Machine Learning: Generative Adversarial Networks

Seminararbeit

Ausgewählte Themen der Informatik

des Studienganges Angewandte Informatik an der

Dualen Hochschule Baden-Württemberg Mosbach

von

Mirco Heck & Johannes Brandau

26. November 2023

Bearbeitungszeitraum	Studienjahr 2023/24
Matrikelnummern , Kurs	7306389 & 6160077, MOS-TINF21A

Inhaltsverzeichnis

Abkürzungsverzeichnis

KNN Künstliches Neuronales Netz

GAN Generative Adversarial Network

CNN Convolutional Neural Network

DCGAN Deep Convolutional Generative Adversarial Network

RAN Recurrent Adversarial Network

Abbildungsverzeichnis

1 Einleitung

Generative Adversarial Networks (GANs) haben in den letzten Jahren erhebliche Aufmerksamkeit in der Forschungsgemeinschaft auf sich gezogen. Sie stellen eine neue Methode dar, um generative Modelle zu trainieren und haben eine Vielzahl von Anwendungen in Bereichen wie Bildsynthese, Super-Resolution und Style Transfer.

Das Ziel dieser Arbeit ist es, einen umfassenden Überblick über GANs zu geben, ihre Funktionsweise zu erklären und einige der Herausforderungen zu diskutieren, die bei ihrer Implementierung und ihrem Training auftreten.

Abbildung 1.1: Beispiel für ein Bild, das von einem GAN generiert wurde. Ausgangssatz: „Ein roter Panda, der ein Einrad fährt“

Das Paper ist wie folgt strukturiert: Nach dieser Einleitung werden in Kapitel 2 die Grundlagen von neuronalen Netzen, Deep Learning und generativen Modellen erläutert. Kapitel 3 ist den GANs gewidmet, wobei ihr Konzept, ihre Architektur und ihr Training im Detail besprochen werden. Kapitel 4 behandelt verschiedene Anwendungen von GANs, während Kapitel 5 einige der Herausforderungen und Lösungsansätze bei der Arbeit mit GANs diskutiert. Schließlich werden in Kapitel 6 Schlussfolgerungen gezogen und ein Ausblick auf zukünftige Forschungsrichtungen gegeben.

2 Grundlagen

2.1 Künstliche Neuronale Netze

Bevor wir uns GANs genauer anschauen können, ist es wichtig, die Grundlagen von künstlichen neuronalen Netzen zu verstehen, auf welchen die Technologie fußt.

2.1.1 Prinzip

Künstliche Neuronale Netze (KNNs) sind ein wichtiger Zweig der Künstlichen Intelligenz und bilden die Basis für Deep Learning-Technologien, welche unter anderem auch GANs umfassen. Sie werden bereits heutzutage erfolgreich in verschiedenen Prozessen, wie Mustererkennung, Kategorisierung- und Prognose von Daten oder Optimierung von Abläufen eingesetzt. Ihre Arbeitsweise liegt darin, eine Menge von Eingaben in sogenannte Eingabevektoren zu kodieren und durch das neuronale Netz daraus eine Menge an Ausgabevektoren zu generieren. Diese Ausgabevektoren können wiederum in ein Ergebnis beliebigen Formats (z.B. Binärdaten, Text, Audio oder Grafiken) kodiert werden. Die Struktur von KNNs sind von der Funktionsweise des menschlichen Gehirns inspiriert und bestehen aus einer Reihe von miteinander verbundenen Knoten, die als Neuronen bezeichnet werden und als simple Prozessoren fungieren. Diese Prozessoren können nur einfache Operationen ausführen, sind aber in der Lage, komplexe Aufgaben zu erledigen, wenn sie in großer Anzahl miteinander verbunden sind. Die Verbindungen zwischen den Neuronen werden als Kanten bezeichnet und haben ein Gewicht, das die Stärke der Verbindung zwischen den Neuronen angibt. Die Menge der Kanten und deren Gewichtungen definieren dabei die Transformation, welche die Eingangsvektoren beim Durchlauf des KNNs erfahren. Die Neuronen sind in Schichten angeordnet, wobei jede Schicht eine Reihe von Neuronen enthält, die eine bestimmte Funktion ausführen. Die erste Schicht wird als Eingabeschicht bezeichnet, die letzte als Ausgabeschicht und alle dazwischen liegenden Schichten werden als versteckte Schichten bezeichnet.[1]

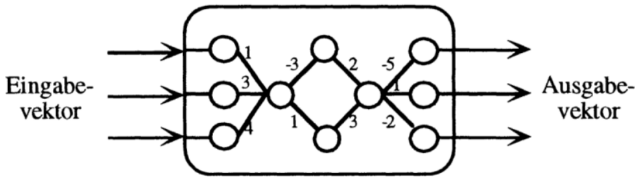

Abbildung 2.1: Schematische Darstellung eines Neuronalen Netzes

2.1.2 Merkmale

KNNs zeichnen sich durch folgende Merkmale aus:

- KNNs sind in der Lage, aus Beispielen zu lernen, ohne explizit programmiert zu werden. Dieser Vorgang wird als Lernen oder Training bezeichnet und ist der wichtigste Aspekt von KNNs. Das Lernen erfolgt durch Anpassung der Gewichtungen der Kanten zwischen den Neuronen, um die gewünschte Ausgabe zu erzeugen und wird wiederholt, bis die KI bei bestimmten Eingaben die erwünschte Resultat erzeugt, oder zumindest annähert. Näheres dazu in Kapitel 2.
- KNNs sind sehr robust und fehlertolerant, da sie in der Lage sind, auch bei fehlerhaften oder unvollständigen Daten zu arbeiten. Dies ist ein großer Vorteil gegenüber herkömmlichen Algorithmen, die bei verrauschten Daten, wie sie beispielsweise in Sensormesswerten vorkommen, häufig nicht mehr zuverlässig funktionieren.
- KNNs können mit einer großen Menge an Daten umgehen und sind in der Lage, Muster in diesen Daten zu erkennen. Dadurch können Lernstrategien oder Entscheidungen auf Basis von Erfahrungen getroffen werden, was ideal für die Inter- und Extrapolation von Daten ist.
- KNNs bieten in ihrer Anwendungsphase eine durchaus gute Performanz, die sich mit der Größe des Netzes und der Anzahl der Trainingszyklen verbessert. Die Trainingsphase hingegen ist sehr rechenintensiv und kann je nach Größe des Netzes und der Anzahl der Trainingszyklen mehrere Stunden oder sogar Tage dauern. Zudem benötigt eine erfolgreiche Trainingsphase eine große Menge an qualitativen Trainingsdaten, die in der Regel manuell ausgewählt werden und vorsichtig administriert werden müssen.

2.2 Deep Learning

Wie bereits im vorherigen Kapitel erwähnt, bieten künstliche neuronale Netze die Möglichkeit, komplexe Zusammenhänge in Daten zu erkennen. Um diese Fähigkeit zu erlangen, müssen die Netze jedoch erst gebaut werden. Dies geschieht durch die Bildung neuer Schichten und die Anpassung der Gewichte der einzelnen Neuronen. Zu den frühen Zeiten der künstlichen Intelligenz, wurden neuronale Netze von Menschen wie Algorithmen aufgebaut, was nicht nur die Komplexität der Netze begrenzte, sondern voraussetzte, dass die Entwickler den Lösungsweg mathematisch beschreiben konnten. Besonders bei Themen, wie der Erkennung von Sprache und Gesichtern, die vom Gehirn intuitiv erledigt werden, erwies sich dies als sehr schwer. Deshalb wurde der Prozess des *Deep Learnings* entwickelt. Hier soll das neuronale Netz nicht manuell gebaut werden müssen, sondern baut sich selbst, ähnlich wie es das menschliche Gehirn tut. Um dies zu erreichen, muss das Netz trainiert werden. Hierfür wird eine Sammlung von Ein- und Ausgangswertpaaren an die KI gefüttert, welche anschließend die Gewichte so anpasst, dass die Ausgabe des Netzes möglichst nahe an der gewünschten Ausgabe liegt. Dieser Vorgang wird als *Backpropagation* bezeichnet. Die Differenz zwischen der gewünschten und der tatsächlichen Ausgabe wird berechnet und auf die Gewichte der einzelnen Neuronen zurückgeführt. Die Gewichte werden dann so angepasst, dass die Differenz zwischen gewünschter und tatsächlicher Ausgabe minimiert wird. Dieser Vorgang wird so lange wiederholt, bis die Differenz zwischen gewünschter und tatsächlicher Ausgabe minimal ist. Durch diesen Prozess entstehen eine oder mehrere sogenannte *hidden layers*, abstrakte Schichten, welche die Ein- und Ausgangsschichten miteinander verbinden und die Daten in immer abstraktere Formen umwandeln. Die Anzahl der hidden layers und die Anzahl der Neuronen in diesen Schichten sind frei wählbar. Je mehr hidden layers und Neuronen vorhanden sind, desto komplexere Zusammenhänge können erkannt werden. Dieser Prozess wird in Abbildung 2.2 dargestellt.[2]

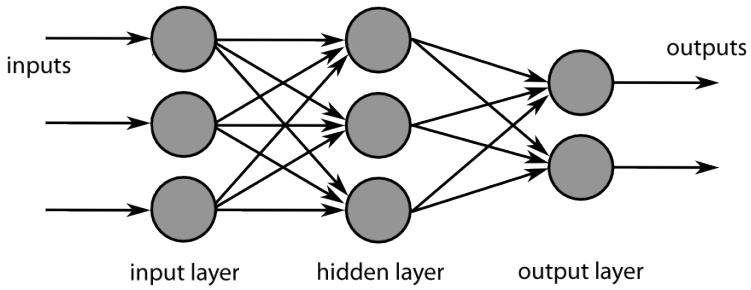

Abbildung 2.2: Beispiel für neuronales Netz mit einer „hidden layer“
Quelle: `http://commons.wikimedia.org`

Deep Learning ist eine Unterkategorie des Machine Learnings und wird auch als *hierarchisches Lernen* bezeichnet, da die Daten in immer abstraktere Formen umgewandelt werden. Dabei ist der Oberbegriff Machine Learning als solches nicht nur auf neuronale Netze beschränkt, sondern kann auch auf andere Technologien angewendet werden, wie zum Beispiel auf Entscheidungsbäume, Bayes-Netze oder Support Vector Machines. Von Deep Learning wird allerdings nur gesprochen, wenn neuronale Netze verwendet werden. Der Name „Deep Learning“ stammt dabei von der Tiefe der verborgenen *hidden layers*.

Der Prozess des Deep Learnings ist sehr rechenintensiv und benötigt eine große Menge an Trainingsdaten, die in der Regel manuell ausgewählt werden und vorsichtig administriert werden müssen. Zudem ist die Trainingsphase sehr langwierig und kann je nach Größe des Netzes und der Anzahl der Trainingszyklen mehrere Stunden oder sogar Tage dauern. Die Anwendungsphase hingegen ist sehr performant und kann in Echtzeit erfolgen. Der eigentliche Prozess des Deep Learnings ist dabei nicht neu, sondern wurde bereits in den 1980er Jahren entwickelt. Damals scheiterte die Technologie jedoch an der mangelnden Rechenleistung der Computer und der geringen Menge an Trainingsdaten. Erst durch die Entwicklung von leistungsstarken Grafikkarten und die Verfügbarkeit großer Datenmengen, wie sie zum Beispiel im Internet zu finden sind, wurde Deep Learning zu einer praktikablen Technologie.

Durch die Fähigkeit aus großen Datensätzen komplexe Zusammenhänge zu erkennen, ist Deep Learning gut für die Replikation abstrakter Prozesse geeignet, für welche sich das Entwickeln eines klassischen Algorithmus als schwierig herrausstellt. Besonders wenn die Daten sehr komplex sind und große Datensätze für das Training verfügbar sind. Exzellente Beispiele für den Einsatz von Deep Learning wären zum Beispiel die Erkennung von Sprache und Gesichtern. Solche Technologien sind dabei schon weitläufig in Anwenderendgeräten im Einsatz, wie beispielsweise in Form von virtuellen Assistenten, die die Sprache des Anwenders verstehen können oder Kamera-Apps, die die Motive und Gesichter auf Fotos erkennen und anhand dessen klassifizieren. Aber auch in der Medizin und der Biologie wird Deep Learning eingesetzt, um zum Beispiel Krebszellen zu erkennen oder die Struktur von Proteinen zu analysieren. Auch in der Robotik wird Deep Learning eingesetzt, um Roboter zu entwickeln, die sich selbstständig bewegen können.

2.3 Generative Modelle

Generative Modelle sind Modelle, welche durch unüberwachtes Lernen erzeugt werden, einer Unterkategorie des Machine Learnings, in welcher sich auch GANs einordnen lassen. Im Gegensatz zu anderen Deep Learning Modellen, wie zum Beispiel Convolutional Neural Networks (CNNs), welche die Daten klassisch als Input-Output-Paar oder auch Daten-Label-Paar zum Training bereitgestellt bekommen, werden bei generativen Modellen keine Labels benötigt, um die Daten zu klassifizieren. Stattdessen werden die Daten selbst analysiert und daraus ein Modell erzeugt, welches die Daten möglichst gut abbildet. Dieses Modell kann dann verwendet werden, um neue Daten zu generieren, welche den Trainingsdaten möglichst ähnlich sind.[3]

Diese Art des Machine Learnings wird in vielen Bereichen eingesetzt, wie zum Beispiel in der Bildverarbeitung, der Spracherkennung, der Sprachsynthese, der Textverarbeitung oder der Musiksynthese. Sie werden genutzt, um Daten zu generieren, die kaum von natürlichen Daten zu unterscheiden sind.

Generative Modelle können in zwei Kategorien unterteilt werden: *explicit generative models* und *implicit generative models*. Bei *explicit generative models* wird die Wahrscheinlichkeitsverteilung der Daten explizit modelliert. Dies geschieht in der Regel durch die Verwendung von Bayes'schen Netzen, welche die Wahrscheinlichkeitsverteilung der Daten durch die Verwendung von Bayes'schen Regeln modellieren. Bei *implicit generative models* wird die Wahrscheinlichkeitsverteilung der Daten nicht explizit modelliert, sondern durch ein Modell approximiert. Dies geschieht in der Regel durch die Verwendung von neuronalen Netzen, welche die Wahrscheinlichkeitsverteilung der Daten durch die Verwendung von neuronalen Netzen approximieren. Bei GANs handelt es sich um ein *implicit generative model*, welches mit neuronalen Netzen arbeitet. In Kapitel 3 werden GANs genauer erläutert.

3 Generative Adversarial Networks

3.1 Konzept

Generative Adversarial Networks (GANs) basieren auf der zuvor genannten Technologie des unüberwachten Lernens und bedienen somit ähnliche Einsatzzwecke, nämlich der Generation von Daten, die vom menschlichen Gehirn nicht von reellen Daten unterscheiden lassen. Im Falle der GANs geschieht dies mit der Hilfe von zwei unabhängigen, impliziten generativen Modellen: dem Generator und dem Diskriminator. Diese beiden Modelle werden als „adversarial“ (gegensätzlich) bezeichnet, da sie im Wettbewerb zueinander stehen und sich gegenseitig trainieren.

Dieses Art der künstlichen Intelligenz hat sich vor allem im Bezug der Bildverarbeitung als sehr erfolgreich erwiesen. Dies wird nicht nur in der reinen Generation von Bildern eingesetzt, sondern erlaubt ebenfalls andere bildbezogenen Methoden, wie die Super-Resolution, welche es erlaubt Bilder mit niedrigem Detailgrad neue Details zu erschaffen, die im Original nicht vorhanden sind. Eine weitere Option ist der sogenannte Style Transfer, welcher die Möglichkeit bietet den Stil eines Bildes, z.B. realstisch, abstrakt, Zeichentrick, etc. auf ein anderes Bild übertragen zu können. Besonders die Super-Resolution, auch AI-based Upscaling genannt, wird bereits heutezutage weitläufig eingesetzt, um beispielsweise Musikvideos, welche vor langer Zeit auf analoge Formate, wie Magnetbändern aufgezeichnet wurde, zu restaurieren und den analogen, 576i-Standard Definition-Look zu entfernen. Dadurch können mit genug Feinarbeit alte Videos so bearbeitet werden, dass sie nicht nur eine morderne Auflösung wie UHD besitzen, sondern auch wesentlich schärfer und klarer aussehen.

Um besser nachvollziehen zu können, wie die zwei konkurrierenden Modelle in solch komplexen Prozessen resultieren können, lohnt es sich hier ein genauer Blick auf die Architektur der GANs zu werfen.

3.2 Architektur

3.2.1 Allgemein

Die Basis eines GANs bilden wie bereits erwähnt zwei Modelle, der Generator und der Diskriminator. Diese beiden Modelle agieren als Gegenspieler und trainieren sich gegenseitig, um zu einem besseren Gesamtresultat zu führen. Somit gehören beide Komponenten stets zu einander und sind jeweils beide für die korrekte Funktion der Netzes unverzichtbar. Die Daten, mit welchen die Modelle arbeiten, können in Form von verschiedenen Formaten definiert sein, wie Bildern, Texten oder sogar Musik, in den nachfolgenden Kapiteln wird allerdings ausschließlich der visuelle Aspekt von GAN betrachtet. Dabei unterscheiden sich die Arbeitsweisen der beiden Modelle grundlegend:

Generator: Der Generator erzeugt neue Daten, die echten Daten möglichst ähnlich sehen sollen. Es ist das Modell der beiden, welches nach dem Abschließen des Trainings für die tatsächliche Generation von neuen Daten verwendet wird. Der Generator wird mit zufälligem Rauschen als Eingabe gestartet, weleches als Latent Space bezeichnet wird. Dieses Rauschen in Form eines Vektors wird als Eingabe in den Generator eingespeist, welcher durch das Feedback des Diskriminators im Laufe der Zeit lernt, Daten zu erzeugen, die von einem echten Datensatz nicht zu unterscheiden sind. Dieser Prozess wird auch als *Backpropagation* bezeichnet und erlaubt des unüberwachte Lernen der Netze.

Diskriminator: Der Diskriminator hat die Aufgabe, zwischen echten Daten und den vom Generator erzeugten Daten zu unterscheiden. Er wird mit einer Mischung aus echten und generierten Daten trainiert und wird nach Abschluss des Trainings für die eigentliche Generation von Bildern nicht mehr benötigt, ist jedoch dennoch für die Funktion des GANs unverzichtbar. Der Diskriminator lernt, die beiden Arten von Daten auseinanderzuhalten, indem nach der Entscheidung „echt/nicht echt“ die Korrektheit der Entscheidung ausgewertet wird. Dies ist möglich, da der Diskriminator im Gegensatz zum Generator Zugriff auf die Realdaten hat. Dieser Prozess geschieht ebenfalls durch Backpropagation.

Der Prozess des Trainings wird in einem Nachfolgenden Kapitel noch näher behandelt.

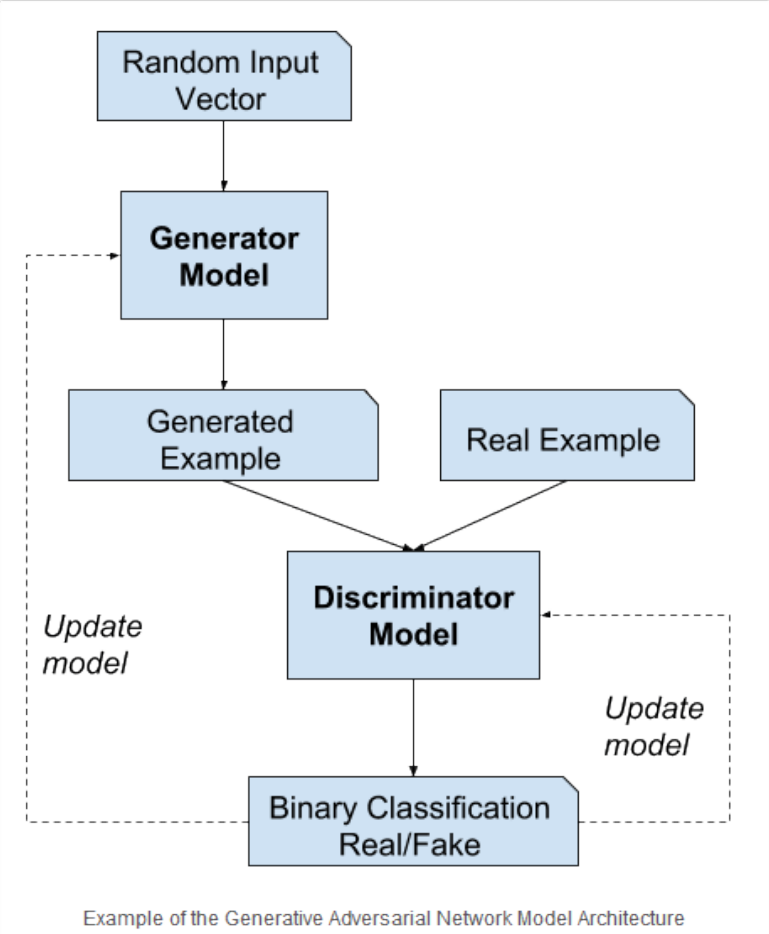

Abbildung 3.1: Funktionsprinzip eines GAN

Die Abbildung zeigt die grundlegende Architektur eines GAN, wie es auch schon in den Grundlagen zuvor erklärt wurde. Der Generator erhält als Eingabe einen zufälligen Vektor, der als Latent Space bezeichnet wird. Dieser Vektor wird als Eingabe in den Generator eingespeist und durchläuft eine Reihe von Schichten, die jeweils eine Reihe von Neuronen enthalten. Die letzte Schicht des Generators, die Ausgabeschicht, gibt einen Vektor aus, der die generierten Daten enthält. Der Diskriminator erhält schließlich als Eingabe entweder diesen oder einen aus echten Daten stammenden Vektor als Eingangswert. Dieser Vektor durchläuft anschließend einen ähnlichen Prozess, wie der Latent Space in Form des Diskriminators. Die letzte Schicht des Diskriminators gibt schließlich einen Vektor aus, der die Wahrscheinlichkeit angibt, ob die Eingabe aus echten oder generierten Daten besteht. Diese Wahrscheinlichkeit bietet in Form der *Backpropagation* die Grundlage für das Training beider Modelle, allerdings auf zwei unterschiedliche Arten und Weisen. Sollte der Diskriminator nämlich falsch liegen und beispielsweise ein künstlich generiertes Bild als real anerkennen, so gilt dies für den Generator als Erfolg, für den Diskriminator allerdings als Fehlschlag. Dieses Verhältnis der beiden Modelle kann durch

eine Funktion, die sogenannte *Minimax-Verlustfunktion* 3.1 beschrieben werden.

$$L_{GAN} = \frac{1}{N_1}\sum_{i=1}^{N_1} \ln D(\mathbf{x}_i) + \frac{1}{N_2}\sum_{j=1}^{N_2} \ln\left(1 - D(G(\mathbf{z}_j))\right) \tag{3.1}$$

Der Diskriminator versucht dabei den oberen Term zu maximieren, indem korrekt zwischen echten und generierten Daten unterschieden wird. Der Generator hat allerdings die Fähigkeit den zwei Term zu beeinflussen, welchen er zu minimieren versucht. Diese Balance zwischen den beiden Modellen und ihren gegenseitigen Einflüssen, wird durch diesen Term gezeigt, welcher seinen Namen *Minimax* exakt auf Grund dieser Eigenschaft besitzt.

3.2.2 Variationen

Die Grundlagen der Funktionsweise eines GANs sind nun bekannt. Allerdings existieren noch viele weitere Variationen, die sich in der Regel auf bestimmte Anwendungsfälle spezialisieren. Die Grundlagen der Funktionsweise bleiben allerdings bei allen Variationen gleich.

Conditional GAN Ein bekanntes Beispiel für eine Erweiterung der GAN-Architektur ist die Verwendung von Datenlabels. Diese Datenlabel werden in der Regel von Menschenhand erstellt und geben an, ob ein Bild beispielsweise eine Katze oder einen Hund zeigt. Diese Datenlabel können nun in den Trainingsprozess mit einbezogen werden, indem der Diskriminator nicht nur zwischen echten und generierten Daten unterscheidet, sondern auch zwischen den verschiedenen Klassen, die durch die Datenlabel definiert werden. Dieser Prozess wird als *Conditional GAN* bezeichnet und besitzt in der Praxis meist einen größeren Nutzen für die Bildgeneration als ein einfaches GAN. Dessen Generator wird nämlich lediglich durch den randomisierten Eingang gesteuert, welcher es nicht zulässt Einfluss auf die Ausgabe zu nehmen. Die Aushabe ist also zufällig. Bei einer Conditional GAN hingegen, kann dem Generator zusätzlich bestimmte Eingangsbedingungen mitgegeben werden, die beispielsweise den Stil oder das Motiv des resultierenden Bildes beeinflussen können.

Deep Convolutional GAN Ein weitere Möglichkeit der Verbesserung der Architektur sind die sogenannten Deep Convolutional Generative Adversarial Networks (DCGANs). Diese Variation der klassischen GANs nutzt spezielle neuronale Netze als Generatoren und Diskriminatoren, die CNNs. Diese besitzten auf der Eingangsseite sogenannte *Convolutional Layer*, was auf Deutsch etwa „faltende Schicht“ bedeutet. Diese Schicht erlaubt es dem Netzwerk Verhältnisse zwischen benachbarten Pixeln zu erkennen, wodurch sich CNNs hervorragend zur Erkennung von Mustern eignet. Dies vereinfacht die Konvertierung von Bilddaten in einem zweidimensionalen Raster zu den Modell-internen Datenstrukturen. Dadurch eignen sich DCGANs besonders gut für visuelle Einsatzzwecke, wobei klassische GANs auch für andere Gebiete eignen.

Recurrent Adversarial Networks Die Architektur der Recurrent Adversarial Networks (RANs) ist eine weitere Variation der GANs, die sich besonders für die Verarbeitung von Texten und Audiodaten eignet und bietet einen Gegenpol zu den DCGANs, welche auf Bilddaten spezialisiert sind. Diese Variation nutzt die sogenannten *Recurrent Neural Networks*, welche sich besonders gut für die Verarbeitung von Sequenzen und Echtzeit-Daten eignen. Sie bestehen aus einem *Convolutional Encoder* und einem *Recurrent Decoder*. Der Encoder wandelt die Eingangsdaten in einen Vektor um, der anschließend vom Decoder in die gewünschte Form zurückgewandelt wird. Diese Architektur eignet sich besonders gut für die Verarbeitung von Texten und Audiodaten, da diese in der Regel in Form von Sequenzen vorliegen.

Dies sind nur einige der vielen Variationen, die sich in der Regel auf bestimmte Anwendungsfälle spezialisieren. Die Grundlagen der Funktionsweise bleiben allerdings bei allen Variationen gleich. Weitere Beispiele für GAN-Variationen, auf die in diesem Paper nicht näher eingegangen wird, sind:

- Laplacian GAN
- Wasserstein GAN
- Energy-based GAN
- CycleGAN
- InfoGAN
- ...

3.3 Training

Das Training von GANs basiert auf unüberwachtem Lernen und erfolgt in einem iterativen Prozess, der in drei Schritten durchgeführt wird:

1. Der Generator erzeugt eine Reihe von Daten auf basis des Latent Spaces.
2. Der Diskriminator erhält zufällig das vom Generator erzeugte Bild oder ein echtes Bild und überprüft die Echtheit.
3. Das Ergebnis des Diskriminators wird ausgewertet und beide Netze werden durch das Resultat beeinflusst.

Dabei werden beiden Modelle gleichzeitig trainiert. Der Generator wird trainiert, um den Diskriminator zu täuschen, indem er Daten erzeugt, die von echten Daten nicht zu unterscheiden sind. Der Diskriminator wird trainiert, um den Generator zu täuschen, indem er die vom Generator erzeugten Daten nicht von echten Daten unterscheiden kann. Die beiden Modelle trainieren sich schließlich gegenseitig, bis der Diskriminator circa die Hälfte der Daten nicht korrekt zuordnen kann. Zu diesem Zeitpunkt ist der Generator in der Lage, Daten zu erzeugen, die von echten Daten nicht zu unterscheiden sind und das Netz gilt als fertig trainiert.

Trotz seiner simplen Architektur ist das Training eines GANs ein komplexer Prozess, der viel Zeit und Rechenleistung benötigt. Der Prozess ist dabei sehr rechenintensiv und benötigt ein großes Maß an Daten. Die rechenintensive Natur wird allerdings im Laufe der Zeit immer weniger zum Problem werden, aufgrund Moore's Law. Dieses besagt, „dass sich die Anzahl an Komponenten in einem einzigen integrierten Schaltkreis bei minimalen Kosten jedes Jahr verdoppelt." [4] Dieser Trend wird sich zwar in den nächsten Jahren verlangsamen, dennoch ist davon auszugehen, dass sich die Leistung von Rechnern weiterhin steigern wird. Für das Problem der großen benötigten Datensätze bietet das Internet eine Lösung. Dieses ist nämlich eine nahezu unerschöpfliche Quelle an Daten. Diese Daten können dabei jedoch nicht einfach so übernommen werden, denn das Trainieren einer KI erfordert Ausgangsdaten, die vorsichtig ausgewählt und überprüft werden müssen. Dies erzeugt einen erheblichen Mehraufwand und ist ein großer einschränkender Faktor für die Entwicklungen von künstlichen Intelligenzen und somit auch GANs.

4 Anwendungen von GANs

Das Ziel von GANs ist es, Modelle zu schaffen, die in der Lage sind, Daten zu generieren, die von menschlichen Beobachtern nicht von echten Daten unterschieden werden können. GANs haben Anwendungen in verschiedenen Bereichen gefunden, darunter Bildgenerierung, Stiltransfer, Super-Resolution, Generierung von realistischen Texten und mehr.

4.1 Bildsynthese

Im Bereich der Bildgenerierung werden diverse GANs bereits erfolgreich eingesetzt. Darunter sind Beispielsweise:

- Artbreeder
- StyleGAN
- BigGAN
- CycleGAN

Jedoch ermöglicht das antagonistische System eines GAN, nicht nur die Möglichkeit bilder zu generieren, sondern auch generierte Bilder zu erkennen. So ist es möglich zu unterscheiden ob es sich beim vorliegenden Bild um ein Original handelt oder es durch eine AI generiert wurde. Dies hat insbesondere Relevanz, da in bezug auf, mittel künstlicher Intelligenz generierter Kunst, immer wieder die Frage des Urheberrechtschutzes im raum liegt. So ist es dank GAN möglich zu erkennen, falls urheberrechtsgeschützte Bilder als Ausgangsmaterial verwendet wurden.

Dadurch könnten zukünftig einige moralische und ethische Fragen der KI-Kunst geklärt werden.

4.2 Super-Resolution

4.3 Style Transfer

5 Herausforderungen und Lösungsansätze

5.1 Mode Collapse

5.2 Training Instability

6 Schlussfolgerungen und Ausblick

Literatur

[1] Andreas Scherer. *Neuronale Netze: Grundlagen und Anwendungen.* Vieweg, Wiesbaden, 1997. ISBN: 978-3-528-05465-6.

[2] John D. Kelleher. *Deep Learning.* The MIT Press, Cambridge, Massachusetts, 2019. ISBN: 978-0-262-53755-1.

[3] Kenny Choo. *Machine Learning kompakt.* Springer Spektrum, Wiesbaden, 2021. ISBN: 978-3-658-32267-0.

[4] Mitja Schmakeit. *Moore's Law.* Juni 2023. URL: `cl-informatik.uibk.ac.at/teaching/ss14/ewa/reports/ss13-MS.pdf`.