Generierung künstlicher Trainingsdaten für die Straßenschilderkennung in Fahrzeugen mittels Generative Adversarial Networks

Studienarbeit

Studiengang Informatik

an der Dualen Hochschule Baden-Württemberg Stuttgart

von

Frederik Esau

08.06.2023

Bearbeitungszeitraum	24.10.2022 - 08.06.2023
Matrikelnummer, Kurs	6526552, TINF20ITA
Betreuer	Prof. Dr. Monika Kochanowski

Erklärung

Ich versichere hiermit, dass ich meine Studienarbeit mit dem Thema: *Generierung künstlicher Trainingsdaten für die Straßenschilderkennung in Fahrzeugen mittels Generative Adversarial Networks* selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Stuttgart, 08.06.2023

Frederik Esau

Abstract

Abstract normalerweise auf Englisch. Siehe: http://www.dhbw.de/fileadmin/user/public/Dokumente/Portal/Richtlinien_Praxismodule_Studien_und_Bachelorarbeiten_JG2011ff.pdf (8.3.1 Inhaltsverzeichnis)

Ein „Abstract“ ist eine prägnante Inhaltsangabe, ein Abriss ohne Interpretation und Wertung einer wissenschaftlichen Arbeit. In DIN 1426 wird das (oder auch der) Abstract als Kurzreferat zur Inhaltsangabe beschrieben.

Objektivität soll sich jeder persönlichen Wertung enthalten

Kürze soll so kurz wie möglich sein

Genauigkeit soll genau die Inhalte und die Meinung der Originalarbeit wiedergeben

Üblicherweise müssen wissenschaftliche Artikel einen Abstract enthalten, typischerweise von 100-150 Wörtern, ohne Bilder und Literaturzitate und in einem Absatz.

Quelle: http://de.wikipedia.org/wiki/Abstract Abgerufen 07.07.2011

Diese etwa einseitige Zusammenfassung soll es dem Leser ermöglichen, Inhalt der Arbeit und Vorgehensweise des Autors rasch zu überblicken. Gegenstand des Abstract sind insbesondere

- Problemstellung der Arbeit,
- im Rahmen der Arbeit geprüfte Hypothesen bzw. beantwortete Fragen,
- der Analyse zugrunde liegende Methode,
- wesentliche, im Rahmen der Arbeit gewonnene Erkenntnisse,
- Einschränkungen des Gültigkeitsbereichs (der Erkenntnisse) sowie nicht beantwortete Fragen.

Quelle: http://www.ib.dhbw-mannheim.de/fileadmin/ms/bwl-ib/Downloads_alt/Leitfaden_31.05.pdf, S. 49

Inhaltsverzeichnis

Abkürzungsverzeichnis

CycleGAN	Cycle-Consistent GAN
GAN	Generative Adversarial Network
GTSRB	German Traffic Sign Recognition Benchmark
KNN	Künstliches Neuronales Netz

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1 Einleitung

Während in großen Teilen des letzten Jahrhunderts Innovationen in der Fahrzeugentwicklung vor allem im Bereich der mechanischen Leistung und Effizienz stattgefunden haben, erwartet man zukünftige Verbesserungen im Automobil besonders softwareseitig [1]. ...

1.1 Problemstellung

1.2 Vorgehensweise

1.3 Ziel der Arbeit

2 Stand der Technik

2.1 Momentane Lösungen zur Straßenschilderkennung

2.2 Künstliche Neuronale Netze

2.2.1 Convolutional Neural Networks

2.3 Bildgenerierung

2.4 Generative Adversarial Networks

Generative Adversarial Networks (GANs) sind eine der eingesetzten Methoden, um Algorithmen zu entwickeln, die neue Daten künstlich generieren können. Dazu zählen zum Beispiel Bilder von Menschen, die zwar realistisch erscheinen, die so aber in der Realität nicht existieren. Die Aufgabe eines GANs ist demnach, Daten zu generieren, die denen des definierten Trainingssatzes in einer solchen Weise ähneln, dass nicht mehr unterschieden werden kann, ob ein Bild künstlich generiert ist oder aus dem Trainingssatz stammt. In dem genannten Beispiel besteht der Trainingssatz aus echten Bildern von Menschen, durch die das GAN lernt, neue Bilder zu erzeugen. Im weiteren Verlauf wird das Thema GANs ausschließlich auf die Bilderzeugung bezogen, auch wenn ebenso andere Anwendungsgebiete existieren. [2]

Ein GAN besteht aus zwei Komponenten. Dem sogenannten *Generator* und dem *Discriminator*. Der Generator erzeugt neue Bilder, während der Discriminator für jedes erzeugte Bild rät, ob es aus dem Trainingssatz stammt, oder ob es künstlich generiert ist. Nur der Discriminator hat dabei Zugriff auf den Trainingsdatensatz. Ihm werden klassischerweise abwechselnd Bilder aus dem Trainingssatz und erzeugte Bilder des Generators gezeigt. Die beiden Komponenten des GANs agieren dabei stets gegeinander. Der Generator versucht den Discriminator in die Irre zu führen, dass seine generierten Bilder in Wahrheit aus dem Trainingssatz stammen würden, während der Discriminator versucht, möglichst gut zu erkennen, ob ein Bild echt ist oder nicht. Bei beiden Komponenten handelt es sich dabei um künstliche neuronale Netze (KNNs). [2]

Während die Bildklassifizierung ein Minimierungsproblem ist, stellt die Bildgenerierung mittels GANs ein Min-Max-Problem dar. Bei ersterem wird versucht, die Loss-Function zu minimieren,

sodass die Predictions möglichst gut den Labels der Daten entsprechen. Bei GANs besteht der Unterschied darin, dass einerseits der Generator die Loss-Function zu minimieren versucht, während andererseits der Discriminator sie zu maximieren versucht.

$$\min_G \max_D V(D,G) = E_x[\log(D(x))] + E_z[\log(1 - D(G(z)))] \tag{2.1}$$

2.4.1 Spieltheorie

Aus spieltheoretischer Sicht kann das Gegenspiel von Generator und Discriminator auch als *Nullsummenspiel* betrachtet werden.

2.4.2 Training

Zu Beginn sind sowohl der Generator als auch der Discriminator nicht gut geeignet für ihre jeweiligen Aufgaben. Der Generator erzeugt Bilder, die denen des Trainingssatzes nicht ähnlich sind, während der Discriminator noch nicht weiß, was die Bilder des Trainingssatzes einzigartig macht. Durch das Training verbessern sich beide Komponenten in ihren Aufgaben. Sie trainieren sich gegenseitig, da sie beide versuchen, das Spiel zu gewinnen.

Das finale Optimum ist, dass der Discriminator so gut in der Unterscheidung zwischen echten und künstlichen Daten wird, wie mit den vorhandenen Daten nur möglich, während der Generator trotzdem in der Lage sein soll, den Discriminator zu überlisten. [2, S. 656]

Bild ist nicht 100% korrekt! G wird z.B. nicht trainiert, wenn D richtig lag, aber das Bild aus dem Datensatz stammte.

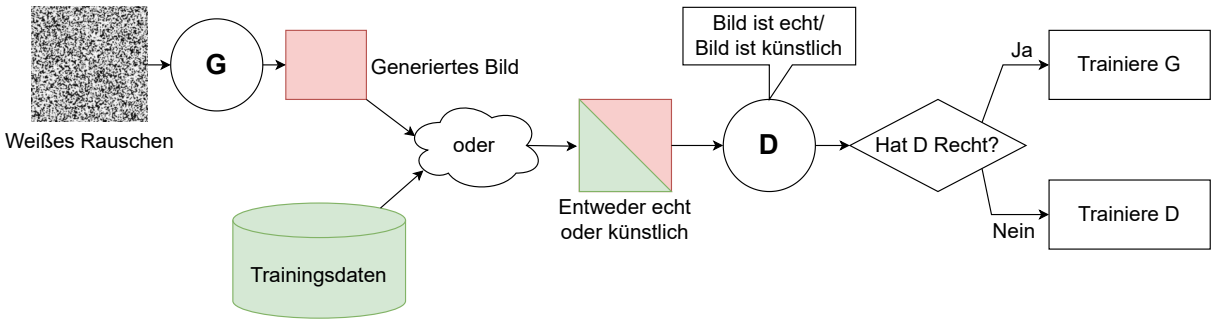

Abbildung 2.1: Trainingsprozess von GANs (eigene Darstellung, angelehnt an [2, S. 654f.])

(Auf Emfindlichkeit von GANs für Hyperparameter eingehen)

2.4.3 Architekturen

Die bisher beschriebene Architektur von GANs wird auch als *Vanilla GAN* bezeichnet. Dies entspricht dem, wie GANs in Goodfellows Publikation definiert werden [3]. Forscher und Anwender haben seit dieser Veröffentlichung verschiedene Limitationen und Probleme bei Vanilla GANs feststellen können. Insbesondere im Hinblick auf spezielle Einsatzgebiete. Ein hier häufig anzutreffender Begriff ist *Modal Collaps*. Damit ist die Situation gemeint, dass der Generator bei beliebigem Input stets dasselbe Bild generiert. Er lernt, dass ein bestimmtes Bild den Discriminator überlisten kann und generiert es deshalb jedes Mal, egal welchen Input man ihm zuführt. Lösen lässt sich dieses Problem beispielsweise mit sogenannten Cycle-Consistent GANs (CycleGANs).

CycleGANs Ein CycleGAN besteht aus zwei miteinander gekoppelten GANs. Diese werden klassischerweise als G und F bezeichnet. Bei G handelt es sich um ein Vanilla GAN, so wie in diesem Kapitel bisher beschrieben. Erweitert wird das Netzwerk jedoch so, dass es den Output von G an F weitergibt. F erhält somit das von G künstlich generierte Bild als Input. Aufgabe von F ist, daraus den ursprünglichen Input, der G zugeführt wurde, zu reproduzieren. Somit soll das gesamte Netzwerk nicht nur aus einem Input einen gewissen Output generieren können, sondern soll auch von einem gegebenen Output zurück auf den Input schließen können. Dies lässt sich mathematisch so darstellen, dass G und F folgende Abbildungen implementieren:

$$\mathbf{G} : X \mapsto Y \ \wedge \ \mathbf{F} : Y \mapsto X \tag{2.2}$$

Das Modell G erzeugt somit aus einem gegebenen X ein Y, wohingegen F aus dem Y auf das X schließen soll. Die Behebung des Modal Collaps findet dadurch statt, dass das Netzwerk den Output $\tilde{X}$ von F überprüft und diesen mit dem tatsächlichen Input X vergleicht. Es wird überprüft, wie ähnlich sich $\tilde{X}$ und X sind. Liegt eine zu hohe Diskrepanz vor, kann das Netzwerk darauf schließen, dass G Outputs erzeugt, die nicht in direkter Abhängigkeit zu X stehen.

Was deutlich wird, ist dass hierbei die Idee verworfen werden muss, G ein weißes Rauschen als Input zuzuführen. CycleGANs sind für spezielle Anwendungsgebiete gedacht, in denen ausgewählte, und somit nicht-zufällige Eingabebilder verwendet werden. Dazu zählen Gebiete wie *Style Transfer* oder die Transformation von Bildern, respektive Bildelementen. Es wird sich im Verlauf der Arbeit zeigen, dass die Problemstellung dieser Studienarbeit ein solches Anwendungsgebiet darstellt.

$$L_{GAN}(G, D_Y, X, Y) = E_y[\log D_Y(y)] + E_x[\log(1 - D_Y(G(x)))] \tag{2.3}$$

$$L_{cyc}(G, F) = E_x[||F(G(x)) - x||_1] + E_y[||G(F(y)) - y||_1] \tag{2.4}$$

$$L(G, F, D_X, D_Y) = L_{GAN}(G, D_Y, X, Y) + L_{GAN}(F, D_X, Y, X) + \lambda L_{cyc}(G, F) \tag{2.5}$$

[4]

Wasserstein GANs [3] [2]

2.5 Vorherige Arbeiten

2.5.1 Verwendung des gleichen Datensatzes

[5]

2.5.2 Verwendung anderer Datensätze

[6]

2.6 Machine Learning Frameworks

2.6.1 Tensorflow

2.6.2 Pytorch

2.6.3 Weitere

3 Konzeption des Generative Adversarial Networks

German Traffic Sign Recognition Benchmark (GTSRB)

(a) (b) (c) (d)

Abbildung 3.1: Beispielbilder aus dem GTSRB Datensatz [7]

[7]

3.1 Datensatz

3.2 Architektur

3.3 Generator

3.4 Discriminator

3.5 Training

4 Implementierung und Training

4.1 Wahl eines Frameworks

Tensorflow, Pytorch

5 Evaluation

6 Zusammenfassung

Literatur

[1] M. Staron, „AUTOSAR (AUTomotive Open System ARchitecture),“ in *Automotive Software Architectures: An Introduction*. Cham: Springer International Publishing, 2021, 97ff. ISBN: 978-3-030-65939-4. DOI: 10.1007/978-3-030-65939-4_5.

[2] A. S. Glassner, „Deep Learning: A Visual Approach,“ in San Francisco: No Starch Press, 2021, Kap. Generative Adversarial Networks, S. 649–673, ISBN: 978-1-7185-0072-3.

[3] I. Goodfellow, Y. Bengio und A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[4] J.-Y. Zhu, T. Park, P. Isola und A. A. Efros, *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*, 2017. DOI: 10.48550/ARXIV.1703.10593.

[5] D. Spata, D. Horn und S. Houben, „Generation of Natural Traffic Sign Images Using Domain Translation with Cycle-Consistent Generative Adversarial Networks,“ in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, S. 702–708. DOI: 10.1109/IVS.2019.8814090.

[6] D. Christine et al., „Synthetic Data generation using DCGAN for improved traffic sign recognition,“ *Neural Computing and Applications*, S. 1–16, Apr. 2021. DOI: 10.1007/s00521-021-05982-z.

[7] J. Stallkamp et al., „The German Traffic Sign Recognition Benchmark: A multi-class classification competition,“ in *EEE International Joint Conference on Neural Networks*, 2011, S. 1453–1460.

Anhang

A. test

B. test

C. test