Bilderzeugung mit Generative Adversarial Networks

Modularbeit

Hauptseminar

von

Manyue Zhang

Juni 2023

Professsor Prof. Dr. Alfred Nischwitz

Inhaltsverzeichnis

Abkürzungsverzeichnis

GAN	Generative Adversarial Network
GANs	Generative Adversarial Networks
CelebA	CelebFaces Attributes Dataset
cGANs	conditional Generative Adversarial Networks
JSD	Jensen-Shannon-Divergenz
FID	Fréchet Inception Distance
DCGANs	Deep Convolutional GANs
WGAN	Wasserstein GAN
cGANs	Conditional GANs

Abbildungsverzeichnis

1 Einleitung

GANs sind zu einem beliebten Forschungsgebiet der künstlichen Intelligenz geworden. Die Grundlage von ACGAN stammt aus dem Zwei-Spieler-Nullsummenspiel der Spieltheorie und besteht aus einem Generator und einem Diskriminator, die durch entgegengesetztes Lernen trainiert werden. Ziel ist es, die potenzielle Verteilung von Samles abzuschätzen und neue Samples zu erzeugen. Die Bildgenerierung ist eine der größten Anwendungen von GANs, die beispielsweise zum Erzeugen realistischer Objekte und Gesichter, zum Erstellen realistischer Innen- und Außenszenen, zum Wiederherstellen von Orginalbildern aus segmentierten Bildern, zum Einfärben von Schwarz-Weiß-Bildern und für Super-Resolution verwendet werden kann. [6]. Darüber hinaus wurden GANs bei Problemen wie der Sprachverarbeitung[13][19], der Überwachung von Computerviren[9] und bei Schachturnieren[5] eingesetzt.

Kapitel 2 erläutert die Funktionsweise und Architektur von GANs und beleuchtet die Vorteile und Herausforderungen von GANs. Außerdem wird auf die Entwicklung von GANs eingegangen.

Kapitel 3 widmet sich den verwendeten Datensätzen, insbesondere dem Datensätzen CelebFaces Attributes Dataset (CelebA) und CIFAR-10.

Kapitel 4 erläutert die Bewertungskriterien Jensen-Shannon-Divergenz (JSD) und FID, die zur objektiven Bewertung der generierten Daten dienen.

Kapitel 5 führt in Conditional GANs ein, beschreibt deren Architektur und demonstriert die Bildgenerierung mit Conditional GANs (cGANs) am Beispiel des CIFAR-10 Datensatzes. Darüber hinaus werden Herausforderungen und zukünftige Entwicklungen von cGANs diskutiert.

2 Generative Adversarial Networks

2.1 Funktionsweise und Architektur

Die Funktionsweise von GANs ist inspiriert von der Spieltheorie, zwei neuronale Netze, der Generator und Diskriminator ergänzen sich gegenseitig um ein Nash-Gleichgewicht beim Training zu erreichen. [7]

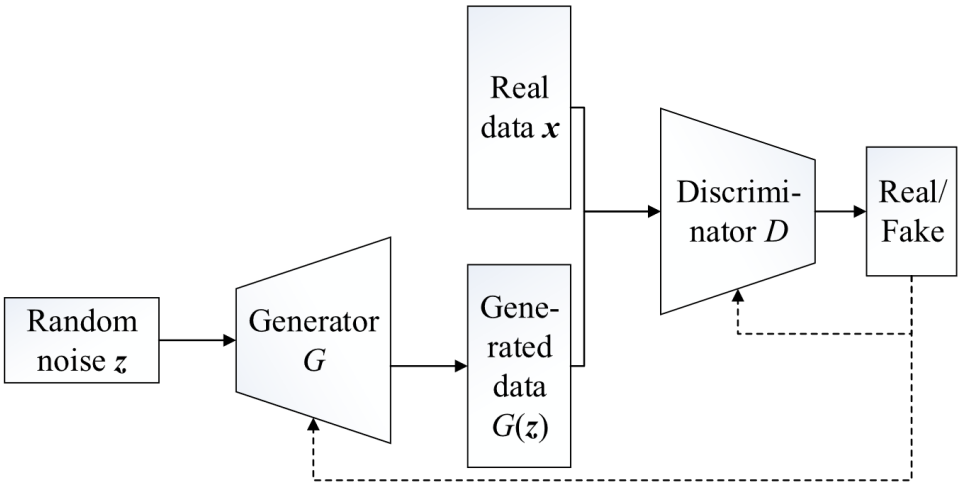


Abbildung 2.1: Grundlegende Architektur eines GANs [2]

Abbildung 2.1 zeigt den Aufbau eines einfachen GANs, bestehnd aus dem Generator G und dem Diskriminator D. Das Ziel von G ist, Daten zu erzeugen, die der Verteilung der echten Daten möglichst ähnlich sind. Das Ziel von D ist dagegen, die von G generierten Samples von den echten Daten zu unterscheiden.

Der Generator G erhält als Eingabe einen zufälligen Noise-Vektor z (meist eine Normal- oder Gleichverteilung). z wird durch den Generator auf ein gefälschtes Sample $G(z)$ abgebildet.

Bei dem Diskriminator D handelt es sich um einen binären Klassifikator. Als Eingabe erhält er sowohl echte Samples x als auch die vom Generator erzeugten Samples $G(z)$ und liefert die Wahrscheinlichkeit zurück, mit der es sich um ein echtes bzw. gefälschtes Sample handelt.

Sobald der Diskriminator D nicht mehr feststellen kann, ob die Daten aus dem Datensatz x oder vom Generator $G(z)$ stammen, ist der optimale Zustand erreicht und der Generator hat die Verteilung der realen Daten x gelernt. [2]

2.2 Entwicklung der GANs

GANs haben sich in den letzten Jahren stark weiterentwickelt. Neue Ansätze, Architekturen und Techniken wurden eingeführt, um die Leistung und Stabilität von GANs zu verbessern. Hier sind einige wichtige Meilensteine in der Entwicklung von GANs:

Originaler GAN: Das erste GAN-Modell [7] wurde 2014 von Ian Goodfellow und seinem Team vorgestellt. Es bestand aus einem Generator-Netzwerk und einem Diskriminator-Netzwerk, die miteinander konkurrieren.

DCGAN: Im Jahr 2016 wurden DCGANs [16] (Abbildung 2.2) eingeführt, um die Bildgenerierung mit GANs zu verbessern. Durch die Verwendung von Convolutional Layers in Generator- und Diskriminatornetzwerken konnten realistischere Bilder erzeugt werden.

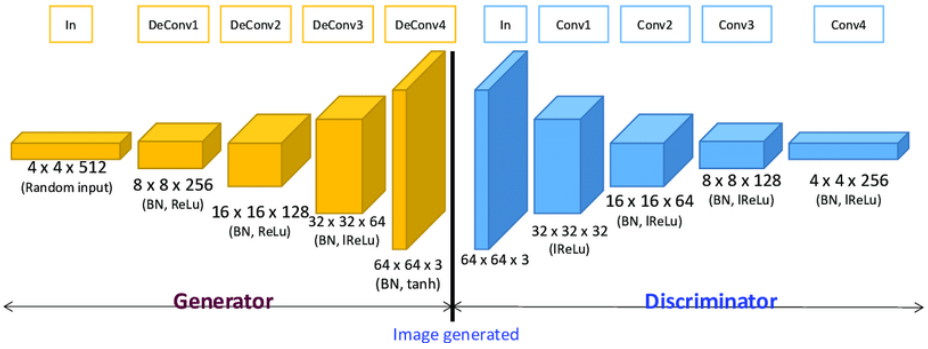


Abbildung 2.2: Archtektur von DCGANs

WGAN: WGANs [1] führten die Wasserstein-Distanz ein, um das Training von GANs stabiler zu machen. Das Problem der verschwindenden Gradienten (Vanishing Gradients) wurde durch die Verwendung von Gradientenabstiegsmethoden in der Diskriminatorverlustfunktion gelöst, siehe Abbildnung 2.3.

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.
Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

while θ has not converged **do**
  for $t = 0, ..., n_{\text{critic}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_r$ a batch from the real data.
    Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
    $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))\right]$
    $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
    $w \leftarrow \text{clip}(w, -c, c)$
  end for
  Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

Abbildung 2.3: Algorithmus von WGAN

Conditional GANs: cGANs [15] ermöglichen die gezielte Generierung von Daten durch die Verwendung zusätzlicher Bedingungen. Diese Bedingungen können beispielsweise Klassenlabels oder Textbeschreibungen sein.

Progressive GANs: Progressive GANs [11] Abbildung 2.4 ermöglichen die schrittweise Generierung von hochauflösenden Bildern. Sie beginnen mit einer niedrigen Auflösung und erhöhen diese schrittweise, um realistische und detaillierte Bilder zu erzeugen.

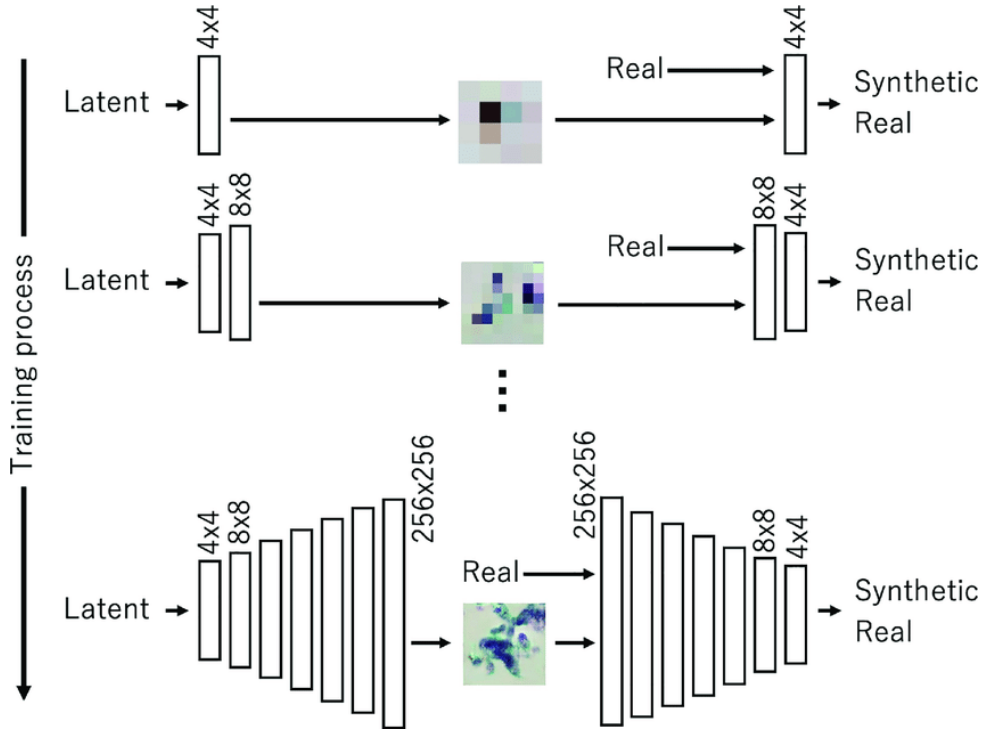


Abbildung 2.4: Architektur von ProGANs

StyleGAN: StyleGANs [10] (Abbildung **??**) erweitern die Kontrolle über die generierten Bilder, indem sie den Stil und die Attribute der generierten Bilder manipulierbar machen. Dadurch ist es möglich, bestimmte Merkmale wie Gesichtsausdruck oder Style von Bilder gezielt zu verändern.

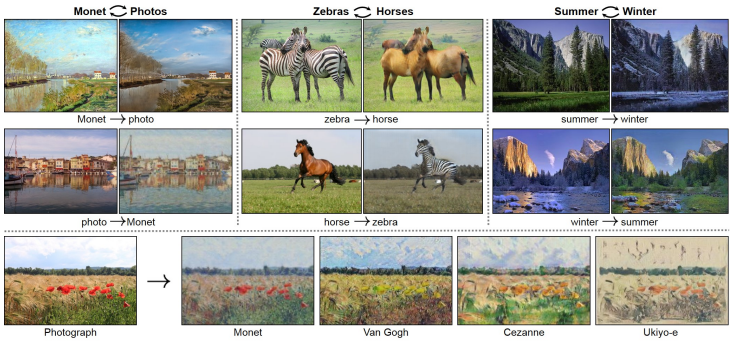


Abbildung 2.5: Beispiele von Style GANs

Diese Fortschritte in der Entwicklung von GANs haben zu beeindruckenden Ergebnissen in der Bilderzeugung, Text-zu-Bild-Synthese, Sprachverarbeitung und vielen anderen Anwendungen geführt. Die kontinuierliche Forschung und Weiterentwicklung von GANs verspricht noch viele zukünftige Entwicklungen und Anwendungen.

2.3 Vorteile und Nachteile / Herausforderungen von GANs

2.3.1 Vorteile von GAN

Das GAN ist ein wirksamer generativer Ansatz zur Generierung von realistischen Daten. Da die Struktur des neuronalen Netzes keine Einschränkungen in Bezug auf die Dimensionalität der erzeugten Daten hat, erweitert es die möglichen generierten Datenmuster, insbesondere bei hoch-dimensionalen Daten. GANs erhalten ihre Flexibilität durch die Verwendung verschiedener Verlustfunktionen. Der Trainingsprozess von GAN nutzt den Kampf zwischen zwei neuronalen Netzen als Trainingskriterium und ermöglicht so eine effiziente Rückwärtspropagation.[17]

Im Vergleich zu anderen Methoden können GANs vielfältige Samples erzeugen. Es erzeugt leicht verständliche Muster und ermöglicht die direkte Generierung hochauflösender und klarer Bilder. Obwohl das GAN ursprünglich nicht für halbüberwachtes Lernen entwickelt wurde, kann es für diese Art des Lernens verwendet werden. Die Kombination der Ergebnisse, die durch das Training mit ungelabelten Datensätzen erhalten wurden, mit einem kleinen Prozentsatz gelabelter Daten kann für klassische Klassifizierungs- und Regressionsverfahren verwendet werden.[3]

2.3.2 Nachteile und Herausforderungen von GANs

Trotz vieler Fortschritte gegenüber anderen Methoden lösen GANs nicht alle Probleme und bringen auch neue Herausforderungen mit sich. Das adversarische Lernkriterium, das das GAN verwendet, ist noch nicht in der Lage, die Konvergenz des Modells und die Existenz von Gleichgewichtspunkten sicherzustellen. Der Trainingsprozess erfordert eine sorgfältige Balance und Synchronisation der beiden Netze, um gute Ergebnisse zu erzielen. Die Kontrolle dieser Synchronisation ist in der Praxis schwierig, wodurch der Trainingsprozess kann instabil sein kann. Zudem zeichnen sich GANs duch eine schlechte Interpretierbarkeit aus, die bei allen auf neuronalen Netzen basierenden Modellen auftritt.[3]

Ein weiteres Problem des GAN ist das Phänomen des Mode Collapse bei dem die Vielfalt der Samples verloren geht und immer die gleichen oder ähnliche Samples erzeugt werden. Dieses Problem wird teilweise von Weiterentwicklungen, die den Trainingsprozess stabilisieren, wie dem WGAN behoben.[18]

Die Lösung des Collapse-Problems und die Verbesserung des Trainingsprozesses bleiben jedoch wichtige Forschungsziele für GANs. Daneben sind die theoretische Konvergenz des

GAN und die Existenz von Gleichgewichtspunkten ebenfalls wichtige Forschungsfragen für die Zukunft.

2.4 Unconditional vs. Conditional GANs

Es können zwei verschiedene Arten von GANs unterschieden werden: conditional GANs und unconditional GANs. Unconditional GANs, die üblicherweise als GANs bezeichnet werden, verwenden nur Bilder als Eingabe, während bei den Conditional GANs zusätzlich Label verwendet werden. Abblidung 5.2 veranschaulicht den Unterschied zwischen den beiden Varianten.

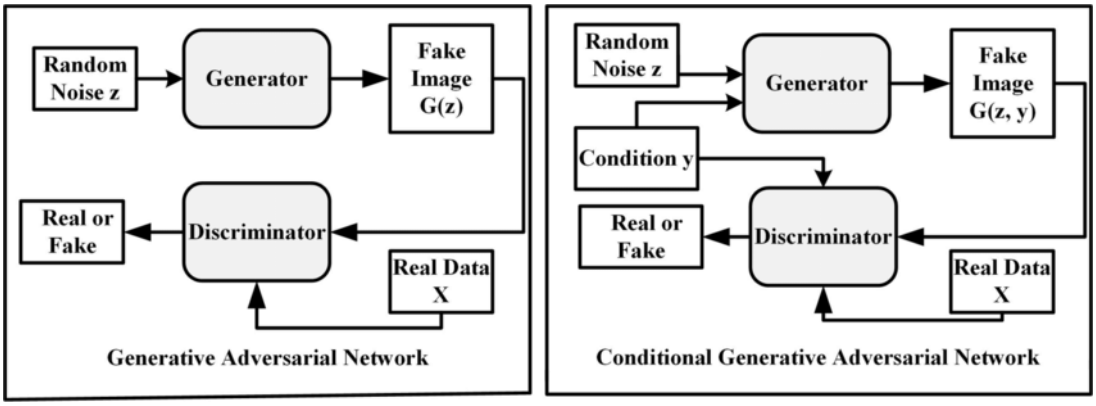


Abbildung 2.6: Vergleich zwischen GAN und conditional GAN [4]

3 Datensätze

3.1 CelebFaces Attributes Datensatz

CelebA [14] ist ein Datensatz mit 202.599 Bildern mit Gesichtern von Prominenten sowie je 40 binäre Label, die für die Bildgenerierung jedoch nicht verwendet werden. Sechs Beispielbilder aus dem CelebA-Datensatz sind in Abbildung 3.1 dargestellt. Die Bilder haben eine Auflösung von 178×218, werden aber für die weitere Verarbeitung auf 64×64 Pixel runterskaliert, einige Beispiele sind in Abbidung 3.2 abgebildet.

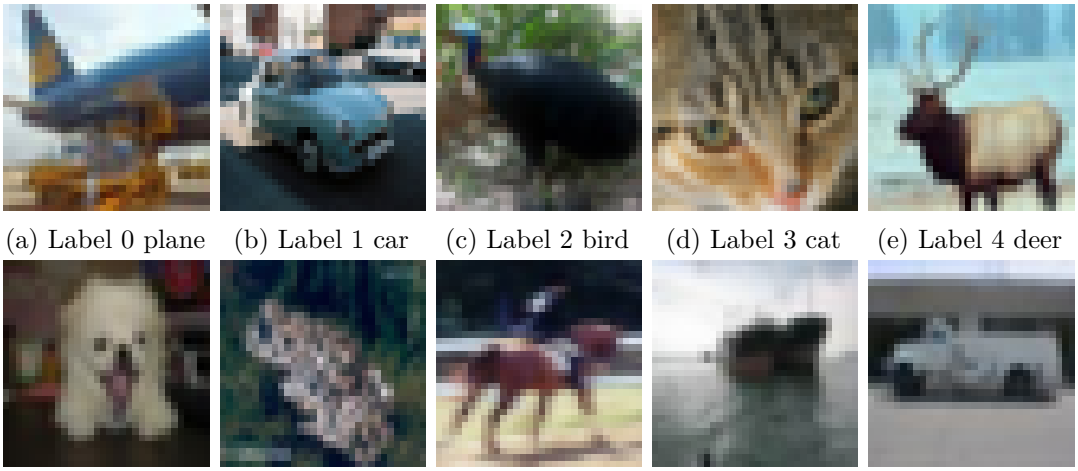
(a) Beispiel-Bild2 (b) Beispiel-Bild3 (c) Beispiel-Bild4

Abbildung 3.1: Beispiele aus dem CeleA-Datensatz

Abbildung 3.2: Auf 64×64 skallierte Beispiele aus dem CeleA-Datensatz

3.2 CIFAR-10 Datensatz

Der CIFAR-10-Datensatz [12] besteht aus 60000 32x32-Farbbildern in 10 Klassen, mit 6000 Bildern pro Klasse. Es gibt 50000 Trainingsbilder und 10000 Testbilder. Der Datensatz ist in fünf Trainings-Batches und einen Test-Batch mit jeweils 10000 Bildern unterteilt. Der Test-Batch enthält genau 1000 zufällig ausgewählte Bilder aus jeder Klasse. Die Trainingsbatches enthalten die restlichen Bilder in zufälliger Reihenfolge, wobei einige Training-Batches mehr Bilder aus einer Klasse als aus einer anderen enthalten können. Insgesamt enthalten die Trainings Batch genau 5000 Bilder aus jeder Klasse.



(a) Label 0 plane (b) Label 1 car (c) Label 2 bird (d) Label 3 cat (e) Label 4 deer

(f) Label 5 dog (g) Label 6 frog (h) Label 7 horse (i) Label 8 ship (j) Label 9 truck

Abbildung 3.3: Beispiele aus dem CIFAR-10 Datensatz

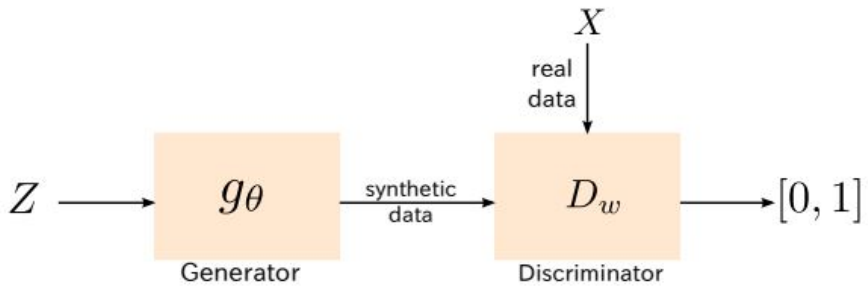
4 Bewertungskriterien

Bei der Bildgenerierung werden die Ergebnisse als "gutëingeschätzt, wenn die generierten Bilder nicht von echten Bildern unterschieden werden können. Ziel der Bewertungskriterien ist es diese Bedinung in eine mathematische Form zu übersetzen. Zu den beliebtesten Metriken gehören der JSD und der FID.

4.1 JSD

Mathematische Formulierung des GANs: $X \in X$ sei die interessierende Zufallsvariable, und P_X sei ihre Wahrscheinlichkeitsverteilung. Das Ziel eines generativen Modells ist die Erzeugung neuer Stichproben aus P_X, gegebene Trainingsdaten $\{X_1, \ldots, X_N\} \sim P_X$.

GAN hat die folgende Architektur:



(a) Darstellung des GAN als mathematische Funktion

Die Parameter des GANs sind (θ, ω). Sie werden durch Lösen des folgenden Min-Max-Problems ermittelt

$$\min_{\theta} \max_{\omega} \mathbb{E}[\log(D_\omega(X)) + \log(1 - D_\omega(g_\theta(Z)))] \tag{4.1}$$

Es gibt verschiedene (spieltheoretische) Möglichkeiten, eine solche Min-Max-Formulierung zu motivieren. Ersetzt man die Maximierung über ω durch die Maximierung über alle Funktionen D, so reduziert sich das Min-Max-Problem auf

$$\min_{\theta} JS(P_X \parallel P_\theta) \tag{4.2}$$

wobei $JS(P_X \parallel P_\theta)$ die JSD zwischen P_X und P_θ ist. Die JS-Divergenz für zwei beliebige Wahrscheinlichkeitsverteilungen p und q ist definiert als

$$JS(p \parallel q) = KL(p \parallel \frac{p+q}{2}) + KL(q \parallel \frac{p+q}{2}) \tag{4.3}$$

4.2 FID

Der FID ist eine Metrik um die Qualität von durch generative Modelle erzeugte Bilder zu bewerten. Die Fréchet Distance $d^2(D_1, D_2)$ zwischen zwei Verteilungen D_1 und D_2 ist definiert durch

$$d^2(D_1, D_2) := \min_{X,Y} E_{X,Y}\|X - Y\|^2$$

wobei eine Minimierung über alle Zufallsvariablen X und Y mit den Randverteilungen D_1 bzw. D_2 erfolgt.

Für multivariate Normalverteilungen D_1 und D_2 kann die Distanz mit

$$d^2(D_1, D_2) := \|\mu_1 - \mu_2\|^2 + Tr(\sum_1 + \sum_2 -2(\sum_1 \sum_2)^{\frac{1}{2}}$$

angegeben werden, wobei μ_i und $\sum_i$ der Mittelwert und Kovarianzmatrix von D_i ist. Der erste Term misst die Distanz zwischen den Mittelwerten der Verteilungen. Der zweite Term

$$d_0(D_1, D_2) := Tr(\sum_1 + \sum_2 -2(\sum_1 \sum_2)^{\frac{1}{2}}$$

definiert eine Metrik auf dem Raum aller Kovarianzmatritzen der Ordnung n.

Für zwei Verteilungen D_R und D_G aus realen bzw. generierten Daten berechnet der FID-Score die Fréchet Distanz zwischen den Verteilungen der echten und generierten Daten unter Verwendung eines Feature Extractors f unter der Annahme, dass die extrahierten Features von eine multivariaten Normalverteilung stammen:

$$FID(D_R, D_G) := d^2(f \circ D_R, f \circ D_G) = \|\mu^R - \mu^G\|^2 + Tr(\sum^R + \sum^G -2(\sum^R \sum^G)^{\frac{1}{2}})$$

wobei μ^R, $\sum^R$ and μ^G and $\sum^G$ die Zentren und Kovarianzmatritzen der Verteilungen $f \circ D_R$ und $f \circ D_G$ sind. Für die Bewertung werden die Mittelwertsvektoren und Kovarianzmatritzen durch Samples der Verteilungen approximiert. [8]

5 Conditional GANs

5.1 Einführung in Conditional GANs

Dieser Abschnitt konzentriert sich auf bedingte (engl. conditional) Varianten von GANs, die besonders für gelabelte Daten wie den in Kapitel 3.1 vorgestellten cifar-10-Datensatz geeignet sind. cGANs sind eine Variante von GANs, bei der sowohl der Generator als auch der Diskriminator zusätzliche Informationen in Form von Bedingungen erhält, die Kategorien oder andere Daten repräsentieren können. Im generativen Modell wird der zufällige Noise-Vektor mit den ConditionalInformationen kombiniert. Die Zielfunktion eines Conditional GAN ist ein binäres Minimax-Spiel mit bedingten Wahrscheinlichkeiten.

5.2 Architektur von Conditional GANs

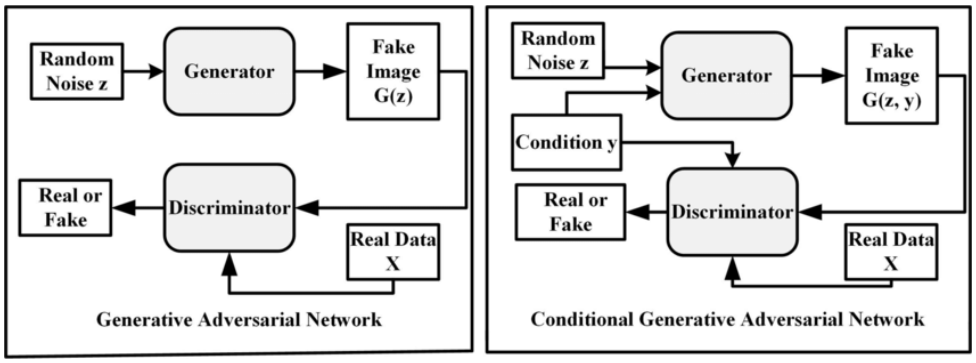


Abbildung 5.1: Vergleich zwischen GANs und cGANs

GGANs können zu einem bedingten Modell erweitert werden, wenn sowohl der Generator als auch der Diskriminator von einer zusätzlichen Information y abhängig gemacht wird. y kann jede Art von zusätzlicher Information sein, wie z.B. Klassenlabels oder Daten aus anderen Datenquellen. Wir können die Bedingung implementieren, indem wir y sowohl dem Diskriminator als auch dem Generator als zusätzliche Eingabeschicht hinzufügen.

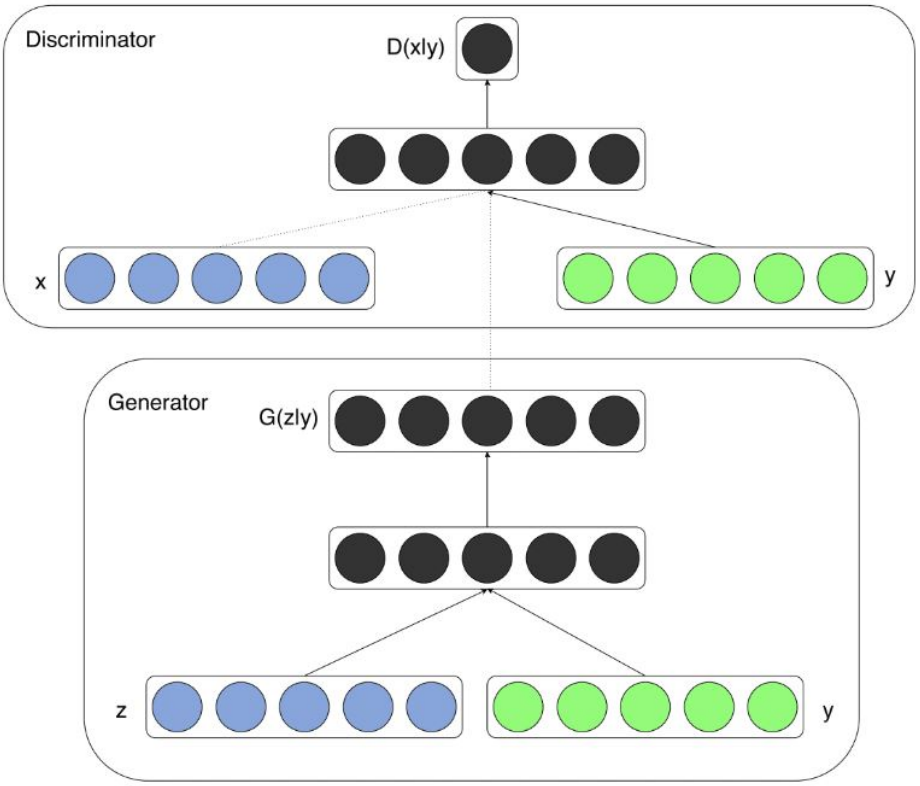


Abbildung 5.2: Archtektur von CGANs

Im Generator werden das Zufallsrauschen $p_z(z)$ und die Bedingung y zu einer gemeinsamen Darstellung kombiniert. Dies ermöglicht dem Generator eine flexible Zusammensetzung dieser verborgenen Repräsentation. Mit Hilfe des adversarialen Trainingsrahmens kann der Generator lernen, wie er das zufällige Rauschen und die Bedingung optimal nutzen kann, um qualitativ hochwertige Bilder zu erzeugen, die den gewünschten Eigenschaften entsprechen. Im Diskriminator werden x und y als Eingaben und für eine Diskriminatorfunktion, in diesem Fall wieder durch einen MLP repräsentiert, dargestellt. Die Zielfunktion eines Minimax-Spiels mit zwei Spielern würde folgendermaßen aussehen:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x|y)] + \mathbf{E}_{z\sim p_z(z)}[\log(1 - D(G(z|y)))] \tag{5.1}$$

5.3 Herausforderungen und zukünftige Entwicklungen von cGANs

CGANs können im Vergleich zu herkömmlichen GANs Bilder mit spezifischen Attributen oder Conditional erzeugen. Allerdings stehen CGANs noch vor einigen Herausforderungen.

Eine der größten Herausforderungen ist die effiziente Generierung von qualitativ hochwertigen Daten auf der Grundlage der gegebenen Bedingungen. Die Qualität der generierten Ergebnisse hängt stark von der Genauigkeit und Repräsentativität dieser Bedingungen ab. Die Auswahl geeigneter und aussagekräftiger Merkmale ist daher von entscheidender Bedeutung, um die Leistungsfähigkeit von cGANs weiter zu verbessern.

Ein weiteres Problem ist die Stabilität des Trainingsprozesses von cGANs. Der so genannte Mode Collapse ist eine Herausforderung, bei der der Generator dazu neigt, nur eine begrenzte Vielfalt von Mustern oder Objekten statt einer breiten Variation von Bildern zu erzeugen. Ein weiteres Problem ist die Stabilität des Trainingsprozesses von cGANs. Der so genannte Musterkollaps stellt eine Herausforderung dar. In diesem Fall neigt der Generator dazu, nur eine begrenzte Vielfalt von Bildern zu erzeugen, statt einer großen Vielfalt von Bildern. Derzeit wird versucht, dieses Problem durch neue Algorithmen zu lösen, um eine größere Vielfalt in den erzeugten Bildern zu erreichen und die Qualität der Bilder zu verbessern.

Darüber hinaus nehmen die Komplexität und die Menge der zu erzeugenden Daten ständig zu, insbesondere im Bereich der hochauflösenden Bilder. Dies stellt eine weitere Herausforderung dar, für fortgeschrittene Techniken und Ressourcen erforderlich sind.

5.4 Bildgenerierung mit Cifar-10 Datensatz

5.4.1 Training

Den gesamten Trainingsprozess haben wir auf Colab ausgeführt und die Programmiersprache Python verwendet. Wir haben das GAN-Modell mit Batch Size 128 75 Epochen lang trainiert. Nach 5 Epochen Abbildung 5.3 sind die erzeugten Bilder von geringer Qualität, grob und pixelig, so dass wichtige Details schwer zu erkennen sind. Bilder mit niedriger Qualität weisen Unschärfen, Artefakte und Verzerrungen auf, was zu einer geringen visuellen Klarheit führt.

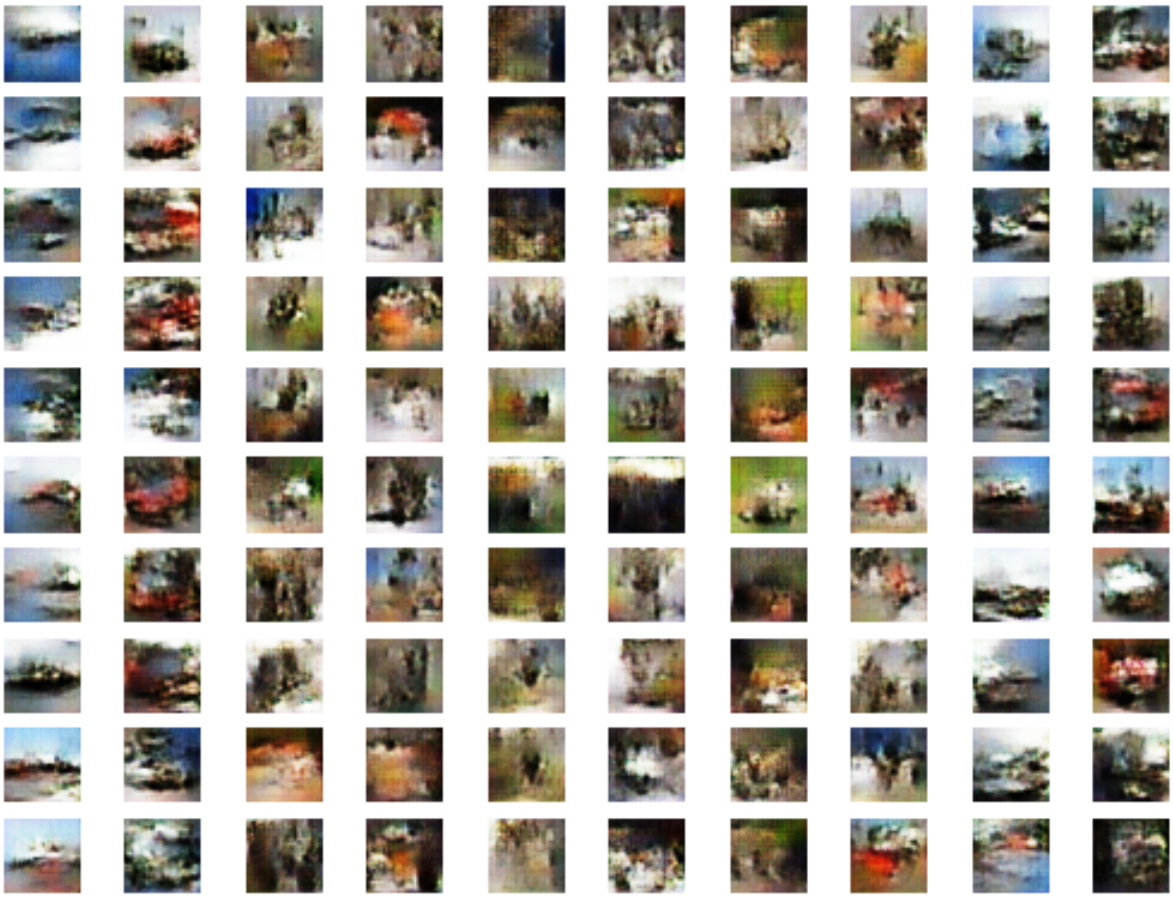


Abbildung 5.3: Ergebnisse nach 5 Epochen

Anschließend wurde die Batch-Size auf 64 erhöht und das Modell für 45 Epochen, siehe Abbildung 5.4, trainiert. Die generierten Bilder zeigen nun eine gewisse Ähnlichkeit mit den Originalbildern des CIFAR-10-Datensatzes. Sie weisen jedoch noch nicht die Feinheit der Details und die Genauigkeit der Farbwiedergabe der Originalbilder auf. Dies deutet darauf hin, dass weitere Verbesserungen des Trainings und der Modellarchitektur erforderlich sind, um eine höhere Qualität der erzeugten Bilder zu erreichen.

Die Analyse der generierten Bilder 5.4 zeigt, dass die Bilder von Flugzeugen und Schiffen überwiegend einen blauen Hintergrund und weiße Objekte haben. Im Gegensatz dazu haben die Bilder von Rehen und Pferden einen dunkelgrünen Hintergrund und braune Objekte. Diese Beobachtungen entsprechen den erwarteten Eigenschaften der jeweiligen Kategorien. Es ist ermutigend zu sehen, dass das Modell in der Lage ist, zumindest einige der charakteristischen Merkmale der Bilder korrekt zu erfassen und wiederzugeben.

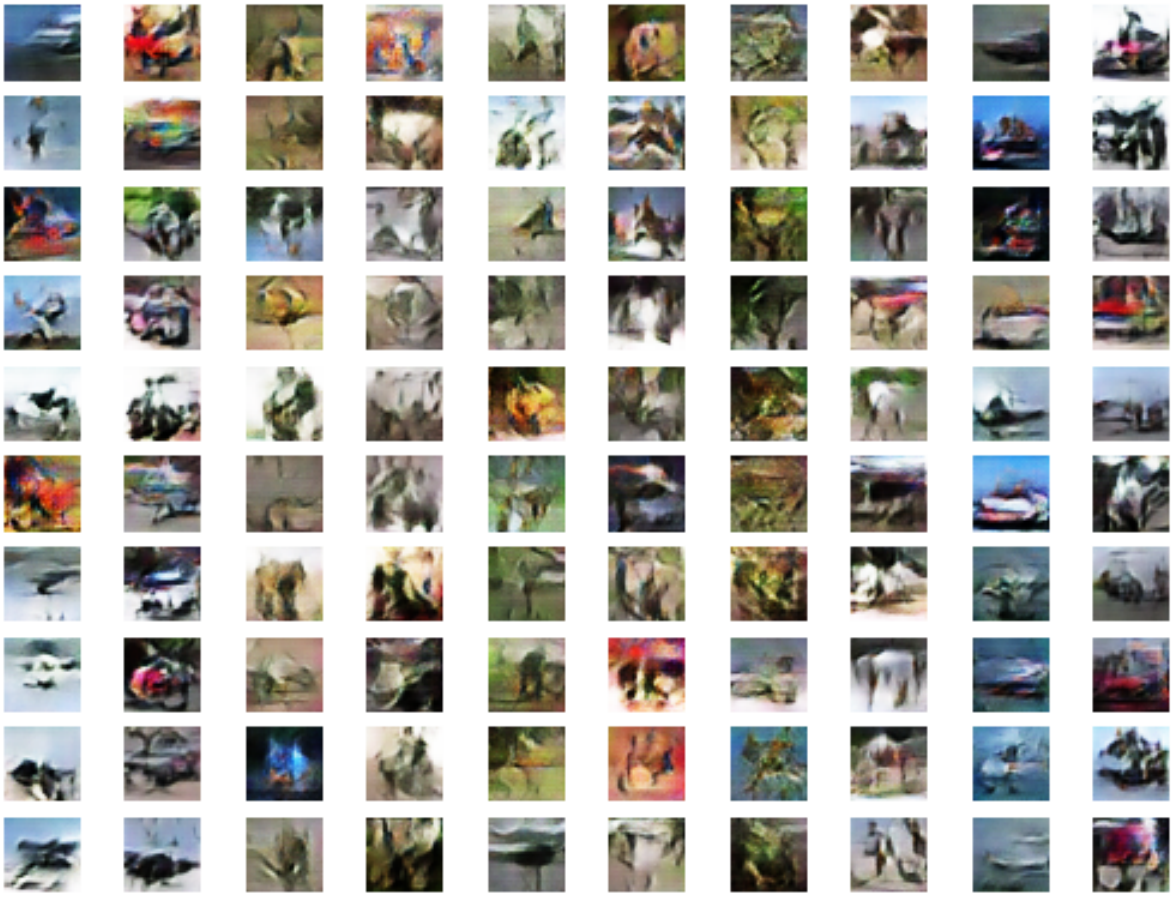


Abbildung 5.4: Ergebnisse nach 45 Epochen

Um bessere Ergebnisse bei der Bildgenerierung zu erzielen, wurde das Conditional GAN (cGAN) für weitere 75 Epochen trainiert, die Ergebnisse sind in Abbildung 5.5 zu sehen. Das Training über einen längeren Zeitraum führte zu einer erheblichen Verbesserung der Bildqualität. Die generierten Bilder zeichnen sich durch schärfere, lebendigere Farben und eine relative erhöhte Detailgenauigkeit aus. Die Abbildung 5.6 zeigt deutliche Verbesserungen bei einigen Kategorien wie Flugzeugen, Autos, Schiffen und Lastwagen. Diese Verbesserungen zeigen sich in einem deutlichen Kontrast zwischen Hintergrund und Vordergrund sowie in einer höheren Detailgenauigkeit. In anderen Kategorien wie Katzen, Hunde und Vögel sind die generierten Bilder aufgrund der geringeren Auflösung jedoch schwer zu erkennen. Die Komplexität der Inhalte in diesen Kategorien führt zu

deutlich schlechteren Ergebnissen. Es wird deutlich, dass die Generierung von qualitativ hochwertigen Bildern in Bezug auf Detailtreue und Erkennbarkeit eine Herausforderung darstellt, insbesondere bei komplexen Inhalten.

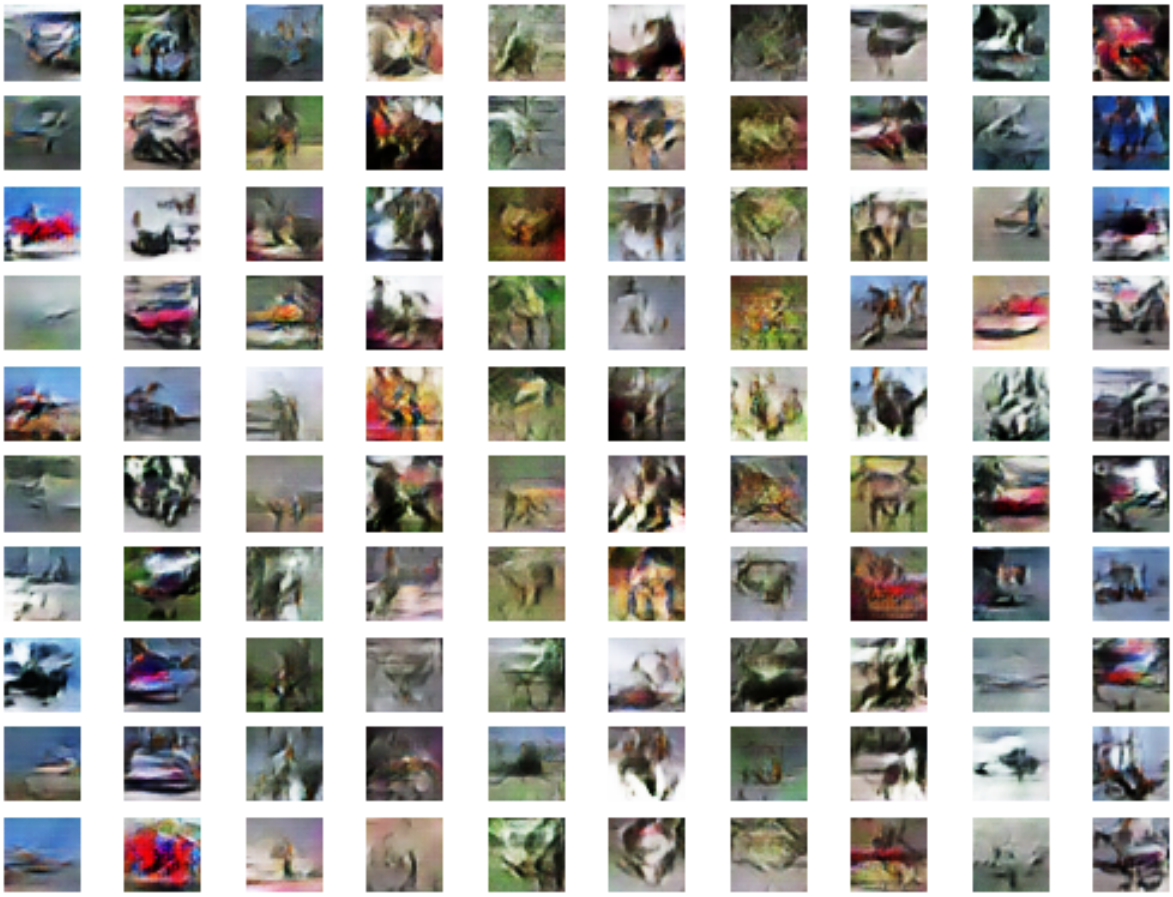


Abbildung 5.5: Ergebnisse nach 75 Epoche

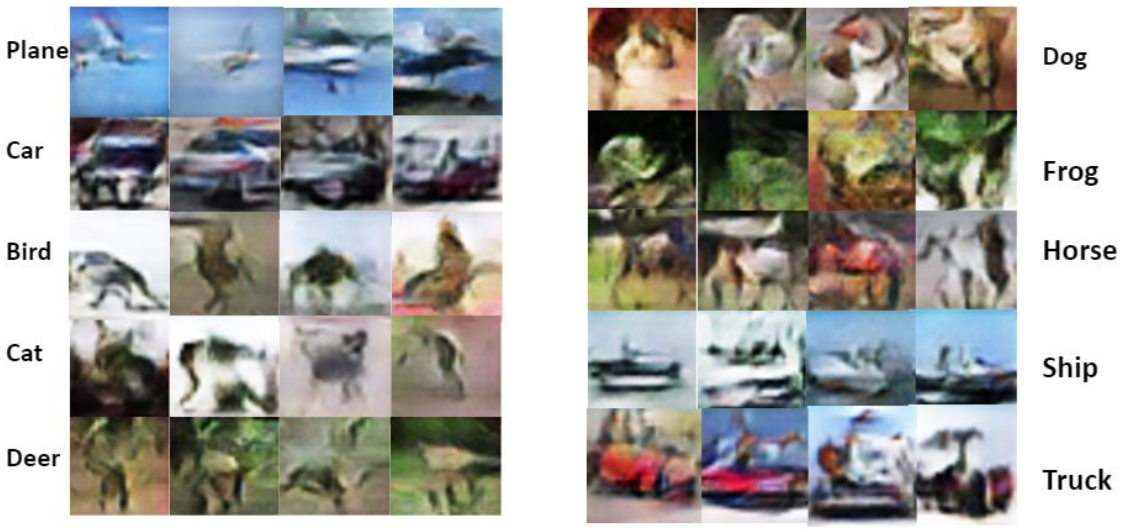


Abbildung 5.6: positive Ergebnisse

Abbildung 5.7 zeigt einige Beispiele für negative Ergebnisse. Obwohl diese Bilder nach 75 Epochen generiert wurden, enthalten sie nur begrenzte Informationen. Die generierten Bilder in diesen Kategorien erscheinen unscharf und weisen eine hohe Rauschintensität auf. Muster und Details sind schwer zu erkennen, was auf eine unzureichende Qualität und Genauigkeit der Generierung hinweist.

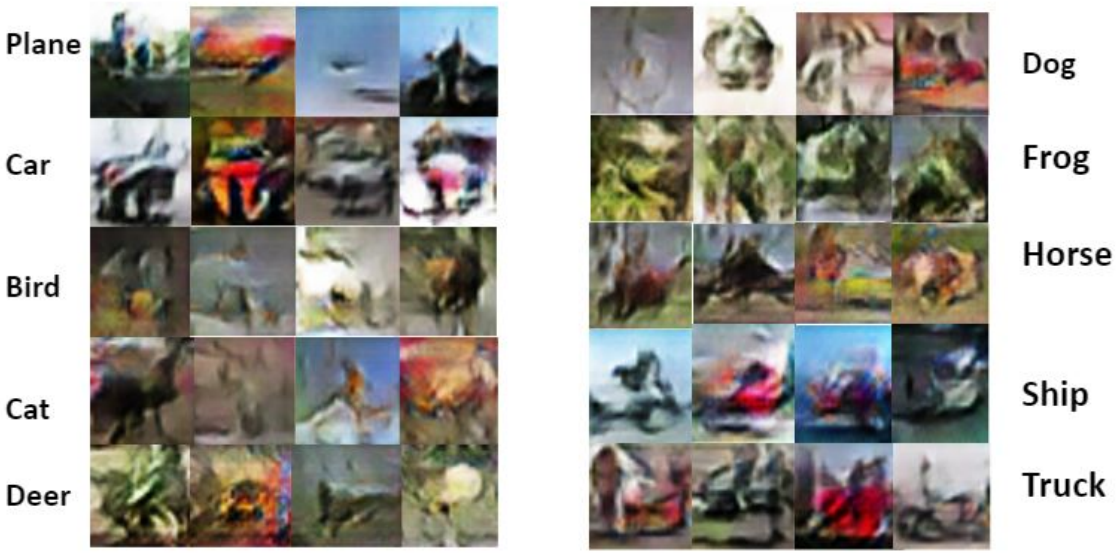


Abbildung 5.7: negative Ergebnisse

5.4.2 Evaluierung

Um die Qualität der generierten Bilder zu messen, werden wir uns auf zwei gängige Evaluierungsmetriken konzentrieren: JSD und FID. Die JSD misst die Abweichung zwischen der Verteilung der generierten Bilder und der Verteilung der realen Bilder. Eine niedrige JSD zeigt an, dass die generierten Bilder der Verteilung der realen Bilder nahe kommen. Der FID hingegen berechnet den Abstand zwischen den eingebetteten Merkmalen der generierten und realen Bilder. Ein niedriger FID bedeutet, dass die generierten Bilder ähnliche statistische Eigenschaften wie die realen Bilder aufweisen.

Für die Generierung von 500 Bildern mit unserem Modell haben wir die folgenden Ergebnisse erhalten: Die JSD (Jensen-Shannon Divergenz) beträgt 0,07118191921655176 und die FID (Frechet Inception Distance) beträgt 129,14386774959672. Diese Metriken dienen als Maß für die Qualität und Ähnlichkeit der generierten Bilder im Vergleich zu den realen Bildern.

Kritrium	JSD	FID
Cifar-10	0.07118	129.1438

Tabelle 5.1: Evaluieren von Cifar-10

6 Zusammenfassung

In diesem Kapitel haben wir uns mit Conditional GANs (cGANs) beschäftigt, einer Erweiterung von GANs, die es ermöglicht, gezielt bestimmte Daten zu generieren. Wir haben die Architektur von cGANs untersucht und einen Blick auf die Bildgenerierung mit dem Cifar-10-Datensatz geworfen.

Dennoch gibt es einige Herausforderungen und potenzielle Verbesserungen für cGANs. Die Auswahl und Darstellung der Konditionierungsinformationen kann einen erheblichen Einfluss auf die Qualität und Vielfalt der erzeugten Bilder haben.

Darüber hinaus gibt es Forschungsansätze zur Verbesserung der Trainingsstabilität von cGANs. Gradientenprobleme, Mode Collapse und das Verhältnis zwischen Generator- und Diskriminatorleistung sind Bereiche, in denen weitere Arbeiten erforderlich sind, um die Leistungsfähigkeit und Zuverlässigkeit von cGANs zu verbessern.

Insgesamt bieten cGANs eine interessante Möglichkeit zur gezielten Erzeugung von Bildern und zur Kontrolle bestimmter Merkmale. Die Forschung auf diesem Gebiet ist jedoch noch nicht abgeschlossen, und es besteht Raum für weitere Entwicklungen und Verbesserungen.

Literatur

[1] Martin Arjovsky, Soumith Chintala und Léon Bottou. *Wasserstein GAN*. 2017. arXiv: 1701.07875 [stat.ML].

[2] Yang-Jie Cao u. a. „Recent Advances of Generative Adversarial Networks in Computer Vision“. In: *IEEE Access* 7 (2019), S. 14985–15006. DOI: 10.1109/ACCESS.2018.2886814.

[3] Yang-Jie Cao u. a. „Recent Advances of Generative Adversarial Networks in Computer Vision“. In: *IEEE Access* 7 (2019), S. 14985–15006. DOI: 10.1109/ACCESS.2018.2886814.

[4] Keyang Cheng u. a. „An analysis of generative adversarial networks and variants for image synthesis on MNIST dataset“. In: *Multimedia Tools and Applications* 79 (Mai 2020). DOI: 10.1007/s11042-019-08600-2.

[5] Muthuraman Chidambaram und Yanjun Qi. *Style Transfer Generative Adversarial Networks: Learning to Play Chess Differently*. 2017. arXiv: 1702.06762 [cs.LG].

[6] Ian Goodfellow. *NIPS 2016 Tutorial: Generative Adversarial Networks*. 2017. arXiv: 1701.00160 [cs.LG].

[7] Ian J. Goodfellow u. a. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML].

[8] Martin Heusel u. a. *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium*. 2018. arXiv: 1706.08500 [cs.LG].

[9] Weiwei Hu und Ying Tan. *Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN*. 2017. arXiv: 1702.05983 [cs.LG].

[10] Tero Karras, Samuli Laine und Timo Aila. *A Style-Based Generator Architecture for Generative Adversarial Networks*. 2019. arXiv: 1812.04948 [cs.NE].

[11] Tero Karras u. a. *Progressive Growing of GANs for Improved Quality, Stability, and Variation*. 2018. arXiv: 1710.10196 [cs.NE].

[12] Alex Krizhevsky. „Learning Multiple Layers of Features from Tiny Images“. In: 2009.

[13] Jiwei Li u. a. *Adversarial Learning for Neural Dialogue Generation*. 2017. arXiv: 1701.06547 [cs.CL].

[14] Ziwei Liu u. a. „Deep Learning Face Attributes in the Wild“. In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dez. 2015.

[15] Mehdi Mirza und Simon Osindero. *Conditional Generative Adversarial Nets*. 2014. arXiv: 1411.1784 [cs.LG].

[16] Alec Radford, Luke Metz und Soumith Chintala. *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*. 2016. arXiv: 1511.06434 [cs.LG].

[17] Kunfeng Wang u. a. „Generative adversarial networks: introduction and outlook“. In: *IEEE/CAA Journal of Automatica Sinica* 4.4 (2017), S. 588–598. DOI: 10.1109/JAS.2017.7510583.

[18] Maciej Wiatrak, Stefano V. Albrecht und Andrew Nystrom. *Stabilizing Generative Adversarial Networks: A Survey*. 2020. arXiv: 1910.00927 [cs.LG].

[19] Lantao Yu u. a. *SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient*. 2017. arXiv: 1609.05473 [cs.LG].