Praktikumsbericht KI und Sicherheit in der Praxis: Age-Estimation mit YOLO

Autor: Santiago del Rio

Matrikelnummer: 2577256

Praktikum: 20-00-1221 Praktikum KI und Sicherheit in der Praxis

Dozenten: Dr.-Ing. Oren Halvani, Eneldo Loza Mencia

Datum: 28. September 2025

Einleitung

Die automatische Altersklassifikation in Bildern ist seit Jahren Gegenstand intensiver Forschung, bleibt aber bis heute eine offene Herausforderung. Bestehende Modelle zeigen häufig gute Ergebnisse auf kuratierten Benchmark-Datensätzen, scheitern jedoch an realistischen „in the wild“-Bildern. Genau diese realweltliche Robustheit wäre jedoch entscheidend für praktische Anwendungen – sei es in der Inhaltsmoderation, bei Zugangskontrollen oder bei sicherheitsrelevanten Szenarien. Ein zentrales Problem ist die Datenlage: Viele veröffentlichte Datensätze sind entweder urheberrechtlich eingeschränkt, ethisch sensibel oder nur in stark kontrollierter Form nutzbar. Gerade für Minderjährige ist die Verfügbarkeit von Bildmaterial mit entsprechenden Lizenzrechten aus naheliegenden Gründen stark begrenzt, was die Entwicklung fairer und verlässlicher Modelle massiv erschwert.

Die aktuelle Forschung verdeutlicht diese Schwierigkeiten. So weist das DeepUAge Paper von Anda et al. aus dem Jahre 2020 (Anda u. a. 2020) darauf hin, dass das Arbeiten mit starren Altersgruppen problematisch ist, da sie wenig dynamisch und schlecht an die natürliche Alterskontinuität angepasst sind. Zudem fehlen in fast allen Datensätzen genügend Bilder von Minderjährigen, und Alterslabels basieren oft auf Schätzungen oder Annahmen statt auf gesicherten Ground Truths. Der IMDB Datensatz ist ein Beispiel, jedoch fehlen auch hier Bilder von Jugendlichen und Bilder in natürlicheren Situationen. Auch moderne Ansätze wie MiVOLO(Kuprashevich und Tolstykh 2023) diskutieren die Abwägung zwischen Klassifikation und Regression: Klassifikationsmodelle sind stabiler zu trainieren, können aber nicht zwischen kleinen Fehlern (z. B. 16 statt 17 Jahre) und großen Abweichungen (z. B. 17 statt 40 Jahre) unterscheiden. Regression erlaubt dagegen eine genauere Abbildung von Altersspannen, ist aber aufgrund der schwierigen Datensituation deutlich schwerer zu trainieren.

Vor diesem Hintergrund lag der Schwerpunkt meines Projekts auf der Erstellung eines eigenen Datensatzes und dem Training von YOLO-basierten Klassifikationsmodellen. Die Grundlage des Projekts bildete ein frei zugänglicher Bilddump von Unsplash, der 25.092 Fotos umfasste. Diese Bilder deckten ein breites Spektrum an Szenen ab. Viele Bilder enhielten keine Personen oder keine Gesichter, viele jedoch auch mehrere. Jedem Bild waren Schlagworte („Tags“) zugeordnet. In der Praxis erwiesen sich diese Tags jedoch als kaum nutzbar: Erwachsene sind nur selten explizit mit „adult“ markiert, während Bezeichnungen wie „girl“ oder „boy“ sowohl bei Kindern als auch bei Erwachsenen auftreten. Selbst der Tag „baby“ führte häufig zu Bildern, auf denen ausschließlich Erwachsene zu sehen waren. Für die besonders sensiblen Randfälle (z. B. 17 vs. 18 Jahre) waren die Tags völlig ungeeignet. Deshalb mussten eigene Methoden zur Annotation entwickelt werden, anstatt auf die Unsplash-Tags zurückzugreifen.

Um aus diesem unsauberen Ausgangsmaterial einen trainierbaren Datensatz zu erstellen, habe ich eine mehrstufige Pipeline implementiert. Zunächst wurden Personen und Gesichter automatisch detektiert, Crops erstellt und anschließend mit Altersvorhersagen (DeepFace, MiVOLOv2) annotiert. Zusätzlich wurden Qualitätsmetriken wie Schärfe, Helligkeit und Kontrast berechnet, um unbrauchbare Bilder zu filtern.

Pipeline zur Erstellung des Datensatzes

Für das Training und die Datenaufbereitung habe ich den Lichtenberg-Cluster der TU Darmstadt genutzt. Über SLURM-Jobs konnte ich GPU-Knoten für das Training und die Vorverarbeitung ansteuern. Der Quellcode lag im Home-Verzeichnis, während die großen Bilddaten im Scratch-Speicher verarbeitet wurden. Die gesamte Pipeline war in einzelne Python-Skripte aufgeteilt, die jeweils einen Verarbeitungsschritt abdecken. Die Versionsverwaltung lief über GIT. Um die Abläufe reproduzierbar und flexibel zu halten, wurden alle Pfade, Schwellenwerte und Parameter zentral in einer config.yaml abgelegt. So konnte ich Änderungen an den Einstellungen vornehmen, ohne den Code in jedem Skript

anpassen zu müssen. Hier nun eine kurze Erklärung jedes einzelnen Schrittes in der Pipeline:

01_detect_bodies.py:

Im ersten Schritt werden mit einem YOLOv8-Modell die Körper von Personen in den ursprünglichen Bildern erkannt. Das Skript lädt die Konfiguration aus der config.yaml sucht alle Bilder im Eingabeverzeichnis und führt eine Personendetektion durch. Für jedes erkannte Objekt wird ein Bounding Box berechnet, per Non-Maximum-Suppression auf Überschneidungen geprüft und anschließend als Bildausschnitt gespeichert. Neben den Crops selbst werden alle relevanten Informationen (Bild-ID, Confidence-Wert, Modellname, Dateiname) in einer CSV-Datei festgehalten. Dieser Schritt reduziert die Datenmenge und stellt sicher, dass nachfolgende Analysen nur noch mit den relevanten Bildausschnitten arbeiten. Sie dienen auch als Input für MiVOLOv2 um das Alter zu schätzen.

02_detect_faces.py:

Anschließend werden in den Körperausschnitten Gesichter lokalisiert. Dafür wird die DeepFace-Bibliothek, die auf verschiedene Backend-Modelle zur Gesichtserkennung zurückgreifen kann. In diesem Fall wurde yolov8 verwendet. Jedes gefundene Gesicht wird abgespeichert und mit den zugehörigen Metadaten (u. a. Confidence, IDs für Bild/Körper/Gesicht) in eine eigene CSV geschrieben. Ich habe mich dafür entschieden das in diese zwei Schritte aufzuteilen, da es Modelle wie MiVOLOv2 gibt, welche zur Alterserkennung den Körper mit hinzunehmen können oder müssen. Der Datensatz soll am Ende jedoch nur Gesichter enthalten. Hier dient dieser zweite Schritt dazu, dass Körper und Gesichter einander zugeordnet werden können.

03_delete_duplicates.py:

Bei einem großen, automatisch erzeugten Datensatz treten häufig redundante Bilder auf. Das kann z. B. auftreten, wenn beim Exportieren von Unsplash Bilder mehrfach in über verschiedene Tags geladen werden oder wenn bei der Detection trotz IOU-Test mehrere

Crops derselben Person aus demselben Bild gespeichert werden. Dieses Skript berechnet deshalb für jedes Gesicht einen perceptual hash (pHash) und entfernt Duplikate. Die dazugehörige faces.csv wird automatisch angepasst, indem alle gelöschten Einträge entfernt werden. Am Ende bleibt ein bereinigter Datensatz übrig, der Redundanzen vermeidet und so die Qualität der Trainingsdaten erhöht.

04_quality_faces.py:

In diesem Schritt wird die Qualität der Gesichtsausschnitte bewertet. Das Skript berechnet drei Metriken: Schärfe, Helligkeit (Durchschnitt der Grauwerte) und Kontrast (Standardabweichung der Grauwerte). Zusätzlich wird überprüft, ob die Gesichtsausschnitte eine Mindestgröße erreichen. Nur wenn alle Bedingungen erfüllt sind, wird ein Gesicht als „usable“ markiert. Diese Information wird in die CSV geschrieben und dient später dazu, ungeeignete Bilder auszuschließen. Damit wird verhindert, dass unscharfe, überbelichtete oder extrem kleine Gesichter die Trainingsqualität verschlechtern.

05_annotate_faces_Deepface.py:

Nachdem die Datenqualität sichergestellt war, wurden die nutzbaren Gesichter mit DeepFace annotiert. Das Skript lädt jedes Bild und führt eine Analyse der Attribute Alter, Geschlecht und Ethnizität durch. Die Ergebnisse werden in die bestehende faces.csv geschrieben, sodass für jedes Gesicht neben technischen Metriken auch semantische Informationen vorliegen. Falls ein Bild als unbrauchbar markiert wurde oder die Analyse fehlschlug, wurde es übersprungen.

Bei der Altersannotation zeigte sich allerdings, dass DeepFace für diesen Zweck nur eingeschränkt geeignet ist: In einem Test mit 107 Bildern lag das geschätzte Alter immer zwischen 20 und 41 Jahren – selbst bei Babys, Kindern und älteren Erwachsenen. Besonders auffällig war, dass drei der fünf Personen mit dem höchsten geschätzten Alter tatsächlich Kleinkinder waren. Zusätzlich wurden weibliche Gesichter deutlich häufiger fälschlich als männlich erkannt, als umgekehrt. Nach dieser Auswertung und einer Analyse der Confidence-Werte habe ich den Grenzwert definiert: Werte unter 50 werden verworfen.

DeepFace diente daher in der weiteren Pipeline nicht als primäre Altersquelle, sondern hauptsächlich für Gender- und Race-Annotationen sowie als Fallback für das Alter.

06_annotate_faces_mivolo2.py:
Zur primären Altersvorhersagen wird hier das Modell MiVOLOv2 eingesetzt. Es kombiniert Gesichts- und Körperinformationen und ist damit robuster gegenüber schwierigen Bedingungen, etwa wenn das Gesicht teilweise verdeckt ist. Das Skript verarbeitet die Daten in Batches und nutzt GPU-Beschleunigung, um effizient Alterswerte vorherzusagen. Die Ergebnisse werden als numerische Altersschätzung mit Dezimalwert in die faces.csv eingetragen.

07_export_and_build_yolo_dataset.py:
Im letzten Schritt wird aus den annotierten Gesichtern ein YOLO-kompatibler Datensatz erzeugt. Zunächst werden nur die als „usable“ markierten Crops exportiert. Anschließend werden Alterslabels anhand definierter Prioritäten (MiVOLOv2 vor DeepFace) ausgewählt und in die vordefinierten Altersklassen Säuglinge (0 – 1 Jahre), Kleinkinder (2 – 3 Jahre), Kinder (4 – 12 Jahre), Jugendliche (13 – 17 Jahre), Erwachsene (18 – 64 Jahre) sowie Senioren (65+ Jahre) überführt. Das Skript balanciert die Daten, indem es pro Klasse eine feste Zielanzahl von Bildern auswählt oder oversampled, und teilt den Datensatz automatisch in Training, Validierung und Test auf. Das Ergebnis ist ein vollständiger, sauber strukturierter Datensatz, der direkt zum Training von YOLO genutzt werden kann.

Datensatz

Aus den ursprünglichen 25.092 Fotos wurden insgesamt 17.987 Gesichter mit ausreichender Bildqualität extrahiert. Ein zentrales Problem stellte die unausgeglichene Klassenverteilung dar. Ohne Sampling war die Trainingsmenge extrem ungleichmäßig verteilt: Über 11.000 Bilder von Erwachsenen standen lediglich 199 Senior:innen, 320 Kleinkindern und 341 Jugendlichen gegenüber. Diese Schieflage hätte dazu geführt, dass

das Modell stark in Richtung der Mehrheitsklasse „Erwachsene“ verzerrt worden wäre. Um dieses Problem zu beheben, habe ich Data Sampling durchgeführt. Dabei wurde die Trainingsmenge künstlich ausgeglichen, indem für jede Altersklasse 1.200 Bilder ausgewählt oder durch Oversampling ergänzt wurden. Die Validierungs- und Testmengen habe ich unverändert gelassen, um die natürliche Verteilung beizubehalten und realistische Evaluationsergebnisse zu erhalten.

Altersklasse	Train (roh)	Train (gesampelt)	Val	Test
Erwachsene	11.213	1.200	1.401	1.403
Kinder	1.918	1.200	239	241
Jugendliche	341	1.200	42	41
Kleinkinder	320	1.200	40	41
Säuglinge	396	1.200	49	50
Senioren	199	1.200	24	26

So ist der Trainingsdatensatz deutlich balancierter, mit insgesamt 7.200 Bildern, ergänzt durch eine Validierungsmenge von 1.795 und eine Testmenge von 1.805 Bildern.

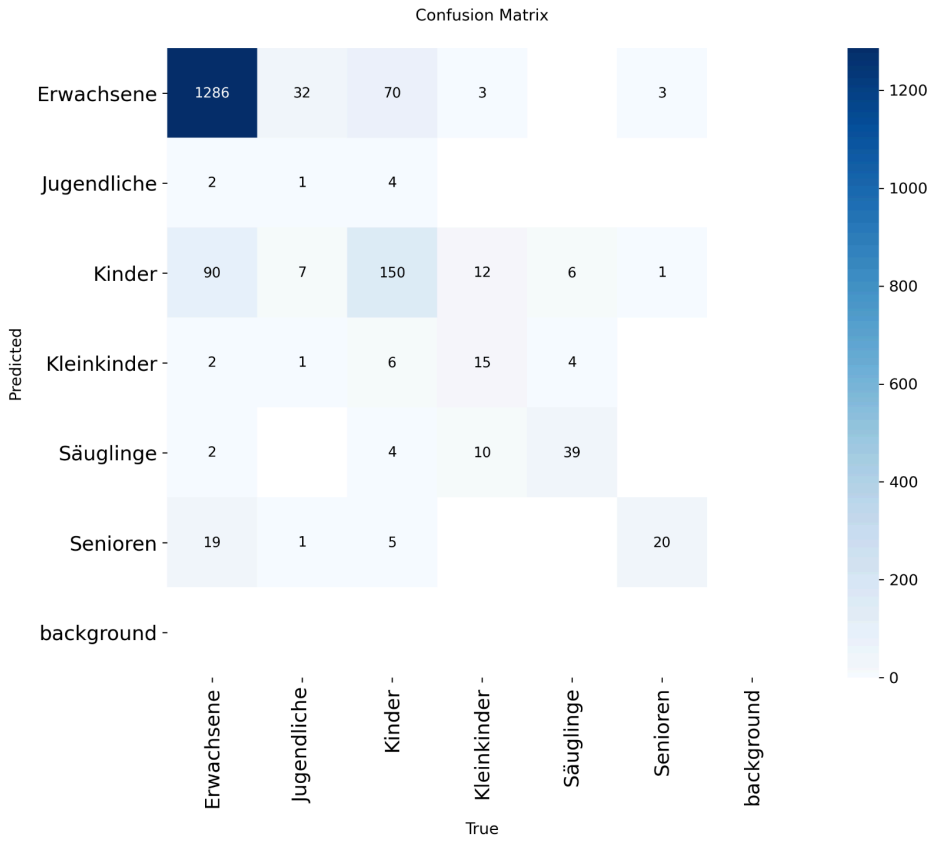
Neben den Altersannotationen habe ich die von DeepFace vorhergesagten Attribute Gender und Race ausgewertet. Die Verteilung zeigt ein deutliches Ungleichgewicht: Etwa 64,7 % der Gesichter wurden als „Man“ und 35,3 % als „Woman“ erkannt. Auch bei den Race-Kategorien ist die Verteilung ungleichmäßig. Der größte Anteil entfiel auf „white“ (42,6 %), gefolgt von „black“ (30,6 %) und „asian“ (15,5 %). Deutlich seltener traten die Kategorien „latino hispanic“ (5,9 %), „middle eastern“ (3,7 %) und „indian“ (1,8 %) auf. Die Auswertung macht deutlich, dass die Unsplash-Daten selbst eine starke Verzerrung aufweisen. Damit ist klar, dass der Datensatz nicht repräsentativ ist und bereits auf Ebene der Rohdaten einen Bias enthält, der sich später auch in die Modellleistung überträgt.

Trainiertes Modell

Für das Training des YOLO-Klassifikationsmodells habe ich einen ersten Lauf mit 10 Epochen, Batchgröße 32 und einer Bildgröße von 224 Pixeln durchgeführt. Als Optimizer wurde AdamW mit einer Lernrate von 0,001 und Momentum von 0,9 verwendet. Weitere Hyperparameter wurden in diesem Stadium nicht getestet, da der Schwerpunkt des Projekts auf der Erstellung und Aufbereitung des Datensatzes lag. Gegen Ende des Praktikums fehlte die Zeit, noch umfangreiche Experimente mit längeren Trainingsläufen oder systematischem Hyperparameter-Tuning durchzuführen. Das ist zwar bedauerlich, erlaubt aber trotzdem bereits eine erste Einschätzung der Modellleistung.

Das Modell erreichte eine Top-1-Accuracy von 80,6 %. Die Konfusionsmatrix zeigt, dass die Klasse Erwachsene mit sehr hoher Präzision und Recall erkannt wird, während kleinere Klassen wie Jugendliche oder Senioren deutlich schwächer abschneiden. Besonders Jugendliche werden oft mit Kindern oder Erwachsenen verwechselt. Insgesamt erreicht das Modell bei den großen Klassen solide Werte, weist jedoch – trotz des angewandten Samplings – nach wie vor Probleme mit den Randklassen auf.

Altersklasse	Precision	Recall	F1-Score
Erwachsene	0,923	0,918	0,92
Jugendliche	0,143	0,024	0,041
Kinder	0,564	0,628	0,594
Kleinkinder	0,536	0,375	0,441
Säuglinge	0,709	0,796	0,75
Senioren	0,444	0,833	0,58
Avg	0,553	0,596	0,554



Die Ergebnisse verdeutlichen zwei Punkte: Erstens funktioniert die Pipeline prinzipiell und ermöglicht es, ein YOLO-Modell auf Basis der automatisch annotierten Daten zu trainieren. Zweitens ist klar, dass für eine robuste und faire Altersklassifikation mehr Trainingszeit, systematisches Hyperparameter-Tuning und eine noch sorgfältigere Behandlung der kleineren Klassen nötig sind.

Besonders kritisch ist die sehr schwache Performance in der Klasse Jugendliche. Recall und F1-Score liegen dort nahe null, was bedeutet, dass das Modell Jugendliche fast nie korrekt erkennt und sie stattdessen meist als Kinder oder Erwachsene einordnet. Damit verfehlt es genau die für den Anwendungsfall zentrale Aufgabe: eine zuverlässige Trennung von Minderjährigen und Erwachsenen. Trotz akzeptabler Gesamt-Accuracy ist das Modell in seiner aktuellen Form für diesen Zweck praktisch unbrauchbar

Ausblick

Für die Weiterarbeit bieten sich mehrere Schritte an. Zunächst könnte das bestehende YOLO-Modell länger und mit systematischem Hyperparameter-Tuning auf dem erstellten Datensatz trainiert werden, um das volle Potenzial auszuschöpfen. Darüber hinaus ließen sich weitere Modelle wie Vision-Language-Modelle (z. B. CLIP, GEMMA), InsightFace oder DEX-Age in die Pipeline integrieren. Mit einem Ensemble dieser Verfahren könnten robustere Labels erzeugt werden, die durch eine manuelle Sichtung insbesondere bei Grenzfällen (z. B. 17 vs. 18 Jahre) zusätzlich abgesichert werden. Auch andere Bilddatensätze könnten mit derselben Pipeline aufbereitet und annotiert werden, um die Datenbasis zu erweitern. Anschließend wäre eine Evaluation der trainierten Modelle auf externen Benchmarks wichtig, um die Generalisierbarkeit über Unsplash hinaus zu prüfen.

Literaur

Anda, Felix, Nhien-An Le-Khac, und Mark Scanlon. 2020. „DeepUAge: Improving Underage Age Estimation Accuracy to Aid CSEM Investigation“. *Forensic Science International: Digital Investigation* 32 (April): 300921. https://doi.org/10.1016/j.fsidi.2020.300921.

Kuprashevich, Maksim, und Irina Tolstykh. 2023. „MiVOLO: Multi-input Transformer for Age and Gender Estimation“. arXiv:2307.04616. Preprint, arXiv, September 22. https://doi.org/10.48550/arXiv.2307.04616.