan nur einen sehr allgemeinen Überblick über die notwendigen Arbeitsschritte bieten.
Der erste Schritt, die Konzeption soll sowohl die Projektentwicklung, als auch die Vorproduktion zusammenfassen. Die sich anschließende Produktionsphase kann grob, wie im Schaubild zu sehen in Bild und Ton unterteilt werden, wobei diese beiden Bereiche nicht immer getrennt voneinander betrachtet werden sollten. Die Postproduktion besteht aus vielen verschiedenen Arbeitsschritten, deren Schwerpunkt auf dem Schnitt und der digitalen Bildnachbearbeitung liegt. Der Schritt der Distribution ist hier der Vollständigkeit halber erwähnt und beschreibt die Filmverwertung.
Der Aufbau der folgenden Ausführungen orientiert sich an Abbildung 1 (vgl. Kapitel 3, 4 und 5).

2.2. Neuronale Netze

Laura

NNs finden unter anderem Anwendung bei der Steuerung von Robotern, Börsenkursanalysen, Medizin oder Fahrzeugsteuerung. In der Bildverarbeitung werden NNs vor allem zur Klassifizierung genutzt.
Sie sind vom menschlichen Gehirn inspiriert, welches laut [9] ein nicht-lineares, komplexes und hoch paralleles System zur Verarbeitung von Informationen darstellt. Ähnlich wie dieses bestehen künstliche NNs aus einer Menge an simulierten Neuronen, die untereinander verbunden sind und in Schichten organisiert sind. Es gibt verschiedene Arten der Vernetzung, die, wie in [9] und [21] beschrieben, in rück- und vorwärts gekoppelte Modelle unterteilt werden

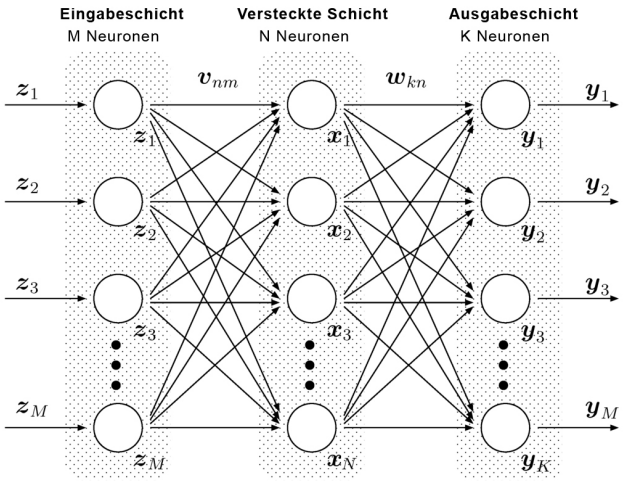

Abbildung 2: Prinzipieller Aufbau MLP nach [10].

können.

Am häufigsten kommen sogenannte *Multilayer Perceptrons* (MLP) [2][16][21] zum Einsatz. Ein generalisierter Aufbau ist beispielhaft in Abbildung 2 zu sehen. Dieses MLP besteht aus einer Eingabe- und Ausgabeschicht mit M bzw. K Neuronen und einer versteckten Schicht mit N Neuronen. Es handelt sich um ein vorwärtsgekoppeltes Modell, bei welchem jedes Neuron einer Schicht mit jedem Neuron der darauffolgenden Schicht verbunden ist. Dies nennt man volle Verbindung.

Die Eingangsschicht dient zum Verteilen der Daten z_m mit $m = 1, ..., M$. Die Ausgabe eines jeden Neurons in der versteckten Schicht, dargestellt durch x_n mit $n = 1, ..., N$, lässt sich durch Formel 1 berechnen. Hierbei steht v_{nm} für die jeweilige Gewichtung der Verbindungen zwischen den Neuronen der Eingabe- und der versteckten Schicht und f für die Aktivierungfunktion [21][9] des jeweiligen Neurons.

$$x_n = f\,(\sum_{m=1}^{M} v_{nm} z_m) \tag{1}$$

Die Ausgangswerte y_k, mit $k = 1, ..., K$, lassen sich äquivalent unter Hereinnahme der Werte x_n und der Gewichte w_{kn}, sowie einer Aktivierungsfunktion g berechnen, und gelten als Vertrauenswerte. Sie müssen gemäß der Aufgabenstellung interpretiert werden.

2.3. Faltungsnetze

Laura

Im Folgenden wird genauer auf CNNs eingegangen, da diese die Grundlage, für die meisten der in den folgenden Kapitel vorgestellten Ansätze, bilden. Vereinfacht ausgedrückt besteht ein CNN aus einer Vernetzung von Faltungsoperationen mit unterschiedlichen Filtermasken. CNNs kommen, bedingt durch ihre Architektur, oft zum

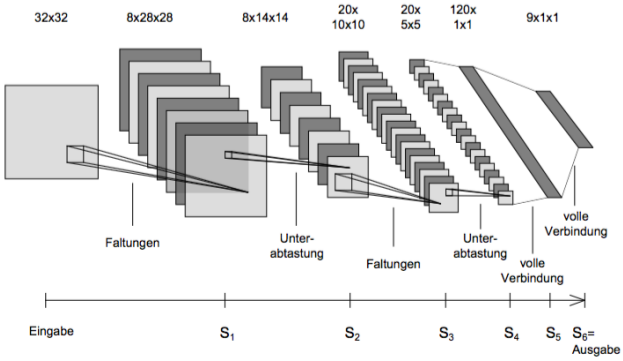

Abbildung 3: Prinzipieller Aufbau Faltungsnetz nach [18].

Einsatz, wenn große Datenmengen von einem NN verarbeitet werden sollen. Ein schematischer Aufbau ist in Abbildung 3 zu sehen.

Jedes Pixel eines Feldes, das auf der Abbildung zu sehen ist, wird durch ein Neuron repräsentiert. Die Felder sind in Schichten organisiert. Die Eingangsschicht fungiert, vergleichbar mit den MLPs aus Kapitel 2.2, als Verteiler der Information an die Neuronen der nächsten Schicht S_1. Die Besonderheit eines CNNs sind die sich abwechselnd durchgeführte Faltung und anschließende Unterabtastung. Zwischen den Schichten S_4 und S_6 ähnelt das Modell einem MLP, da die Neuronen schichtweise voll verbunden sind.

Im Allgemeinen wird für eine Faltung eine Filtermaske h, also ein endlicher, zweidimensionaler Koeffizientensatz, wie in Formel 2 zu sehen, verwendet. Hierbei stehen x und y jeweils für die horizontale bzw. die senkrechte Bildkoordinate. Die Anzahl der Koeffizienten a_{xy}, wird in der Horizontalen mit N_{hx} und im Vertikalen mit N_{hy} bezeichnet.

$$h(x,y) = \begin{cases} 0 \; für \; x < -\lfloor \frac{N_{hx}-1}{2} \rfloor \; \vee \; y < -\lfloor \frac{N_{hy}-1}{2} \rfloor \\ 0 \; für \; x > \lfloor \frac{N_{hx}-1}{2} \rfloor \quad \vee \; y > \lfloor \frac{N_{hy}-1}{2} \rfloor \\ a_{xy} \end{cases} \tag{2}$$

Formel 3 beschreibt die Faltung eines Eingangssignals s mit einer Filtermaske h, wobei I das Ausgangssignal in Abhängigkeit von x und y beschreibt.

$$I(x,y) = (s * h)(x,y) = \sum_{m_x=-\infty}^{\infty} \sum_{m_y=-\infty}^{\infty} s(m_x, m_y) \cdot h(x - m_x, y - m_y) \tag{3}$$

Im Fall eines CNNs wird die Faltung, die wie in Abbildung 3 zu sehen, beispielsweise zwischen der Eingangsschicht und S_1 vollzogen wird, durch die Verbindung zwischen den Neuronen zweier Felder modelliert. Dabei entsprechen die Gewichte der Neuronen genau den Filterkoeffizienten a_{xy}. Für ein jedes Feld sind diese Koeffizienten konstant, was bedeutet, dass alle Neuronen eines Feldes mit nur einem Gewicht auskommen. Dieses Prinzip nennt man geteilte Gewichte.

Im CNN wird nach jeder Faltung eine Unterabtastung

durchgeführt um zu gewährleisten, dass die Dimension der Eingangsdaten schrittweise an die Dimension des Ausgangsvektors angepasst wird. Hierzu wird meist eine bilineare Unterabtastung um den Faktor 2 vorgenommen. Allgemeiner betrachtet werden nxn Werte zu einem Wert zusammengefasst.
Wie zu Beginn des Kapitels erwähnt, haben CNNs gegenüber den MLPs den Vorteil, dass sie nahezu beliebig hochskaliert werden können und somit gut geeignet für große Datenmengen sind. Dies liegt vor allem daran, dass die Neuronen nur lokal verbunden sind und sich somit das Prinzip der geteilten Gewichte zu Nutze gemacht werden kann. Ein weiterer Vorteil von CNNs, der vor allem in der Bildverarbeitung genutzt wird, ist das sie translationsinvariant sind.

2.4. Training Faltungsnetze

Laura

Meistens werden CNNs mittels der *back-propagation* Methode trainiert. Bei dieser überwachten Lernmethode bedarf es einer großen Menge an vorher klassifizierten Eingabematerialien [13].
In den Faltungsschichten kann der Fehler der vorangegangen Schicht nach Formel 4 berechnet werden. Dabei steht E für den Fehler in der jeweiligen Schicht l gemacht wird. Während x^l für die Eingabe in die Schicht steht, bezeichnet y^l die Ausgabe der entsprechenden Schicht. Die Größe der Eingabe wird der Einfachheit halber als quadratisch, also mxm-groß angenommen. Die Gewichtung wird mit w bezeichnet. Um die Formel in der Realität anzuwenden, muss die linke und obere Grenze de Eingabeinhaltes, z.B. eines Bildes, mit Nullen ergänzt werden. Ansonsten wäre es nicht möglich den Fehler für Pixel zu berechnen, welche näher als m an den entsprechenden Rändern liegen.

$$\begin{aligned}\frac{\delta E}{\delta y_{ij}^{l-1}} &= \sum_{a=0}^{m-1}\sum_{b=0}^{m-1}\frac{\delta E}{\delta x_{(i-a)(j-b)}^{l}}\frac{\delta x_{(i-a)(j-b)}^{l}}{\delta y_{ij}^{l-1}} \\ &= \sum_{a=0}^{m-1}\sum_{b=0}^{m-1}\frac{\delta E}{\delta x_{(i-a)(j-b)}^{l}} w_{ab}\end{aligned} \tag{4}$$

Die Schichten, in denen die Unterabtastung stattfindet, leisten kaum Beitrag zum eigentlichen Lernprozess des CNNs. Hier wird das Problem allerdings reduziert, da nxn Werte in einem einzigen resultieren.
Weil alle Gewichtungen w mittels des *back-propagation* Algorithmus während des Trainings angepasst werden, können CNNs laut LeCun als Erzeuger ihrer eigenen Merkmalextraktion gesehen werden [12].

3. Konzeption

Vera

Vor der eigentlichen Produktion von Videomaterial muss das Projekt zunächst konzipiert und detailliert geplant werden. Dies bezieht sich vor allem auf die kreativen Prozesse des Drehbuchschreibens und die darauffolgende gesamte Projektplanung und -vorbereitung. Naturgemäß sind NN weniger sinnvoll für die Planung und das Management von Projekten. Doch in den letzten Jahren wurde versucht mit Hilfe von NN kreative Prozesse umzusetzen, wie sie für das Schreiben von Drehbüchern benötigt werden. Es gibt bereits erste Versuche mit Hilfe von NN automatisch sinnvolle Texte und Dialoge zu erstellen, die auf bekannten Texten und Storylines basieren [24]. Dieses Verfahren wird in der Literatur auch *Deep Writing* genannt. Es können auch Romane, Dialoge oder Songtexte automatisch mit einem LSTM Recurrent NN erstellt werden [4].

3.1. Aktueller Stand

Vera

Ein solches LSTM Recurrent NN wurde mit allen bekannten Episoden der Serie *Silicon Valley* trainiert [5]. Zu Beginn des Schreibprozesses wird ein beliebiges Wort genutzt um den ersten Satz zu initialisieren. Das NN sucht im Anschluss das Wort, welches am häufigsten nach dem Startwort in den Trainingsdaten genannt wurde. Mit diesem Wortpaar wird nach dem selben Prinzip das dritte Wort des Textes ermittelt. Dieser Prozess wird solange wiederholt bis der generierte Text die gewünschte Länge erreicht hat [4]. Die generierten grammatikalisch korrekten Sätze beziehungsweise Dialoge ergeben keine sinnvollen Zusammenhänge und folgen keiner ersichtlichen Storyline.

Der Film *Sunspring* [17] wurde nach einem ähnlichen Prinzip erstellt und im Anschluss von einem professionellen Filmteam realisiert. Der größte Unterschied zu [5] ist, dass das NN nicht nur Wörter unterscheidet. Stattdessen wird zunächst alles in Buchstaben zerlegt und dann in neue Wörter und Sätze zusammengesetzt. Das verwendete NN nannte sich selber *Benjamin* und wurde mit einer großen Anzahl an Drehbüchern von Science Fiction Filmen aus den 80er und 90er Jahren trainiert. Heraus kam ein Drehbuch mit ähnlich grammatikalisch korrekten Sätzen, die aber häufig inhaltlich keinen Sinn ergaben. Ein anderes Problem war der Umgang mit Namen, da diese sprachlich anders behandelt werden. Aus diesem Grund mussten alle Charaktere nur mit einzelnen Buchstaben benannt werden. Dies hatte zur Folge, dass das NN zwei Charaktere mit der selben Bezeichnung betitelt wurden und nachträglich umbenannt wurden.

Aus den mangelnden Zusammenhängen lässt sich ableiten, dass derzeit NN keine kreativen Prozesse simulieren können. Selbst mit einer sehr großen Anzahl von Trainings-

daten konnten keine kohärenten Dialoge generiert werden und keine konstante Storyline verfolgt werden. Weiterführend, sind die NN nicht in der Lage neue Charaktere oder Geschichten zu erfinden, sonder kombiniert lediglich bekannte wörtliche Zusammenhänge neu. Somit gibt es zur Zeit keinen brauchbaren Ansatz, welcher den Aufwand des Drehbuchschreibens minimieren könnte.

4. Produktion

Laura

Die Produktion ist von allen drei besprochenen Kategorien diejenige, die bisher am wenigsten automatisiert durchgeführt werden kann. Moderne Kameras- und Tonaufnahmesysteme bieten zwar eine einfachere Handhabung und Fehlerkontrolle, als es noch bei der analogen Aufzeichnung der Fall war, dennoch braucht eine Filmproduktion viele menschliche Arbeitsstunden.
Im Folgenden soll geprüft werden, ob es heutzutage realisierbare Austomatiersungsansätze gibt. Dabei wird oberflächlich auf die Generierung von Musik eingegangen, die später in der Postproduktion unter die einzelnen Szenen gelegt werden kann. Daran schließt sich eine ausführlichere Betrachtung eines Ansatzes zur automatisierten Generierung von Szenen Dynamiken an.

4.1. Aktueller Stand Musikgenerierung

Laura

Es gibt verschiedene Ansätze NNs zu nutzen, um Musik automatisiert generieren zu lassen. Im Folgenden werden drei Ansätze kurz vorgestellt, welche alle rückgekoppelte Neuronale Netze (RNN) benutzen.
An der *University of Washington* ist im Rahmen einer Projektarbeit ein Musik Generator namens *Algo Rythm* entstanden [25]. Für die Umsetzung haben die Studierenden um Timmerman RNNs geeignet trainiert. Der Quellcode kann auf Github [26] eingesehen werden.
Ein weiterer Ansatz, der in [6] beschrieben wird, arbeitet ebenfalls mit einem RNN, welches mit einem Autokorrelation-basierte Prädiktor kombiniert wird. Dabei soll die Struktur von Musikstücken erlernt werden, indem zunächst die folgende Note einer Tonreihenfolge vorausgesagt werden soll.
Der letzte Ansatz benutzt ebenfalls RNNs um auf Grundlage einer Notensequenz eine Musikstück zu komponieren [3]. Dabei wird die Eingabesequenz zunächst interpretiert. Anschließend sorgen zwei RNN basierte Algorithmen für die Produktion von sowohl Rhythmus, als auch die Vorhersage der nächsten Note.

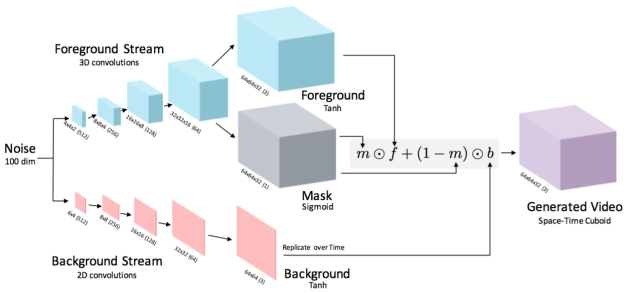

Abbildung 4: Aufbau Generator nach [29].

Alle drei Ansätze weisen hohes Potential auf, wenn es darum geht Musikstücke automatisiert zu generieren. Über das Genre des zugeführten Musikmaterials lässt sich die gewünschte Ausgabe begrenzt steuern. Es wäre durchaus vorstellbar, so generierte Musik für eine Filmproduktion zu benutzen.

4.2. Szenendynamik nach Vondrick et al.

Laura

Mit dem Ansatz von Vondrick et al. [29] können aus Einzelbildern ganze Szenen Dynamiken erstellt werden, welche zum einen für die Klassifizierung und zum anderen für die Vorhersage von Bewegung genutzt werden kann. Das Resultat sind kleine Videos mit einer Auflösung von 64x64 Pixeln und 32 Einzelbildern.
Die Grundlage für dieses Verfahren bilden zwei CNNs, die als *Generative Adversarial Networks* (GAN) [8] fungieren. Die beiden Netze können als Gegenspieler angesehen werden. Während der Generator G für die Generierung von einer Videosequenz auf Grundlage eines Standbildes zuständig ist, soll der Diskriminator D zwischen den so erzeugten und realen Videosequenzen differenzieren können. Um dies zu erreichen werden beide GANs, wie in Formel 1 des Papers [29] zu sehen gegeneinander trainiert. Das Minimierungs- bzw. Maximierungsproblem wird hierzu mittels Gradientenverfahren gelöst.
Das Training der beiden Netze erfolgt unüberwacht. Da ähnliche Objekte in der Regel auch ähnliche Bewegungsmuster aufweisen, lassen die Autoren das zur Verfügung stehende Videomaterial in Kategorien (Strand, Babys, Golf und Züge) einteilen. Zudem sorgt ein Stabilisierungsalgorithmus dafür, dass bei dem gesamten Testmaterial von einer statischen Kamera ausgegangen werden kann und die zu erlernende Szenen Dynamik nicht durch eine Bewegung der Kamera verfälscht wird. Für jede dieser Kategorien wird dann ein eigenes Pärchen bestehend aus Generator und Diskriminator trainiert.
Bevor die beiden CNNs G und D auf die beschriebene Art trainiert werden können, müssen sie erst einmal ihren Aufgaben entsprechend aufgebaut werden.
Den Aufbau des Generators, welchen Vondrick et al. genutzt haben, ist in Abbildung 4 zu sehen. Im Gegensatz

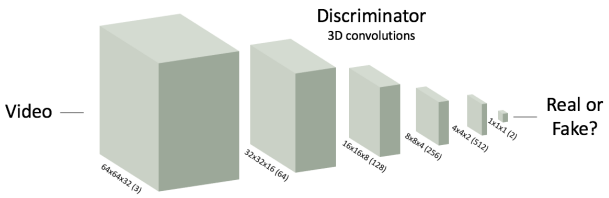

Abbildung 5: Aufbau Diskriminator nach [29].

zu dem in Kapitel 2.3 beschriebenen, gewöhnlichen CNN, wird hierbei keine Unterabtastung vorgenommen, sondern die Menge der Daten von Schicht zu Schicht erweitert. Zudem werden die Eingabedaten für die Verarbeitung in zwei Datenströme aufgeteilt. Während in dem einen Strom der statische Hintergrund entsprechend vergrößert wird, wird in dem anderen die Bewegung des sich im Vordergrund befindlichen Objektes vorhergesagt. Beide Teile werden dann am Ausgang nach Formel 5 zusammengefasst.

$$G(z) = m(z) \odot f(z) + (1 - m(z)) \odot b(z) \tag{5}$$

Dafür ist die binäre Maske m so gewählt, dass sie überall den Wert eins hat, wo der Vordergrund übernommen werden soll und ansonsten nur Nullen enthält.
Der Diskriminator hat einen wesentlich simpleren Aufbau, der in Abbildung 5 zu sehen ist. Dabei muss seine Eingabeschicht eben so groß sein, wie die Ausgabeschicht des Generators.
Die Bedeutung für so erzeugte Bewegtbildsequenzen für die Automatisierung der Produktion von Filmen, wird im nächsten Kapitel diskutiert. Die Resultate können sich auf der Internetseite der Autoren eingesehen werden [28].

4.3. Bewertung des Ansatzes von Vondrick

Laura

Das in Kapitel 4.2 beschriebene Verfahren schafft es automatisiert aus einem Einzelbild eine kurze und niedrig aufgelöste Bewegtbildsequenz zu erstellen. Hierbei wir vor allem die Szenen Dynamik in den meisten Fällen annähernd realistisch vorhergesagt, wobei die Modellierung des sich bewegenden Bildinhaltes eher mangelhaft ist. Trotz des hohen Bedarfs an Trainingsmaterial und den damit verbundenen Zeitaufwand, kann das System ohne menschliche Hilfe eigenständig Kategorisieren, Trainieren und letztendlich Szenen Dynamiken generieren. Gegen einen Einsatz in der Videoproduktion sprechen die geringe Auflösung und Zeitspanne die bisher generiert werden kann. Zudem kann in einer Szene nicht immer nur von einem sich bewegenden Objekt ausgegangen werden. Und ein trainieren von Netzen für alle denkbaren Kategorien ist sehr aufwendig. Dennoch könnte der Ansatz als Grundlage für einen ausgereifteren Algorithmus dienen, der beispielsweise die Anweisungen des Drehbuchs mit einbezieht und somit eine Generierung von planbareren Szenen ermöglicht.
Der jetzige Stand des Algorithmus könnte beispielsweise für die Generierung von GIFs oder Simulationen genutzt werden. Zudem könnte der Ansatz, mit etwas Anpassung, anstatt Szenen Dynamiken in Form von Videos darzustellen, Bewegungsvektoren aus einem Standbild vorhersagen und somit in der Videocodierung Einsatz finden.

5. Postproduktion

Vera

In diesem Kapitel wird die Verwendung von NN in der Videopostproduktion beschrieben. Gerade in diesem Produktionsabschnitt finden NN eine breites Anwendungsfeld, da gerade die CNN im Bereich der Bildverarbeitung in den letzten Jahre sehr erfolgreich eingesetzt werden konnten. Vor allem für die Bildklassifikation und Stilsynthese erweisen sich CNN als äußerst sinnvoll. Stilsynthese-Verfahren extrahieren idealerweise einen Bildstil und übertragen ihn sinngemäß auf ein weiteres unabhängiges Bild ohne den Bildinhalt zu modifizieren. Weiterführend stellte sich heraus, dass CNN auch für *Text-to-Speech* Verfahren erfolgreich eingesetzt werden können.

Im nächsten Abschnitt werden zunächst die interessantesten auf NN-basierenden Bildverabeitungs- und Text-to-Speech Ansätze in der Postproduktion vorgestellt und bewertet. Auf zwei Verfahren zur Stilsynthese wird in den darauffolgenden Anschnitten näher eingegangen inklusive einer ausführliche Analyse über die Verwendung der Verfahren in der professionellen Videoproduktion. Der zweite Ansatz ist eine Weiterentwicklung des ersten und somit bauen die Algorithmen aufeinander auf. Zuletzt wird ein Ausblick auf Einsatzmöglichkeiten gegeben. .

5.1. Aktueller Stand

Vera

Für einige Produktionen, wie Dokumentationen oder Nachrichtensendungen, ist erforderlich sogenannte *Voice-Over* auf das Videomaterial zu legen. Diese müssen vorab in einem Tonstudio mit einem Sprecher produziert werden. Dies kann zukünftig durch *Text-to-Speech-* Verfahren ersetzt werden. Ein vielsagender Ansatz ist *WaveNet*, welches im Gegensatz zu den meisten anderen Ansätzen auf einem CNN basiert [27]. Dieses CNN ist wie ein pixelbasiertes CNN aufgebaut. In das Netz werden textbasierte Eingangsdaten gegeben, welche vom CNN in Audiodaten gewandelt werden. Diese Audiodaten klingen mit Vergleich mit anderen bekannten Verfahren nahezu fehlerfrei und natürlich. Die generierten Audiodaten werden fast als

menschliche Stimme wahrgenommen. Doch verfahrensbezogen können nur Stimmen imitiert werden von denen vorab Stimmproben in das NN geben wurden.
Das *Deep Voice* Verfahren [1] baut auf dem *WaveNet* Prinzip auf und entwickelt es weiter. Während *WaveNet* mehrere Minuten Rechenzeit benötigt um eine Sekunde an Audiodaten zu generieren, ist *Deep Voice* Realtime-fähig. Ein weiterer Vorteil ist, dass *Deep Voice* ein eigenständiges System ist, welches es für die Nutzung in der Videoproduktion interessanter macht. Einen Vergleich bezüglich der Klangqualitätsunterschiede zwischen den beiden Verfahren gibt es leider nicht.

Nachdem alle notwendigen Daten produziert wurden ist es der erste Schritt die Daten zu sichten, kennzeichnen und in entsprechende Mediendatenbanken für die weitere Produktion einzupflegen. Alle drei Schritte können mit Hilfe von NN erheblich automatisiert werden. Vorallem in der Objekterkennung und -klassifikation konnten mit CNN in den letzten Jahren eine sehr geringe Fehlerquote und hohe Robustheit erreicht werden. Aus diesem Grund werden sie flächendeckend in der Praxis verwendet. Eine bekannte Bibliothek ist *Clarifai* [19], welche konfigurierte CNN anbietet um optimierte Bild- oder Videodatenbanken anzulegen und zu verwalten. Weitere aktuelle Veröffentlichungen [31][30][11] konzentrieren sich auf die Verbesserung der Leistungsfähig um auch große Videodaten robust verarbeiten zu können. Dies ist für die große Menge an Videodaten einer Filmproduktion unbedingt notwendig. Zum jetzigen Zeitpunkt erfordert die Erkennung dieser Mengen noch sehr große Rechenkapazität und ist somit entsprechen kostenintensiv während sie dennoch Personalaufwand einspart.

Neben der verbreiteten Klassifikation können CNN auch Bilddaten generieren oder synthetisieren. Zwei vielversprechende Ansätze sind das Deep Dream [15] und das Deep Style Verfahren [7]. Ein großer Vorteil beider CNN-basierten Verfahren gegenüber den bekannten Synthese-Verfahren ist dass sie die gesamte Semantik eines Bildes vom Inhalt eindeutig zu trennen und mit einem unabhängigem Bild verschmelzen können. Bildlich gesprochen bedeutet dies, dass die Verfahren den gesamten Stil eines Gemäldes mit der Farbwelt, dem Pinselstrich und dem Grad der Abstraktheit erfassen und übertragen können während die bekannten Verfahren nur einzelne Ausschnitte der Gemäldemerkmale erkennen. Hierfür ist es notwendig die höherwertigen Texturmerkmale des stilgebenden Bildes zu extrahieren und ein Zielbild so zu transformieren, dass es der extrahierten Semantik gleicht ohne dessen Bildinhalt zu verändern [14]. Dabei wird strikt zwischen den stilgebenden und inhaltsbezogenen Merkmalen unterschieden um eine Modifikation des Inhalts verhindern.

5.2. Faltungsnetze zur Stilsynthese

Vera

Um die folgenden Algorithmen in Abschnitt 5.3 und 5.4 besser verstehen zu können muss zunächst näher auf die Prozesse innerhalb eines CNN eingegangen werden. In Abbildung 3 wird deutlich, dass mit jeder weiteren Schicht das Bild stärker unterabgetastet wird und die Größe der Merkmalsvektoren stetig steigt. Mit jeder weiteren Schicht wird ein größer Ausschnitt des Bildes erfasst und dessen Merkmale extrahiert. Die Merkmalsvektoren werden größer, da ein größerer Bereich mehr Merkmale enthält und diese folglich immer komplexer werden.
Im oberen Bereich der Abbildung 6 wird veranschaulicht, dass mit steigender Schichtanzahl die stilgebenden Merkmale immer höherwertig werden und sich der Semantik des Stilbildes annähern. Während in der ersten Schicht die Stilmerkmale sehr klein sind und nicht der Bildsemantik ähneln, können in den höherwertigen Schichten Stilelemente aus dem gesamten Bild gefunden werden, wie zum Beispiel die gelben Kreise und die Strudel im Himmel. Im untereren Abbildungsbereich sind die inhaltsbezogenen Merkmale dargestellt. Hier ist die Unterabtastung klar zu erkennen und mit jeder Schicht wird die Darstellung der Merkmalsvektoren immer gröber. Trotzdem bleibt es stets erkennbar welches Gebäude abgebildet wird. Die Inhaltsinformationen bleiben demzufolge erhalten.

Gleichzeitig veranschaulicht die Abbildung 6, dass die Inhalte der verborgenen Schichten eines trainierten CNN visualisiert werden können. Um dieses Wissen zu erlangen, wurden der übliche Prozess zur Objekterkennung invertiert. Das trainierte CNN wurde mit einem Eingangsbild initialisiert, welches ausschließlich zufälliges Rauschen enthält. Dieses Eingangsbild solange durch das CNN iteriert bis das gewünschte Objekt klassifiziert werden konnte. In diesem Bild konnten teilweise eindeutige Merkmale dieses Zielobjektes ausgemacht werden. An dieser Stelle sei hinzugefügt, dass das CNN ein Objekt nicht immer präzise bestimmen kann, da die Klassifikation stark von den Trainingsdaten abhängt. Zum Beispiel bei der Bestimmung von Hanteln kann es dazu kommen, dass auch Arme zum Merkmalsvektor hinzugefügt werden, da diese überdurchschnittlich häufig auf den Trainingsdaten mit den Hanteln zu sehen waren. Trotzdem führt dieser invertierte Prozess zu der Erkenntnis, dass CNN auch für die Generierung von Bilddaten genutzt werden und nicht ausschließlich für die Klassifizierung [15].
Ein anderer Ansatz lässt das CNN selbst entscheiden, welche Merkmale verstärkt werden. Das beliebige Eingangsbild wird durch das CNN propagiert und die Merkmalsvektoren der einzelnen Schichten untersucht. In den niedrigeren Schichten können nur primitive Texturmerkmale aus-

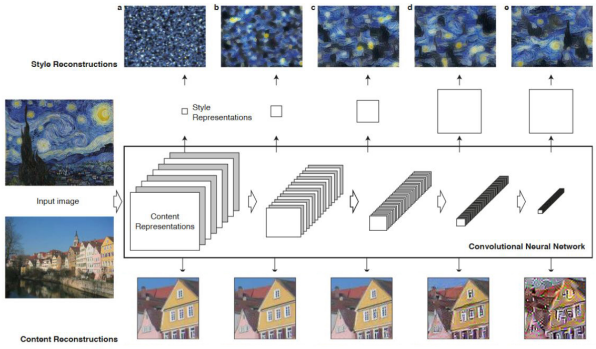

Abbildung 6: Merkmalsextrahierung in einem Faltungsnetz[7].

gemacht werden, wie Linien oder Rechtecke, während der Bildinhalt unverändert bleibt. Mit steigender Schichtanzahl werden die verstärkten Muster immer komplexer und der Bildinhalt wird teilweise modifiziert. So wird deutlich, dass CNN in den höheren Schichten die benötigten Informationen der Semantik enthalten und diese eindeutig vom Bildinhalt separiert werden können.

5.3. Deep Dream

Vera

Der folgende Abschnitt stellt das Stilsynthese-Verfahren *Deep Dream* [15] vor, welches ein Nebenprodukt des Versuches den Inhalt eines CNN zu visualisieren ist. *Deep Dream* basiert auf einem *GoogLeNet* CNN mit 22 Schichten und fünf Unterabtastungsschichten [23]. Die Schichten haben eine besondere Architektur, welche *Inception Layer* genannt wird. Mit Hilfe dieser Architektur können auch effizient große Merkmalsvektoren verarbeitet werden. Dieses *GoogLeNet* wird im ersten Schritt mit dem *ImageNet* Trainingsdatenset trainiert.
An dieser Stelle sei zum besseren Verständnis hinzugefügt, dass ein wichtiger Unterschied zu den bereits bekannten Faltungsnetzmodellen aus Abschnitt 2.3 besteht. Die Schichten zur eigentlichen Klassifizierung sind für die Stilsynthese nicht relevant sind. Interessant sind die Faltungsschichten, welche die benötigten Merkmalsvektoren extrahierten und speichern.
Zunächst wird das stilgebende Bild, im folgenden Style Source (SS) genannt, bis zu einer beliebigen Schicht vorwärtspropagiert und der entsprechende Merkmalsvektor als $\vec{f_{GS}}$ gespeichert. Dieser enthält die Merkmale der gewünschten Semantik und steuert die Transformation in Richtung des gewünschten Stils. Würde $\vec{f_{GS}}$ entfallen nähert das CNN die Transformation an die Trainingsdaten an. Bezogen auf Abschnitt5.2 empfiehlt es sich höherwertige Schichten zu wählen um eine zufriedenstellendes Synthese zu generieren.

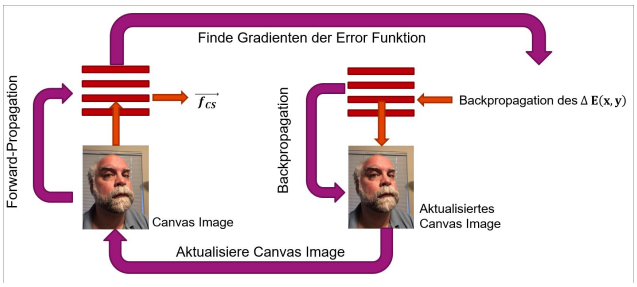

Abbildung 7: Ablaufdiagramm des Deep Dream Verfahrens.

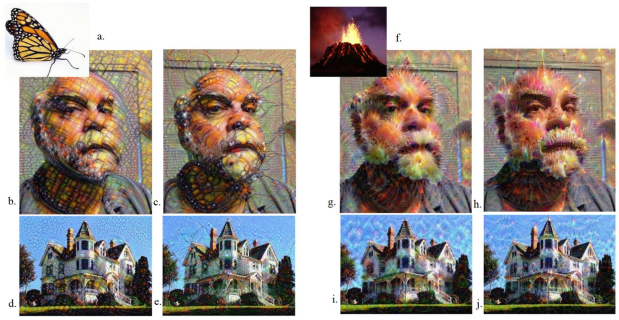

Abbildung 8: Resultate des Deep Dream Verfahrens [14].

Den weiteren Transformationsprozesses stellt Abbildung 7 schematisch dar. Das Zielbild, im folgenden Canvas Image (CI) genannt, wird iterativ transformiert. Zu Beginn einer Iteration wird es bis zur selben Schicht wie $\vec{f_{CS}}$ vorwärtspropagiert und als $\vec{f_{CS}}$ gespeichert. Im Anschluss wird der Gradient der Fehlerfunktion $\Delta E(x, y)$ aus Gleichung 6 berechnet. Diese ist als das Skalarprodukt aus $\vec{f_{CS}}$ und $\vec{f_{CS}}$ definiert, welches maximal ist, wenn beide Vektoren in die selbe Richtung zeigen.

$$E(x, y) = \vec{f_{GS}} * \vec{f_{CS}} \tag{6}$$

Der resultierende $\Delta E(x, y)$ wird zurückpropagiert und die Gewichtungen innerhalb des Netzes in Richtung des SS korrigiert. Das daraus entstehende Bild wird als neues CI aktualisiert und für die nächste Iteration verwendet. Dieser Ablauf wird solange wiederholt bis sich $\vec{f_{CS}}$ möglichst dem Ausgangsbild $\vec{f_{GS}}$ angenähert hat und der Fehler minimal ist. In diesem Fall ist $E(x, y)$ maximal und der gesamte Prozess wird als Maximierung von Gleichung 6 interpretiert.

Einige Ergebnisse des *Deep Dream* Verfahrens sind in Abbildung 8 dargestellt. Die kleinen Bilder in der oberen Ecke sind die verwendeten SS. Auf der linken Seite eines Blockes sind jeweils die Ergebnisse mit einer niedrigen Schicht und rechts solche einer höheren Schicht dargestellt. Auf den ersten Blick fällt direkt auf das die linken

Bilder stärker die kleinen Merkmales übernommen haben während rechts eindeutig größere Texturen übernommen wurde, wie zum Beispiel die Form des Schmetterlings oder Vulkans am Hals. Gleichzeitig fällt rechts auf, dass markante Bildpunkte, wie die Augen oder die Nase, stark modifiziert wurden und teilweise sogar deren Form verändert wurde. Somit können teilweise Inhalte des ursprünglichen Bildes verloren gehen. Die Nase wurde in c einer Hundeschnauze angenähert, da die Trainingsdaten von *ImageNet* einen Überschuss an Hundemotiven enthält. Besonders deutlich wird dies, wenn dein SS verwendet wird[14]. Andere Ergebnisse zeigten auch deutlich, dass durchaus auch die Merkmale der Trainingsdaten gegenüber denen des SS überwiegen können.
Es ist demzufolge für den Nutzer nicht möglich die Endergebnisse zu kontrollieren oder in den Prozess einzugreifen. Die einzige Möglichkeit der Kontrolle bietet die Wahl des SS und der Trainingsdaten, welche einen nicht vorhersehbaren Einfluss auf das Endergebnis haben. Daher kann von keiner strikten Trennung von Semantik und Inhalt ausgegangen werden.
Weiter kann nicht gewährleistet werden die Ergebnisse zu reproduzieren und ähnliche Bilder auf kohärente Weise zu transformierten, da die Gewichte des CNN stetig während einer Iteration modifiziert werden. Diese Reproduzierbarkeit und Kohärenz ist ein wichtiges Kriterium für die Videoproduktion, da eine Sequenz von ähnlichen Bilder entsprechen gleichförmig transformiert werden muss. Aus diesem Grund ist dieses Verfahren mehr als neue Kunstform und interessante Spielwiese um die Funktionalität von CNN besser zu verstehen. Für die Videoproduktion ist sie gänzlich ungeeignet.

5.4. Deep Style

Gerade die mangelnde Einflussnahme und die fehlende Reproduzierbarkeit des Nutzers beim *Deep Dream* Verfahren [15] verlangt nach einer Weiterentwicklung. Diese ist durch das *Deep Style* Verfahren [7] gegeben, welche ein ähnliches Schema verfolgt. Der große Unterschied ist die strikte Trennung und Berücksichtigung von stilgebenden und inhaltsbezogen Informationen, einer erweiterten Fehlerfunktion und der Initialisierung durch ein Bild mit zufälligem Rauschen.
Zusätzlich wird eine andere CNN-Architektur verwendet. Es wird ein trainiertes VGG Faltungsnetz [20] mit 16-19 Schichten benutzt und dessen Schichten in erster Linie aus 3×3 Faltungskernen besteht. Dieses VGG wurde ebenfalls mit den *ImageNet* Trainingsdatenset trainiert.

Nach dem Training wird zunächst ein SS vorwärtspropagiert bis zur höchsten Schicht. Für jede Schicht wird im Gegensatz zu *Deep Dream* der entsprechende Merkmalsvektor $\vec{GS}$ abgespeichert. Diese Vektoren beinhalten alle Stilmerkmale auf allen Ebenen, welche für die spätere Berechnung des Fehlergradienten benötigt werden. Im Anschluss wird das Zielbild bis zur letzten Schicht propagiert, welches im Folgenden Content Source Image (CS) genannt wird. An dieser Stelle wird nur der Merkmalsvektor $\vec{GC}$ der letzten Schicht verwendet.

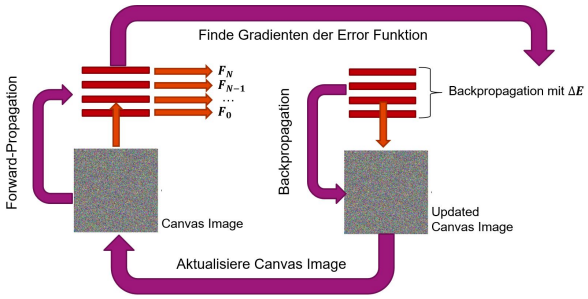

Abbildung 9: Ablaufdiagramm des Deep Style Verfahrens.

Abbildung 9 zeigt den schematischen Ablauf nach der erfolgreichen Initialisierung. Wie bereits erwähnt, beinhaltet das Canvas Image (CI) zufälliges Rauschen, welches während jeder Iteration in die Richtung von $\vec{GC}$ und $\vec{GS}$ transformiert wird. Das CI wird ebenfalls vorwärtspropagiert und mit den resultierenden Merkmalsvektoren f aus jeder Schicht die Korrelation G zwischen den Merkmalsvektoren innerhalb einer Schicht ermittelt, wie in Gleichung 7. In dieser Gleichung ist l die Anzahl Schichten im VGG und sie ist der erste Schritt zur Fehlerberechnung der Stilmerkmale in Gleichung 11.

$$G_{ij}^{l} = \frac{1}{2} \sum_{k=0} (\vec{f_{ik}} - \vec{f_{jk}})^2 \tag{7}$$

Im Anschluss wird mit diesen Korrelationen G_{ij}^{l} jeweils pro Schicht die quadrierte Fehlerfunktion F_l aus Gleichung 8 berechnet. In diesem Fall bezeichnet N die Anzahl der Merkmalsvektoren in einer Schicht und M deren Größe. Für den gesamten Fehler der Stil F_{style} werden alle F_l wie in Gleichung 11 gewichtet aufsummiert.

$$F_l = \frac{1}{2N_l^2 M_l^2} \sum_{k=0}^{N} (G_{ij}^{l} - \vec{GS_{ij}^{l}})^2 \tag{8}$$

$$F_{style}(SS, CI) = \sum_{k=0} w_l F_k \tag{9}$$

Neben F_{style} wird auch $F_{content}$ berechnet, doch dieses ist weniger komplex. $F_{content}$ ist als simple quadrierte Fehlerfunktion aus den Merkmalsvektoren von GC und denen des f-Vektors der letzten Schicht definiert.

$$F_{content}(\vec{f_{GC}}, \vec{f}) = \frac{1}{2} \sum_{i,j} (\vec{GC_{ij}} - \vec{f_{ij}})^2 \tag{10}$$

Die gesamte Fehlerfunktion $F(SS, CS, CI)$ ergibt sich aus der gewichteten Summe von $F_{content}$ und F_{style}. Die

Abbildung 10: Resultate des Deep Style Verfahrens [14].

beiden Gewichte α und β ermöglichen dem Nutzer das Verhältnis zwischen Stil und Inhalt zu modifizieren und somit das Endergebnis seinen Vorstellungen anzupassen.

$$F(SS, CS, CI) = \alpha F_{con} + \beta F_{style} \tag{11}$$

Von $F(SS, CS, CI)$ wird, wie beim *Deep Dream* Verfahren, auch der Gradient bestimmt um eine Rückwärtspropagation durchzuführen. Das weitere Ablaufschema gleicht dem des *Deep Dream* und es wird solange iteriert bis sich die Merkmalsvektoren von CI an die Initalisierungsvektoren von SS und CS nahezu ähneln.

5.5. Ausblick

In der Zukunft kann das *Deep Style* Verfahren für verschiedene Anwendungen sinnvoll sein. Zunächst sei hier die qualitativ hochwertige und automatische Coloration und Texturierung von animierten Filmen ohne die aufwendige manuelle Texturierung von einzelnen 3D Objekten.
Die Rekonstruktion und Restauration von verlorenen Filmmaterial, wie zum Beispiel *Metropolis* von Fritz Lang ist vorstellbar. Da verlorene Szenen nachgestellt werden könnten und an den Stil der verbliebenen Szenen angepasst werden.
Auch eine individuelle Anpassung an den Geschmack und Stimmung des Nutzers von Filmen ist für *Video-on-Demand* denkbar. Gerade die Farbgebung beeinflusst die Wahrnehmung des Zuschauers. So könnte zum Beispiel die Bedrohlichkeit eines Thrillers verschärft oder minimiert werden. Dasselbe gilt auch für Computerspiele, welche unter Umständen an Geschmack von jüngerem oder erwachsenem Publikum angepasst werden könnte.

6. Fazit

Vera

Literatur

[1] S. Ö. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, J. Raiman, S. Sengupta, and M. Shoeybi. Deep voice: Real-time neural text-to-speech. *CoRR*, abs/1702.07825, 2017.

[2] H. Braun. *Neuronale Netze Optimierung durch Lernen und Evolution*. Springer, 1997.

[3] C. Browne. System and method for automatic music generation using a neural network architecture, Oct. 2 2001. US Patent 6,297,439.

[4] M. Deutsch. How to write with artificial intelligence. https://medium.com/deep-writing/how-to-write-with-artificial-intelligence-45747ed073c. Abgerufen: 19.08.2016.

[5] M. Deutsch. Silicon valley: A new episode written by ai. https://medium.com/deep-writing/silicon-valley-a-new-episode-written-by-ai-a8f832645bc2. Abgerufen: 19.08.2016.

[6] D. Eck and J. Lapalme. Learning musical structure directly from sequences of music. (1300), 2008.

[7] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[9] S. Haykin. *Neural Networks A Comprehensive Foundation*. Macmillan College Publishing Company, 1994.

[10] T. Isokawa, H. Nishimura, and N. Matsui. Quaternionic multilayer perceptron with local analyticity. *Information*, 3(4):756, 2012.

[11] D. Kang, J. Emmons, F. Abuzaid, P. Bailis, and M. Zaharia. Optimizing deep cnn-based queries over video streams at scale. *CoRR*, abs/1703.02529, 2017.

[12] Y. LeCun and Y. Bengio. The handbook of brain theory and neural networks. chapter Convolutional Networks for Images, Speech, and Time Series, pages 255–258. MIT Press, Cambridge, MA, USA, 1998.

[13] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 253–256, May 2010.

[14] G. McCaig, S. DiPaola, and L. Gabora. Deep convolutional networks as models of generalization and blending within visual creativity. *CoRR*, abs/1610.02478, 2016.

[15] A. Mordvintsev. Inceptionism: Going deeper into neural networks. https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html. Abgerufen: 16.08.2016.

[16] D. Nauck, F. Klawonn, and R. Kruse. *Neuronale Netze und Fuzzy Systeme*. Vieweg, 1994.

[17] A. Newitz. Movie written by algorithm turns out to be hilarious and intense. https://arstechnica.com/gaming/2016/06/an-ai-wrote-this-movie-and-its-strangely-moving/. Abgerufen: 19.08.2016.

[18] M. Osadchy, Y. LeCun, and M. Miller. Synergistic face detection and pose estimation with energy-based models. *Journal of Machine Learning Research*, 8:1197–1215, 2007.

[19] T. Simonite. A startup's neural network can understand video. https://www.technologyreview.com/s/534631/a-startups-neural-network-can-understand-video/. Abgerufen: 16.08.2016.

[20] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[21] J. Stanley and E. Bate. *Neuronale Netze Computersimulation biologischer Intelligenz*. Systhema Verlag GmbH, 1991.

[22] statista. Anzahl der aktiven film- und fernsehproduktionsfirmen in deutschland in den jahren 1998 bis 2014. https://de.statista.com/statistik/daten/studie/243238/umfrage/anzahl-der-film-und-fernsehproduktionsfirmen-in-deutschland/. Abgerufen: 10.08.2016.

[23] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[24] M. Thoma. Creativity in machine learning. *CoRR*, abs/1601.03642, 2016.

[25] G. Timmermann. Algo rhythm: Music composition using neural networks. https://medium.com/@granttimmerman/algo-rhythm-music-composition-using-neural-networks-f89897ff2df7. Abgerufen: 18.08.2016.

[26] G. Timmermann. Algorithmic music composition using artificial neural nets. https://github.com/grant/algo-rhythm. Abgerufen: 18.08.2016.

[27] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR*, abs/1609.03499, 2016.

[28] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating Videos with Scene Dynamics. `http://carlvondrick.com/tinyvideo/`. Abgerufen: 17.08.2016.

[29] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating Videos with Scene Dynamics. *CoRR*, abs/1609.02612, 2016.

[30] H. Ye, Z. Wu, R.-W. Zhao, X. Wang, Y.-G. Jiang, and X. Xue. Evaluating two-stream cnn for video classification. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, ICMR '15, pages 435–442, New York, NY, USA, 2015. ACM.

[31] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.