

# **Erstellung eines Trainingsdatensatzes für object detection mit echten und synthetischen Bildern**

Tim Tomczek  
Fachhochschule Südwestfalen

Schriftliche Ausarbeitung im Modul  
„Projekt KI“

5. Juli 2025

# **Inhaltsverzeichnis**

<b>1. Einleitung</b>	<b>3</b>
1.1. Projekthintergrund . . . . .	3
1.2. Projektbeschreibung und Ziele . . . . .	3
1.3. Zeitplanung . . . . .	4
<b>2. Durchführung</b>	<b>5</b>
2.1. Recherche und Vergleich von Modellen . . . . .	5
2.2. Kurzvorstellung der Modelle . . . . .	5
2.3. Erstellung der Testumgebung . . . . .	7
2.4. Vergleich der Modelle . . . . .	8
2.5. Erstellung des Datensatzes . . . . .	11
2.5.1. Sammeln von echten Bildern . . . . .	11
2.5.2. Generierung von synthetischen Bildern . . . . .	12
2.5.3. Annotation der Bilder . . . . .	14
<b>3. Projektabschluss</b>	<b>17</b>

# **1. Einleitung**

Diese Arbeit entstand im Rahmen des Moduls Projekt KI des Verbundstudiengangs Angewandte Informatik an der Fachhochschule Südwestfalen Standort Iserlohn. Sie beschäftigt sich mit der Erstellung eines Bilddatensatzes für die Objekterkennung mit echten und synthetischen Bildern. Nachfolgend wird die Vorgehensweise bei der Erstellung des Datensatzes beschrieben. Dabei wird zuerst auf die Auswahl der Modelle und die Testumgebung für die Auswahl eines Modells eingegangen. Anschließend wird die Erstellung des Datensatzes beschrieben. Dabei wird zuerst auf die Auswahl der echten Bilder und anschließend auf die Erstellung der synthetischen Bilder eingegangen, bevor dann die Annotation der Bilder beschrieben wird. Zum Schluss werden die Ergebnisse des Projekts zusammengefasst.

## **1.1. Projekthintergrund**

Dieses Projekt soll als Vorarbeit für die nachfolgende Bachelorarbeit dienen. In dieser soll ein KI-Modell zur Objekterkennung um eine weitere Klasse erweitert werden. Als Grundlage soll der COCO Datensatz verwendet werden [1]. Dabei soll untersucht werden wie gut die Erkennung anhand von echten Bildern und synthetischen (mit einem KI-Modell generierte Bilder) ist.

## **1.2. Projektbeschreibung und Ziele**

Ziel dieses Projektes ist es einen Datensatz aus KI generierten (synthetischen) und echten Bildern zu erstellen und diese mit Annotationen zu versehen. Der erstellte Datensatz soll für den Vergleich der Leistung von Objekterkennungsmodellen verwendet werden. Verglichen werden soll die Erkennungsleistung von Modellen zur Objekterkennung an einer neuen Kategorie von Objekten anhand von echten, synthetischen und gemischten Bildern des, in diesem Projekt erstellten, Datensatzes. Als Basis werden Modelle verwendet die auf dem COCO Datensatz [1] trainiert wurden. Dieser umfasst 80 Objektkategorien. Eine Kategorie die darin nicht enthalten ist, sind Musikinstrumente. Aufgrund des zeitlichen Rahmens des Projektes wird die Kategorie dabei auf ein Instrument beschränkt. Da für den COCO Datensatz kein entsprechender Datensatz existiert soll dieser in diesem Projekt erstellt werden.

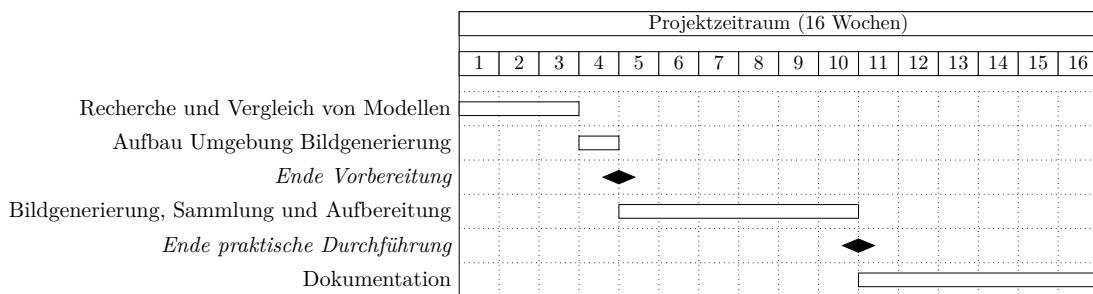
Der Fokus in diesem Projekt liegt dabei hauptsächlich auf der Erstellung des synthetischen Teils des Datensatzes. Für die Erstellung dieses Teils wird ein lokales (FH Cluster oder lokaler Computer) Text-zu-Bild Modell verwendet, welches die Bilder generieren soll. Zuerst wird nach aktuellen Modellen für die Bildgenerierung recherchiert. Danach

werden zwei bis drei der Modelle miteinander verglichen, indem, mit der gleichen Texteingabe, einige Bilder erstellt und verglichen werden. Das beste Modell wird dann weiter verwendet. Als Quelle für Modelle kann zum Beispiel Hugging Face [2] dienen. Die zur Bildgenerierung verwendete Umgebung, das Modell und die Texteingabe werden dabei erläutert. Anschließend werden die Bilder mit einer Annotationssoftware aufbereitet. Für die Aufbereitung der Bilder wäre der Einsatz von CVAT [3] oder Label Studio [4] möglich. Beide Programme sind kostenlos und lokal einsetzbar. Hierbei wird erläutert wie die Software bei der Annotation unterstützt und worauf zu achten ist, um einen hochwertigen Datensatz zu erstellen.

### 1.3. Zeitplanung

Das Gantt-Diagramm 1.3 zeigt die Zeitplanung für dieses Projekt. Die Planung enthält die verschiedenen Phasen und Meilensteine. Bei der Planung der Dauer der einzelnen Phasen wurde berücksichtigt, dass auch weitere Projekte anderer Module durchgeführt werden. Die Phasen wurden entsprechend länger angesetzt.

Die erste Phase Recherche und Vergleich von Modellen umfasst die Recherche nach aktuellen Bildgenerierungsmodellen und deren Evaluation anhand von testweise erstellten Bildern mit den Modellen. In der zweiten Phase Aufbau Umgebung Bildgenerierung wird das zuvor ausgewählte Modell und entsprechende Infrastruktur für die Erstellung von einer großen Anzahl an Bildern erstellt. An dieser Stelle ist der Meilenstein Ende Vorbereitung erreicht und die Erstellung der Bilder kann beginnen. Für die Phase Bildgenerierung und Aufbereitung ist ein Zeitraum von sechs Wochen vorgesehen. In dieser Zeit sollen so viele Bilder wie möglich erstellt und gelabelt werden. Es wird dabei eine Anzahl von 100 Bildern anvisiert. Nach dieser Phase ist der zweite Meilenstein erreicht und es wird in der letzten Phase mit der Dokumentation begonnen.



## **2. Durchführung**

Den Start des Projekts bildet das Kickoff-Meeting mit dem Projektbetreuer und dem Projektdurchführenden am 29.03.2025. In diesem Meeting wurde das Projekt vorgestellt und die Ziele sowie der Zeitplan besprochen. Anschließend wurde mit der Abgabe des Projektantrags das Projekt offiziell angemeldet.

### **2.1. Recherche und Vergleich von Modellen**

Das Projekt beginnt mit der Recherche nach aktuellen KI-Modellen, die für die Generierung von Bildern geeignet sind. Dafür wurde zuerst nach entsprechend Benchmarks für die Vergleich von Modellen gesucht. Ein solcher Benchmark hätte es ermöglicht, die Modelle direkt miteinander zu vergleichen und die besten Modelle für das Projekt auszuwählen. Leider gibt es keinen allgemeingültigen Benchmark, der für alle Modelle geeignet ist. Daher wurde beschlossen, die vier obersten Modelle auf der Hugging Face Website unter der Kategorie Text-To-Image, nach der Sortierung Trending, zu vergleichen. Dabei wurden verschiedene Versionen des gleichen Modells nicht berücksichtigt und jeweils das größte Modell ausgewählt. Fine-Tunings von Modellen wurden auch aus der Auswahl ausgeschlossen um einen möglichst repräsentativen Vergleich von verschiedenen Architekturen zu erhalten. Daraus entstand am 17.04.2025 die folgende Liste von Modellen, die für den Vergleich verwendet werden:

- HiDream-ai/HiDream-I1-Full
- black-forest-labs/FLUX.1-dev
- stabilityai/stable-diffusion-3.5-large
- stabilityai/stable-diffusion-xl-base-1.0

Diese Liste stellt die Modelle in der Reihenfolge dar, in der sie nach der Trending Sortierung angezeigt wurden.

### **2.2. Kurzvorstellung der Modelle**

HiDream-I1-Full ist das neueste der vier Modelle es wurde Anfang April 2025 von Vivago AI als Open-Source Modell veröffentlicht. Es ist in den Varianten HiDream-I1-Full, HiDream-I1-Dev, HiDream-I1-Fast als Text-zu-Bild oder als HiDream-E1 für die Bearbeitung von Bildern verfügbar. Nach eigenen Angaben im entsprechenden Paper [5, Abstract] wurden neueste Fortschritte bei der Verbesserung der Qualität von Bildern durch eine Steigerung der Rechenkomplexität erzielt. Dies führt zu besseren Ergebnissen aber auch zu längeren Inferenzzeiten. Um diesem Problem entgegenzuwirken, wurde

das HiDream-I Modell entwickelt um hochwertige Ergebnisse in kurzer Zeit zu liefern. Um dies zu erreichen wurde eine neue Architektur entwickelt, der *Sparse Diffusion Transformer*. Das Modell verwendet zunächst drei Encoder um die Texteingabe zu verarbeiten. Anschließend folgt ein *Dual-stream Diffusion Transformer* (DiT) Block der die Texteingabe und die Bilddaten getrennt verarbeitet. Diese beiden Streams werden dann in einem *Single-stream* DiT Block zusammengeführt. Beide Blöcke verwenden ein *sparse Mixture-of-Experts* (MoE) Konzept um gezielt Teile des Bildes zu verarbeiten. Die Experts werden dabei von einen *dense feed-forward network* (FNN), welches als Router fungiert, gespeist. Durch die gezielte, spärliche, Aktivierung der Experts lässt sich die benötigte Rechenleistung reduzieren. Das vollständige Modell besitzt 17 Milliarden Parameter und ist damit das größte Modell in dieser Auswahl [5, S. 7-9].

Das Flux.1 Modell wurde im August 2024 von Black Forest Labs (BFL), eines in Freiburg im Breisgau gegründeten Start-Ups, veröffentlicht. Es ist in den drei Varianten FLUX.1-pro, FLUX.1-dev und FLUX.1-schnell verfügbar, wobei das FLUX.1-pro Modell nicht frei verfügbar ist. Das Modell basiert nach eigenen Angaben von BFL auf ihrer Webseite aus “einer hybriden Architektur aus multimodal und parallel diffusion Transformer Blöcken die auf 12 Milliarden Parameter skaliert wurden. Die Architektur wird durch flow Matching weiter verbessert. Die Performance und Hardwareeffizienz wird durch den Einsatz von rotary positional embeddings und parallel attention layers optimiert” [6].

Das Stable Diffusion 3.5 stammt von Stability AI und wurde im Oktober 2024 veröffentlicht. Es ist in den Varianten Large, Large Turbo und Medium verfügbar. Diese dürfen für die Forschung, für nicht-kommerzielle und kommerzielle bis zu einem Jahresumsatz von einer Million US Dollar frei verwendet werden. Die in diesem Projekt verwendete Large Variante hat 8.1 Milliarden Parameter. Dieses Modell setzt auf eine multimodal Diffusion Transformer Architektur (MMDiT). Dies erlaubt es entgegen anderen Architekturen, wie zum Beispiel der einfachen Diffusion Transformer Architektur, dass eine Zwei-Wege Kommunikation zwischen den Text und Bild Daten stattfindet. Dies erlaubt eine bessere Text-zu-Bild Generierung. Trainiert wurde das Modell auf dem ImageNet und CC12M Datensatz und wurde gegen den COCO-2014 Validierungssatz getestet [7, S. 4-5].

Das Stable Diffusion XL (SDXL) Modell stammt auch von Stability AI und wurde im Juli 2023 veröffentlicht. Es basiert auf Stable Diffusion 2.1 ist aber mit 3,5 Milliarden Parameter eine dreimal größer als sein Vorgänger. Dies liegt an einem größeren attention block und einem größeren cross-attention context durch einen zweiten Text Encoder. Genau wie der Vorgänger Stable Diffusion 2.1 basiert es auf einem Latent Diffusion Model mit einem UNET zur Reduzierung des Rauschens in den Bildern [8, Abstract]. Die Besonderheit bei diesem Modell besteht darin, dass es sich hierbei um ein Assemble of Experts Modell handelt. Dabei handelt es sich um die zwei Modelle Stable Diffusion XL Base und Stable Diffusion XL Refiner, die zusammen verwendet werden. Das Base

Modell generiert dabei ein erstes Bild, welches dann vom Refiner Modell, welches auf die Erstellung von Details angepasst wurde, weiter verfeinert wird. Dadurch soll eine höhere Qualität der Bilder erreicht werden [8, S. 2].

### 2.3. Erstellung der Testumgebung

Für den Vergleich der Modelle wurde eine Testumgebung in einem Jupyter Notebook erstellt. Der Code ist in der beiliegenden Datei `evaluation-of-image-model.ipynb` zu finden. Für die Ausführung der Modelle mit der Hugging Face diffusers-Bibliothek [9] muss momentan (Stand 26.05.2025) die Bibliothek aus dem diffusers GitHub Repository installiert werden, da die aktuellste veröffentlichte Version der Bibliothek (v0.33.1 vom 10.04.2025) nicht die Pipeline für das HiDream Modell enthält. Zudem wird ein Hugging Face Token benötigt, um auf die Tokenizer und Encoder, des von HiDream benötigten, `meta-llama/Meta-Llama-3.1-8B-Instruct` Modells zugreifen zu können. Für eine einfache Ausführung der Modelle wurde eine Funktion erstellt, die eine Modell Pipeline und einige Parameter entgegennimmt und dann Bilder generiert und diese in einem Verzeichnis abspeichert. Die Funktion generiert für jede Texteingabe drei Bilder, wobei der *guidance\_scale* um den, vom Modell empfohlenen Wert, plus eins und minus eins variiert wird. Die Anzahl der Schritte wird der Funktion beim Aufruf mitgegeben und entspricht der vom Modell empfohlenen Anzahl. Die definierte Funktion wird dann nach dem Laden der Modelle über die jeweiligen Pipelines aufgerufen.

Die Tests der Modelle wurden auf dem JupyterHub der Fachhochschule Südwestfalen durchgeführt. Dort werden verschiedene Python-Umgebungen bereitgestellt und bieten Zugriff auf verschiedene GPU Konfigurationen mit Nvidia A100 Grafikkarten mit 80 GB VRAM. Die zuerst verwendete Konfiguration bietet einen geteilten Zugriff auf 4 A100 Grafikkarten. Dies führte zu Problemen bei der Ausführung der Modelle, da nicht ausreichend VRAM für die Modelle, insbesondere für das HiDream Modell, zur Verfügung stand. Daher wurde für das weitere Vorgehen eine Konfiguration mit einer dedizierten A100 Grafikkarte mit 40 GB VRAM gewählt. Diese Konfiguration bot für einen ausreichenden Zeitraum Zugriff auf ausreichend VRAM für die meisten Modelle. Lediglich das HiDream Modell stieß weiterhin auf Probleme mit dem Speicher. Diese konnten durch die Aktivierung von zwei Flags in der Pipeline behoben werden. Die Flags waren `enable_model_cpu_offload` und `enable_vae_tiling`. Das Flag `enable_model_cpu_offload` sorgt dafür, dass das Modell nicht vollständig im VRAM gehalten wird, sondern im RAM und Submodule bei Bedarf in den VRAM geladen werden. Dadurch lässt sich der benötigte GPU-Speicher reduzieren, zum Preis einer längeren Ausführzeit, da die Module zwischen Speichern hin und her geladen werden müssen. Das Flag `enable_vae_tiling` sorgt dafür, dass der VAE (Variational Autoencoder) des Modell das Bild in sich überlappenden

Kacheln verarbeitet und diese am Ende wieder zusammensetzt. Dadurch kann der benötigte Speicher für die Verarbeitung für größere Bilder reduziert werden [10].

Als Texteingaben für die Modelle wurden zwei verschiedene Eingaben gewählt. Diese sind:

- a hyperrealistic photo of a group of musicians playing various instruments in a band, set in a random location
- a hyperrealistic photo of a common living room with a variety of music instrument scattered around the room

Diese beiden Texteingaben wurden gewählt um ein möglichst breites Spektrum an Instrumenten und Szenen zu generieren. Das Ziel dabei war es ein Modell auszuwählen, dass mit einer möglichst allgemeinen Texteingabe möglichst viele verschiedene Instrumente in verschiedenen Szenen generieren kann. Mit dem Zusatz *hyperrealistic photo* soll dabei sichergestellt werden, dass die generierten Bilder möglichst realistisch aussehen. Mit diesem Testaufbau wurden dann insgesamt 24 Bilder erstellt.

## 2.4. Vergleich der Modelle

Für die Auswahl des Modells wurden verschiedenen Kriterien festgelegt, die für ein Modell welches den Anforderungen des Projekts gerecht werden soll, wichtig sind.

Das erste Kriterium ist die Anzahl der Fehler in den Bild. Für dieses Kriterium wurden die generierten Bilder durch den Autor auf visuelle Fehler untersucht. Als visueller Fehler wird dabei eine Abweichung von einem realen Bild der gleichen Szene definiert. Dies können Artefakte, falsche Anatomie von Personen oder Fehler in der Darstellung der Instrumente sein.

Das zweite Kriterium ist die Diversität der dargestellten Instrumente. Da in der nachfolgenden Bachelorarbeit ein Modell für Objekterkennung von Gitarren trainiert werden soll, ist es wichtig, dass das Modell verschiedene Gitarren darstellen kann, aber auch andere, ähnliche, Instrumente wie Geigen, Cellos, oder Banjos. Die Diversität wird dabei als Anzahl unterschiedlicher Instrumente und deren Variationen in den generierten Bildern definiert. Dafür wird die Anzahl der verschiedenen Instrumente in den generierten Bildern gezählt. Eine Variation eines Instruments wird mit einem halben Punkt bewertet und ein neues Instrument mit einem Punkt. Die Punkte werden dann addiert und ergeben die Diversität der Instrumente in den generierten Bildern.

Das dritte Kriterium ist die Diversität der Szene. Diese wird ähnlich wie die Diversität der Instrumente bewertet. Pro Modell werden Bilder mit zwei Eingaben generiert. Die Modelle verwenden dabei die empfohlenen Parameter, wobei pro Eingabe drei Bilder erstellt werden. Pro Bild variiert der *guidance\_scale* um den, vom Modell, empfohlenen



Abbildung 1: Eine Auswahl an markierten Bildern für die Bewertung der Modelle. Von links nach rechts: HiDream-I1-Full, FLUX.1-dev, Stable Diffusin 3.5 Large, Stable Diffusion XL Refiner.

Wert plus zwei und minus zwei. Jede Szene wird dabei mit einem Punkt bewertet. Eine Variation einer bereits vorhandenen Szene wird mit einem halben Punkt bewertet.

Die Gesamtwertung pro Bild wird dann aus der Summe der Punkte der Diversitätskriterien abzüglich der Anzahl der Fehler in den Bildern gebildet. Für die Modellwertung werden dann die Einzelwertungen addiert und durch die Anzahl der Bilder geteilt. Das Modell mit der höchsten Gesamtwertung wird dann als das beste Modell ausgewählt.

In Tabelle 1 sind die Ergebnisse der Bewertung der Modelle dargestellt. Die untersuchten Bilder mit den markierten Fehlern und Instrumenten sind im Projektordner `eval_files_marked` zu finden. Eine Auswahl von Bildern ist in der Abbildung 1 zu sehen.

Anhand der Ergebnisse des Vergleichs der Modelle in Tabelle 1 sieht man, dass die Bilder des Modells Stable Diffusion XL (SDXL) mit 44 die meisten Fehler aufweisen. Das Modell HiDream-I1-Full hat mit 34 Fehlern die zweitmeisten Fehler. Mit einem Abstand von 12 Fehlern folgt das Modell Stable Diffusion 3.5 Large (SD35) mit 22 Fehlern. Das Modell FLUX.1-dev hat, mit einem Abstand von 9, die wenigsten Fehler mit insgesamt nur 13 Fehlern. Bei der Diversität der Instrumente fällt besonders auf, dass das Modell SDXL nur einen einzigen Punkt für die Instrumente erhält. Dies liegt an der Tatsache, dass es nur ein einziges erkennbares Instrument in allen Bildern erstellt hat. An vorletzter Stelle befindet sich das HiDream Modell mit einer Punktzahl von 13,5. Knapp davor mit 14 Punkten liegt das SD35 Modell. Mit einem Abstand von 1,5 Punkten bietet das FLUX.1 Modell mit einer Punktzahl von 15,5 die größte Diversität an Instrumenten. In der letzten Bewertungskategorie, der Diversität der Szenen, haben fast alle Modelle die volle Punktzahl von erhalten, da sie in jedem Bild eine andere Szene erstellt haben. Nur das HiDream Modell erhält mit 4 Punkten eine niedrigere Wertung, da es viermal nur eine Variation einer bereits erstellten Szene enthielt.

Basierend auf den Ergebnissen aus Tabelle 1 ergeben sich die folgenden Gesamtwertungen für die Modelle:

Dateiname	Anzahl Fehler	Diversität Instrumente	Diversität Szene	Wertung
HiDream-I1-Full_prompt-0_guidance-3.0_steps-50.png	7	3,5	1	-2,5
HiDream-I1-Full_prompt-0_guidance-5.0_steps-50.png	9	1,5	0,5	-7
HiDream-I1-Full_prompt-0_guidance-7.0_steps-50.png	8	4	0,5	-3,5
HiDream-I1-Full_prompt-1_guidance-3.0_steps-50.png	5	2	1	-2
HiDream-I1-Full_prompt-1_guidance-5.0_steps-50.png	5	1	0,5	-3,5
HiDream-I1-Full_prompt-1_guidance-7.0_steps-50.png	3	2,5	0,5	0
FLUX-1-dev_prompt-0_guidance-1.5_steps-50.png	2	2,5	1	1,5
FLUX-1-dev_prompt-0_guidance-3.5_steps-50.png	4	3	1	0
FLUX-1-dev_prompt-0_guidance-5.5_steps-50.png	3	4	1	2
FLUX-1-dev_prompt-1_guidance-1.5_steps-50.png	2	1	1	0
FLUX-1-dev_prompt-1_guidance-3.5_steps-50.png	1	3,5	1	3,5
FLUX-1-dev_prompt-1_guidance-5.5_steps-50.png	1	3,5	1	3,5
sd35_prompt-0_guidance-1.5_steps-28.png	2	1	1	0
sd35_prompt-0_guidance-3.5_steps-28.png	2	2,5	1	1,5
sd35_prompt-0_guidance-5.5_steps-28.png	4	3	1	0
sd35_prompt-1_guidance-1.5_steps-28.png	5	3	1	-1
sd35_prompt-1_guidance-3.5_steps-28.png	4	3,5	1	0,5
sd35_prompt-1_guidance-5.5_steps-28.png	5	2,5	1	-1,5
sdxlrefiner_prompt-0_guidance-5.5_steps-40.png	7	0	1	-6
sdxlrefiner_prompt-0_guidance-7.5_steps-40.png	7	0	1	-6
sdxlrefiner_prompt-0_guidance-9.5_steps-40.png	8	0	1	-7
sdxlrefiner_prompt-1_guidance-5.5_steps-40.png	7	0	1	-6
sdxlrefiner_prompt-1_guidance-7.5_steps-40.png	8	0	1	-7
sdxlrefiner_prompt-1_guidance-9.5_steps-40.png	7	1	1	-5

Tabelle 1: Bewertung der Modelle

- HiDream-I1-Full: -3
- FLUX-1-dev: 1,75
- sd35: 0,08
- sdxlrefiner: -6,1

Das Modell FLUX.1 dev hat die höchste Gesamtwertung und wird daher für die weitere Arbeit verwendet. Das Modell Stable Diffusion 3.5 Large (sd35) hat zwar auch eine positive Gesamtwertung, jedoch ist die Diversität der Instrumente und Szenen geringer als bei dem FLUX-Modell. Die Modelle HiDream-I1-Full und Stable Diffusion XL Refiner(sdxlrefiner) schneiden jeweils mit einer negativen Gesamtwertung ab. Daher werden diese drei Modelle nicht weiter betrachtet.

## **2.5. Erstellung des Datensatzes**

Nachdem nun ein Modell für die Generierung der Bilder ausgewählt wurde, kann mit der Erstellung des Datensatzes begonnen werden. Der Datensatz wird aus zwei Teilen bestehen. Der erste Teil wird aus echten Bildern bestehen, die aus dem Internet gesammelt werden. Der zweite Teil wird aus synthetischen Bildern bestehen, die mit dem ausgewählten Modell generiert werden. Die gesammelten und generierten Bilder werden dann mit einem Programm für die Annotation annotiert.

### **2.5.1. Sammeln von echten Bildern**

Die echten Bilder sollten aus dem Internet gesammelt werden. Bei der Auswahl der Bilder wurde darauf geachtet, dass sie unter einer freien Lizenz zur Verfügung standen, um das Urheberrecht nicht zu verletzen. Um dies zu gewährleisten, wurden die Bilder von der Webseite <https://pixabay.com/> verwendet. Diese Webseite bietet eine große Auswahl an Bildern, die unter der Pixabay Lizenz stehen. Diese Lizenz erlaubt es, die Bilder kostenlos zu verwenden, auch für kommerzielle Zwecke, ohne dass eine Namensnennung des Urhebers erforderlich ist. Die Bilder können auch bearbeitet werden. Bei der Auswahl der Bilder wurde auf ein möglichst natürliches Szenario geachtet. Um später aussagekräftige Ergebnisse bei der Objekterkennung zu erhalten, wurden Bilder mit ähnlichen und unterschiedlichen Instrumenten ausgewählt. So gibt es in den Bildern ähnliche Instrumente wie Gitarren, Banjos, Geigen und Cellos, aber Instrumente wie Saxophone, Pianos und Schlagzeuge. Zudem wurde auf verschiedene Betrachtungswinkel der verschiedenen Instrumente geachtet, sodass diese nicht immer vollständig oder von vorne zu sehen sind, sondern teilweise von der Seite oder von hinten oder oder durch verdeckt sind. Dies soll später für eine robustere Erkennung sorgen. Die Bilder wurden dabei in einer



Abbildung 2: Eine Auswahl aus den gesammelten Bildern.

Auflösung von annähernd 1024x1024 Pixeln ausgewählt, da diese Auflösung auch später für die Generierung der synthetischen Bilder verwendet wird. Insgesamt wurden 50 Bilder gesammelt die in dem Projektverzeichnis `code/dataset/real` zu finden sind.

### 2.5.2. Generierung von synthetischen Bildern

Auch die Generierung der synthetischen Bilder erfolgt in einem Jupyter Notebook. Welches unter dem Namen `image_generation.ipynb` zu finden ist. In diesem Notebook wird wieder die Pipeline der Hugging Face diffusers-Bibliothek verwendet. Dabei wird das zuvor ausgewählte FLUX.1-dev Modell verwendet. Für die Generierung werden Bilder in der Größe 1024x1024 Pixel generiert und unterschiedliche Parameter für die `guidance_scale` verwendet. Wie auch schon bei der Evaluation wird hier der empfohlene Wert von 3.5 verwendet, sowie einen Punkt darunter und darüber. Auch die Anzahl der Schritte wird auf den empfohlenen Wert von 28 gesetzt [11]. Um zwei gleich große Datensätze zu erhalten wird auch bei der Generierung ein Datensatz von 50 Bildern angestrebt. Dafür werden 10 verschiedene Texteingaben erstellt, die jeweils 5 Bilder generieren. Die Texteingaben sind dabei ähnlich zu den Eingaben der Evaluation, jedoch etwas allgemeiner gehalten, um eine größere Diversität an Instrumenten zu erhalten. Die Texteingaben sind:

- raw photo looking down on a simple table with guitars, electric keyboards, violins and banjos and other common items laying on the table
- raw photo of a simple living room with a piano, guitars, trumpets, cellos, potted plants and other items
- raw photo of a band of three people passionately playing in a park. There is a drummer, a guitar player, a singer, and a bass player
- raw photo of a wide busy shopping street with lots of people walking around and a violin player playing in front of a shop. A second violin is laying in a case next to the person



Abbildung 3: Eine Auswahl aus den generierten Bildern.

- raw photo of a stage with a rock band playing a concert. On the stage is a drummer, two guitar players, a singer and a guy playing a stage piano
- raw photo a woman sitting in a cozy room in front of a music stand with note sheets practicing the guitar, far shot
- raw photo of a guitar, a banjo, a violin and a cello next to each other
- raw photo of a music studio with big windows and a lot of music equipment and instruments
- raw photo of a band practicing in a rehearsal room with various instruments and equipment
- raw photo of musicians posing with their instruments including guitars, trumpets, saxophones, violins and electric guitars

Da vorraussichtlich nicht alle Bilder eine für den Datensatz ausreichende Qualität haben werden, werden pro Texteingabe und Parameterkombination 15 Bilder generiert. Dadurch wird sichergestellt, dass am Ende mindestens 50 Bilder generiert werden, die für den Datensatz verwendet werden können. Die generierten Bilder werden in dem Projektverzeichnis `code/dataset/synth` mit dem Namensschema

`synth_prompt-count_guidance-scale_image-count.png` gespeichert. Damit lässt sich später nachvollziehen, welche Bilder mit welcher Texteingabe und welchen Parametern generiert wurden. Die Auswahl der Bilder für den Datensatz aus den generierten Bilder erfolgte in mehreren Durchläufen durch den Autor, basierend auf den bereits für die Evaluierung verwendeten Kriterien. Dafür wurden im ersten Durchlauf die Bilder mit groben Fehlern aussortiert. Anschließend wurden die Bilder immer weiter reduziert bis die angestrebte Anzahl von 50 Bildern erreicht wurde. Bei der Auswahl wurde auch die Diversität der Instrumente und Szenen berücksichtigt.

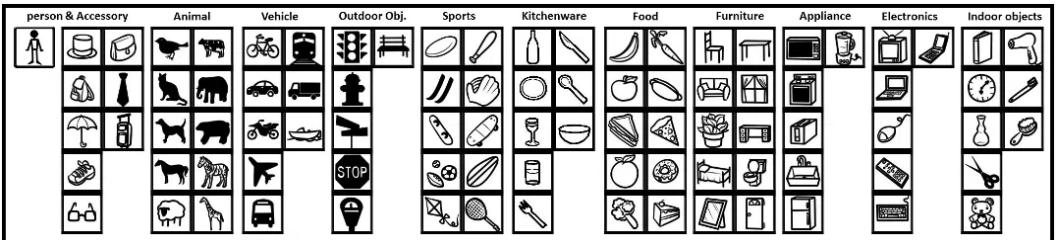


Abbildung 4: Eine Abbildung der Kategorien des COCO Datensatzes gruppiert nach Oberkategorien. Quelle: [13, S. 13]

### 2.5.3. Annotation der Bilder

Um möglichst einfach vergleichbare Ergebnisse beim Vergleich der Objekterkennung zu erhalten, sollen die Bilder im Format des COCO Datensatzes annotiert werden. Dadurch können sie einfach mit Modellen verwendet werden, die bereits auf dem COCO Datensatz trainiert wurden. Er bietet außerdem eine einheitliche Struktur für Metriken um die Leistung von Modellen zu vergleichen. Zu den Metriken gehören unter anderem die *average precision* (AP) und der *average recall* (AR) [12]. Der COCO Datensatz ist ein weit verbreitet Datensatz der für verschiedene computer vision Aufgaben benutzt werden kann. Dazu gehören verschiedene Aufgaben der Objekterkennung, wie zum Beispiel die Objektdetektion mittels *bounding boxes*, die Segmentierung von Objekten oder die Klassifikation von Objekten und die Erkennung von *keypoints*. Außerdem gibt es Label für Bildbeschreibung. Der Datensatz enthält 330.00 Bilder von denen über 200.000 Bilder annotiert sind. Der Datensatz enthält 91 Objektkategorien, die in 11 Gruppen unterteilt sind [1]. Allerdings enthalten die Bilder nur Annotationen für 80 Kategorien. Alle Kategorien sind in der Abbildung 4 dargestellt. Die Annotationen des COCO Datensatzes sind im JSON Format gespeichert. Dabei werden die Bilder, Kategorien und Annotationsdaten in separaten Listen gespeichert. Die Annotationsdaten enthalten dabei die Koordinaten der *bounding boxes* der Objekte, die Kategorie des Objekts und eine ID für die Annotation. Die Kategorien sind dabei in einer separaten Liste gespeichert, die eine ID und den Namen der Kategorie enthält. Die Bilder sind ebenfalls in einer separaten Liste gespeichert, die eine ID, den Dateinamen und die Größe des Bildes enthält.

Für die Annotation gibt es verschiedene Programme, die das Annotieren von Bildern erleichtern, indem sie eine grafische Oberfläche bieten, inder die Bilder angezeigt werden und die Annotationen bequem mit der Maus gesetzt werden können. Viele dieser Programme bieten zusätzliche Funktionen, für die Annotation von Datensätzen, wie zum Beispiel die Möglichkeit mehrere Datensätze zu verwalten, eine Benutzerverwaltung um mehrere Benutzer gleichzeitig an einem Datensatz arbeiten zu lassen, oder die Möglichkeit die Annotationen in verschiedenen Formaten zu exportieren. Einige Programme bieten auch

die Möglichkeit bestehende KI-Modelle zu verwenden, um die Annotationen automatisch zu erstellen. Die Programme unterscheiden sich dabei in der Anzahl der unterstützten Formate, der Benutzerfreundlichkeit und den Funktionen. Einige Programme sind Open-Source und können kostenlos verwendet werden, andere sind kostenpflichtig und bieten zusätzliche Funktionen.

Im Rahmen der vorbereitenden Recherche für dieses Projekt wurden bereits die Programme Label Studio, CVAT und COCO Annotator entdeckt. Diese drei Programme sind für die nicht-kommerzielle Nutzung kostenlos und bieten eine Vielzahl von Funktionen. Für die Annotation der Bilder wurde schlussendlich der COCO Annotator gewählt, da dieser die einfachste Einrichtung bietet. Für die Installation bietet der COCO Annotator im GitHub Repository eine Docker-Compose Datei an, die die Installation und Konfiguration des Programms vereinfacht. Dadurch kann das Programm schnell und einfach auf einem Computer mit Docker installiert werden.

Nach der Installation des Programms konnten die Bilder in das Programm importiert werden. Dies war geschah durch die Ablegen der Bilder in einem Ordner, der an die Docker-Container gebunden ist. In diesem Ordner wurde ein weiterer Ordner erstellt, dessen Name als Name des Datensatzes in der Anwendung dient. Die darin enthaltenen Bilder wurden dann von der Anwendung automatisch erkannt. Als nächsten wurden die bestehenden Kategorien des COCO Datensatzes in das Programm importiert. Anschließend wurde anhand einer ersten Sichtung der Bilder weitere Kategorien den bestehenden hinzugefügt. Insgesamt wurden sieben neue Kategorien unter der Oberkategorie *instrument* hinzugefügt. Diese sind:

- Guitar (Gitarre), sowohl akustische als auch elektrische Gitarren
- Benjo (Banjo)
- Violin (Geige)
- Cello
- Piano (Klavier), sowohl Flügel, Keyboards als auch Klaviere
- Saxophone (Saxophon)
- Drums (Schlagzeug), Schlagzeug, Trommeln und Percussion Instrumente

Nachdem die Vorbereitungen abgeschlossen waren, konnte mit der Annotation der Bilder begonnen werden. Aufgrund der verfügbaren Zeit wurden für die Bilder nur *bounding box* Annotationen erstellt. Diese erfordern keine aufwendige Erstellung von Masken für die Objekte, sondern nur die Angabe einer rechteckigen Box um das Objekt. Diese können einfach mit der Maus direkt auf dem Bild gesetzt werden. Bei der Annotation wurden nicht nur die Objekte der neuen Kategorien annotiert, sondern auch die Objekte der bestehenden Kategorien des COCO Datensatzes. Die Annotation nahm insgesamt

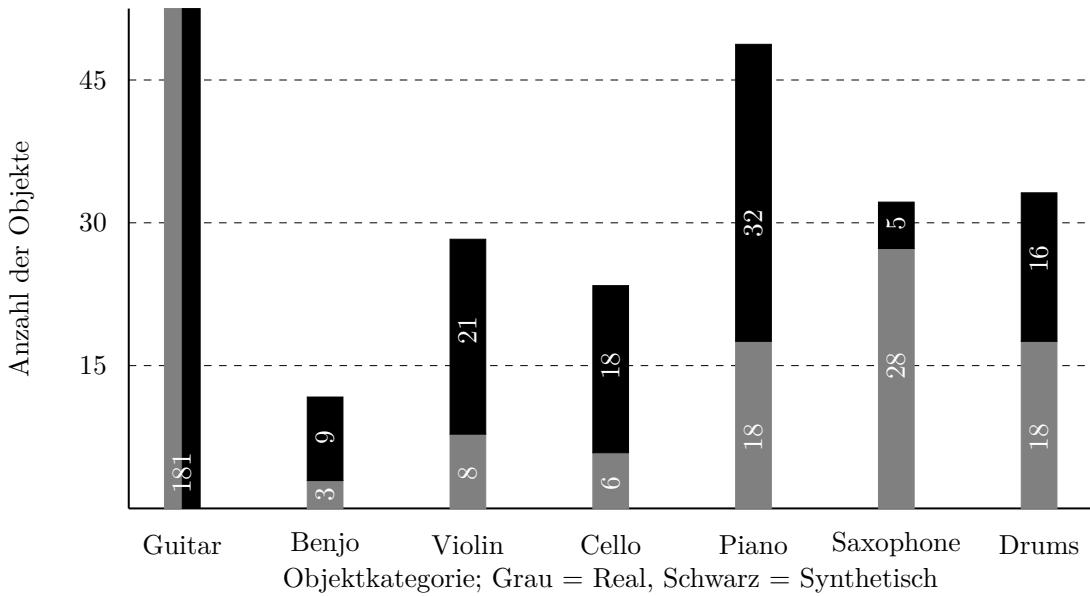


Abbildung 5: Säulendiagramm zur Auswertung der Anzahl der annotierten Objekte pro Kategorie.

ca. 10 Stunden in Anspruch. Nachdem alle Bilder annotiert wurden, konnte den die Annotationen mittels eines Buttons in der Oberfläche als JSON-Datei exportiert und direkt im Browser heruntergeladen werden.

Abbildung 5 zeigt ein Säulendiagramm der Anzahl der annotierten Objekte pro Kategorie. Dabei ist zu erkennen, dass mit Abstand die meisten Objekte in der Kategorie *Gitarre* (Gitarre) annotiert wurden. Da diese weit über den Bereich Wertebereich des Diagramms hinausgehen. Hier enthalten die echten Bilder 81 Instanzen und die generierten Bilder 100 Instanzen. Dies ist sowohl bei den realen als auch bei den synthetischen Bildern mehr als doppelt so viel wie die nächst kleinere Kategorie *Piano* (Klavier). Grundsätzlich ist zu erkennen, dass die Anzahl der Annotationen in den synthetischen Bildern höher ist als in den realen Bildern. Dies kann beim Vergleich der Ergebnisse mit diesen Trainingsdaten zu Verzerrungen führen und sollte bei der weiteren Verwendung beachtet werden. Dies kann zum Beispiel durch das Sammeln von weiteren realen Bildern ausgeglichen werden. Aufgrund der verfügbaren Zeit des Projektes ist dies jedoch in diesem Projekt nicht mehr möglich.

Die JSON-Datei `real-synth_image_annotations.json` im Verzeichnis `code/dataset` enthält aktuell alle Annotationen für die 100 Bilder. Für ein getrenntes Training der realen und generierten Bilder wurden die Annotationen in zwei Dateien aufgeteilt. Die Annotationen der realen Bilder sind in der Datei `real_image_annotations.json` und die Annotationen der synthetischen Bilder in der Datei `synth_image_annotations.json`

im jeweiligen Unterverzeichnis der Bilder zu finden. Dies ermöglicht eine getrennte Verwendung der beiden Teile des Datensatzes, aber auch eine gemeinsame Verwendung, da es innerhalb der JSON-Dateien keine sich überschneidenden IDs gibt.

Um die Verwendbarkeit des Datensatzes zu testen, wurde ein weiteres Jupyter Notebook erstellt, in dem der erstellte Datensatz geladen und für das Training eines Modells verwendet wird. Um den Datensatz zu laden wird das Modul `COCODetection` aus dem `torchvision` Paket verwendet. Dieses Modul ist speziell für das Laden von COCO Datensätzen entwickelt worden und bietet eine einfache Möglichkeit, die Bilder und Annotationen zu laden und zu verarbeiten. Um ein `Dataset` Objet zu erstellen, genügt es der Klasse `CocoDetection` den Pfad zu dem Verzeichnis mit den Bildern und den Pfad zur JSON-Datei mit den Annotationen zu übergeben. Zusätzlich wird noch ein Transformationsobjekt übergeben, welches die Bilder in das richtige Format für das Training bringt. In diesem Fall wird das Transformationsobjekt `ToTensor` verwendet, welches die Bilder in Tensoren umwandelt. Für das Training wird das *Faster R-CNN* Modell aus dem `torchvision` Paket verwendet. Um das Dataset mit diesem Modell nutzen zu können, muss der Datensatz in das richtige Format gebracht werden. Dafür wurde die Funktion `convert_target` erstellt. Diese wandelt eine COCO-Annotation in das vom Modell erwartete Format um. Dieses besteht aus einem *Dictionary*, mit den Schlüsseln `boxes`, `labels`. Dabei ist `boxes` eine Liste von *bounding boxes* in der Form `[x1, y1, x2, y2]`. Die `labels` sind die IDs der Kategorien, die den Objekten in den Bildern zugeordnet sind. Diese IDs entsprechen den IDs in der JSON-Datei mit den Annotationen. Die Funktion `collate_fn` wird verwendet um die Bilder und Annotationen richtig zusammenzuführen. Anschließend wird das Modell mit der `torchvision` Funktion `train_fasterrcnn_resnet50_fpn` geladen, in den Trainingsmodus versetzt und, falls möglich, auf eine GPU verschoben. Das Modell wird dann in 10 Epochen und einem *SGD-Optimizer* trainiert. Als Ergebnis wird in der ersten Epoche ein Loss von 25.9 erreicht und in der zehnten Epoche ein Loss von 4.6.

### 3. Projektabschluss

Mit dem Erfolg des Tests des Datensatzes wurde das Projekt erfolgreich und innerhalb der Zeitvorgaben abgeschlossen. Zusammenfassend wurden im Rahmen dieses Projekts sowohl echte als auch synthetische Bilder gesammelt, sorgfältig annotiert und in einem einheitlichen Format bereitgestellt. Durch die Evaluation verschiedener Bildgenerierungsmodelle konnte ein geeignetes Modell für die Erstellung synthetischer Daten ausgewählt werden. Die Annotation im COCO-Format ermöglicht eine direkte Weiterverwendung mit etablierten Frameworks und Modellen im Bereich der Objekterkennung. Die im Projekt gewonnenen Erkenntnisse und der erstellte Datensatz bilden eine solide Grundlage

für weiterführende Arbeiten, insbesondere für die geplante Bachelorarbeit, in der die Leistungsfähigkeit von Objekterkennungsmodellen auf Basis dieses Datensatzes untersucht werden soll. Darüber hinaus kann der Workflow zur Datensatzgenerierung und -annotation als Vorlage für ähnliche Projekte dienen. Die Ergebnisse des Projektes werden in einer noch folgenden abschließenden Präsentation vorgestellt.

Die vollständigen Dateien des Projekts, einschließlich der generierten Bilder, Annotationen und Jupyter Notebooks, sind im GitHub Repository <https://github.com/fhswf-projekte/real-synth-object-detection-dataset-titom001> verfügbar.

## Literatur

- [1] C. Consortium. „Common Objects in Context.“ (2015), Adresse: <http://cocodataset.org> (besucht am 09.05.2025).
- [2] I. Hugging Face. „HuggingFace - The AI community building the future.“ (2025), Adresse: <https://huggingface.co/> (besucht am 09.05.2025).
- [3] C. Corporation. „Computer Vision Annotation Tool.“ (2025), Adresse: <https://www.cvcat.ai/> (besucht am 09.05.2025).
- [4] I. HumanSignal. „Label Studio.“ (2025), Adresse: <https://labelstud.io/> (besucht am 09.05.2025).
- [5] Q. Cai, J. Chen, Y. Chen u. a., *HiDream-I1: A High-Efficient Image Generative Foundation Model with Sparse Diffusion Transformer*, 2025. arXiv: 2505.22705 [cs.CV]. Adresse: <https://arxiv.org/abs/2505.22705>.
- [6] B. F. Labs, *Announcing Black Forest Labs*, 2024. Adresse: <https://bfl.ai/announcements/24-08-01-bfl> (besucht am 15.05.2025).
- [7] P. Esser, S. Kulal, A. Blattmann u. a., *Scaling Rectified Flow Transformers for High-Resolution Image Synthesis*, 2024. arXiv: 2403.03206 [cs.CV]. Adresse: <https://arxiv.org/abs/2403.03206>.
- [8] D. Podell, Z. English, K. Lacey u. a., *SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis*, 2023. arXiv: 2307.01952 [cs.CV]. Adresse: <https://arxiv.org/abs/2307.01952>.
- [9] P. von Platen, S. Patil, A. Lozhkov u. a., *Diffusers: State-of-the-art diffusion models*, <https://github.com/huggingface/diffusers>, 2022.
- [10] I. Hugging Face, *Memory Optimization in Diffusers*, 2025. Adresse: <https://huggingface.co/docs/diffusers/optimization/memory> (besucht am 15.05.2025).
- [11] B. F. Labs, *Hugging Face - FLUX.1-dev*, 2025. Adresse: <https://huggingface.co/black-forest-labs/FLUX.1-dev> (besucht am 15.05.2025).

- [12] C. Consortium. „Common Objects in Context - Metrics.“ (2015), Adresse: <http://cocodataset.org/#detection-eval> (besucht am 15.05.2025).
- [13] T.-Y. Lin, M. Maire, S. Belongie u. a., *Microsoft COCO: Common Objects in Context*, 2015. arXiv: 1405.0312 [cs.CV]. Adresse: <https://arxiv.org/abs/1405.0312>.

## **Erklärung**

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und dabei keine anderen als die angegebenen Hilfsmittel benutzt habe. Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach Werken anderer Autoren entnommen sind, habe ich als solche kenntlich gemacht. Die Arbeit wurde bisher weder gesamt noch in Teilen einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

5. Juli 2025

Tim Tomczek