Digital Business University of Applied Sciences

Data Science und Business Analytics

WAI81 AI – Theory, Methods and Frameworks

Tobias Heuser

Emotionserkennung zur Bewertung von Kampagnenalternativen

Studienarbeit

Eingereicht von: Mareike Lass-Hennemann

Matrikelnummer: 190190

25.05.2025

Inhaltsverzeichnis

Abbildungsverzeichnis

Einführung

Um Online-Marketing-Kampagnen effektiver zu gestalten und besser auf die Bedürfnisse der Zielgruppen eingehen zu können, soll für ein Publishing-Unternehmen zukünftig in kleinen Testgruppen die emotionale Wirkung von Kampagnen-Varianten auf die Konsumenten getestet werden. Emotionale Wirkung ist im heutigen Marketing ein zentrales Mittel, um die Aufmerksamkeit und Erinnerung von Konsumenten zu steigern, Markenbindung und Differenzierung zu fördern und Kaufentscheidungen positiv zu beeinflussen. Konsumenten stehen täglich einer Vielzahl von Werbebotschaften gegenüber, wobei emotionale Botschaften dabei helfen können, sich von der Masse abzuheben und eine tiefere Verbindung zum Kunden aufzubauen. Eine Studie von Kantar Millward Brown zeigte 2017, dass der Absatz steigt, wenn der Fokus der Werbung auf emotionaler Relevanz liegt. Dennoch stützten sich zum damaligen Zeitpunkt nur wenige Werbeformate allein auf die emotionale Wirkung (Speck, A. 2017). Insbesondere im Online-Marketing stehen zur Bewertung von Kampagnen oft nur Klicks, Conversions und Verweildauern zur Verfügung. Da die persönliche Interaktion vollständig fehlt, kann bzw. konnte bisher nicht objektiv beurteilt werden, wie eine Kampagne emotional ankommt, selbst wenn sie subjektiv emotional gestaltet wurde.

Im hier vorgestellten Projekt soll ein ML-Modell eingesetzt werden, das Emotionen automatisiert aus Gesichtsausdrücken direkt beim Betrachten der Werbemittel ableiten kann. So sollte ein konkret messbares Feedback entstehen, das sich ggf. mit anderen Metriken kombinieren lässt. Varianten von Werbekampagnen können objektiv mittels eines A/B-Tests in einer Testgruppe auf

ihre emotionale Wirkung hin getestet werden, was datenbasierte Entscheidungen vor der Veröffentlichung erlaubt.

Der Vorteil besteht vor allem darin, dass ein ML-Modell automatisiert und bei Bedarf live viele Bilder gleichzeitig auswerten kann, wohingegen manuelle Auswertungen zeitaufwendig, evtl. subjektiv beeinflusst und aufgrund personellen Aufwandes teuer sind. Zudem könnten die Tests vollständig unabhängig von Ort und Zeit durchgeführt werden. Mitglieder einer Testgruppe sollen idealerweise per Webcam ihres eigenen Devices beim Anschauen der Werbemittel fotografiert, die Bilder an das ML-Modell weitergeleitet und die emotionalen Reaktionen direkt ausgewertet werden.

Herausfordernd können hierbei vor allem die Punkte Datenschutz und Bildmaterial sein. Es müssen selbstverständlich Datenschutz- und ethische Richtlinien eingehalten werden. Da aber das Modell zunächst in Tests mit eingegrenzten Testgruppen eingesetzt werden soll, können diese gezielt aufgeklärt werden.

Um eindeutige Vorhersage-Ergebnisse zu erhalten, muss sichergestellt werden, dass die Gesichter auf dem Bildmaterial in bestimmter Größe, Klarheit und in Farbe abgebildet sind. Zudem werden vermutlich Bilder allein nicht umfassend die Emotionen analysieren können. Für den hier vorgestellten Zweck wird es sinnvoll sein, zusätzliche Informationen zur emotionalen Reaktion der Testpersonen einzuholen z.B. in Form einer kleinen Feedback-Skala, die dann gemeinsam mit den Ergebnissen des Modells ausgewertet wird.

Initialisierung und Daten

Das beschriebene Projekt lässt sich als ML-Problem im Bereich der Bildklassifikation operationalisieren. Anhand von Bildern von Gesichtern soll eine passende Emotion abgeleitet werden. Zunächst wurde versucht ein Modell der Multiclass-Klassifizierung von fünf Klassen bzw. Emotionen zu trainieren. In einem weiteren Versuch wurden Modelle der binären Klassifikation von zwei Emotionen trainiert. Für beide Klassifizierungsaufgaben wurden sowohl CNN-Modelle als auch Transfer Learning Modelle trainiert

Der hier verwendete Datensatz stammt von kaggle (Yadav, 2024). Der ursprüngliche Datensatz besteht aus 6 Ordnern, die jeweils die Label einer Emotion darstellen: Happy, Sad, Angry, Neutral, Surprised, Ahegao. Der Ordner Ahegao wurde von vornherein nicht genutzt. So sind 14.248 Bilder in 5 Klassen vorhanden. Alle Bilder liegen mit dem Farbwert RGB vor. Es sind die Formate JPEG, PNG und BMP in einer durchschnittlichen Größe von 412 x 450 Pixeln vorhanden. Um das Training für diese Arbeit effektiv zu halten, wurden für das erste Experiment aus jedem Ordner zufällig 500 Bilder zwischen 200 bis 600 Pixeln in Länge und Breite für Training und Tests ausgewählt

(emotion_dataset). Für das zweite Experiment wurden 2000 Bilder der Klasse „Sad“ und 1937 Bilder der Klasse „Happy“ verwendet (emotion_dataset_v2). Die Trainings- und Testbilder werden mittels tf.keras.utils.image_dataset_from_directory auf eine einheitliche Größe von 256 x 256 Pixeln für die CNNs und 224x224 für die Transfer Learning Modelle gebracht.

Die ML-Architektur ergibt sich wie folgt:

- Modelltyp: CNN || Transfer Learning CNN
- Framework: Tensorflow, Keras
- Eingabeschicht: 3-Kanal RGB, Rescaling 0 – 1
- Encoder (Backbone): 3× Conv2D + MaxPooling2D || MobileNetV2
- Decoder: Flatten + Dense || Pooling + Dense
- Ausgabeschicht: logits || softmax
- Verlustfunktion: SparseCategoricalCrossentropy (from_logits = True) || SparseCategoricalCrossentropy (from_logits = False)
- Optimierer: Adam
- Monitoring: Accuracy und Loss, visualisiert mittels Matplotlib

Modellierung

a) Vorverarbeitung

Zur Erstellung von zwei balancierten Sub-Datasets wurde im Notebook 01_vorarbeit.ipynb aus dem ursprünglichen Datensatz die .bmp-Bilder gelöscht und dann für emotion_dataset aus jeder Klasse zufällig, aber zur Sicherstellung der Reproduzierbarkeit mit Seed, 500 Bilder innerhalb der Pixel 200 bis 600 in Länge und Breite ausgewählt. Diese Auswahl wird noch einmal unterteilt in je einen train (je 400 Bilder) und einen test Datensatz (je 100 Bilder) und in einem gesonderten Ordner „datasets“ gespeichert. Für emotion_dataset_v2 wurden aus den Klassen „Happy“ und „Sad“ alle vorhandenen Bilder zwischen 200 bis 600 Pixeln in Länge und Breite ausgewählt, sodass 3937 Bilder blieben. Auch hier wurde noch einmal unterteilt in train (1549 bzw. 1600) und test (388 bzw. 400).

b) Training

Es werden vier verschiedene Notebooks genutzt, in denen getrennt nach Datensätzen und Modelltyp trainiert wird. Zunächst wurde je ein einzelnes CNN und ein einzelnes Transfer Learning Modell mit dem Datensatz emotion_dataset trainiert. Hierzu werden zunächst die Batchgröße (für beide Modelle 16), Bildhöhe und -breite (256 x 256) bestimmt. Nun wird der Datensatz aus dem Ordner train geladen und mittels image_dataset_from_directory sowohl das Resizing auf einheitliche Pixel, als auch ein Split für Training und Validierung vorgenommen. Hierauf folgt die

Konfiguration auf prefetch mit Autotune, um schnelleres Laden in der Pipeline zu gewährleisten. Um Overfitting zu reduzieren, wird ein Augmentation-Layer festgelegt, der kleine Transformationen wie horizontales Spiegeln, Rotationen und Zoom an den Bildern vornimmt. Für alle Transformationen wird eine geringe Rate von 10% festgelegt, da die Gesichter trotzdem gut erkennbar bleiben müssen.

Um Overfitting zu reduzieren, wird eine Drop-out Rate von 0.2 für alle Modelle festgelegt. Lt. Geron (2020, S. 368 ff.) macht es Sinn für CNN höhere Drop-out Raten zu nutzen. Insgesamt könne durch Dropout die Genauigkeit von Modellen noch einmal entscheidend gesteigert werden, da die Netze robuster werden und besser verallgemeinern. Hier wurde dennoch die niedrigere Rate von 0.2 genutzt, da eine höhere Rate in einem Versuch eine schlechtere Accuracy für die einzelnen Klassen hervorbrachte.

Für alle Modelle wird als Optimizer Adam bestimmt und als Loss-Funktion SparseCategoricalCrossentropy. Als Metrik wird zusätzlich die Accuracy bestimmt.

Für alle Modelle werden callbacks festgelegt: early stopping überwacht den Validation Loss und stoppt das Training, wenn dieser sich innerhalb von fünf Epochen nicht mehr verringert – dies soll das Modell davor bewahren „auswendig“ zu lernen; reduce_lr steuert die Lernrate und verlangsamt das Training (halbiert die Lernrate), wenn sich innerhalb von fünf Epochen der Validation Loss nicht verringert – dies hilft, das Training zu stabilisieren und die Leistung auf den Validierungsdaten in späteren Trainingsphasen zu verbessern. Lt. Geron (2020, S.328 und S.364 bis 368) ist die Lernrate der wohl wichtigste Hyperparameter, wobei Performance Scheduling, wie das hier implementierte und durch Keras sehr einfach anzuwendende, im Vergleich verschiedener Methoden recht gut abschneidet. Zuletzt wird als callback außerdem checkpoint genutzt, sodass das beste Modell bei entsprechend niedrigstem Validation Loss gespeichert wird.

Es wird für alle Modelle eine Obergrenze von 40 Epochen festgelegt, die aber keins der Modelle ausnutzt.

Um die Trainingsergebnisse zu überwachen, werden Accuracy und Loss jeweils für Training und Validation mittels Matplotlib als Liniengraphen dargestellt.

Die Modelle, die mit emotion_dataset_v2 trainiert wurden, werden mittels einer for-Schleife, die über drei verschiedene Batchgrößen (16, 32, 64) iteriert, in einem Notebook für CNNs und einem weiteren für Transfer Learning nach demselben Vorgehen wie zuvor beschrieben, trainiert. Lt. Geron (2020, S.328) ist die Batchgröße, ebenso wie die oben genannte Lernrate, ein weiterer entscheidender Hyperparameter. Es wird von Hinweisen berichtet, dass große Batchgrößen zu Instabilitäten im Training und schlechterer Generalisierung führen, während kleinere Batchgrößen

zu besseren Modellen in kürzerer Trainingszeit führten. Da aufgrund der vorhandenen Hardware ganz große Batches nicht möglich sind, wird hier mit einer Auswahl im Bereich zwischen kleinen und größeren Batches experimentiert.

Für das Transfer Learning wird MobileNetV2 als Backbone genutzt. Dieses Modell wurde ausgewählt, da es lt. allgemeiner Recherche wenig Rechenleistung verbraucht und somit für ein schnelles Training ohne GPU geeignet ist. Außerdem soll es oft eine vergleichbare Genauigkeit bei Aufgaben mittlerer Komplexität im Vergleich zu größeren Modellen erreichen und besonders für kleine und mittlere Datensätze geeignet sein und dennoch weniger Overfitting zeigen. In der Studie von Bhagat et al. (2023) zum Thema „Facial Emotion Recognition using CNN“ wurden zudem verschiedene andere übliche Modelle zum Transfer Learning ausprobiert, die alle keine überzeugenden Ergebnisse zeigten, sodass hier explizit ein anderes Modell ausprobiert werden sollte.

c) Evaluation

Zur Evaluation werden die Bilder aus dem test-Ordner mittels image_dataset_for_directory und den bekannten Pixel- und Batchgrößen geladen. Es werden die Labels und Predictions gesammelt und in Konfusionsmatrizen als Heatmaps dargestellt. Zudem werden Klassifikations-Reports mit den Metriken precision, recall, f1-Score und accuracy ausgegeben. Zum Abschluss werden für eine Visualisierung der Vorhersagen zufällig immer zwei Bilder aus dem Testdatensatz ausgewählt und inklusiv der durch das Model vorhergesagten Klasse und Wahrscheinlichkeit mittels Matplotlib dargestellt.

Abbildung 1: Vorhersagen des Modells

Inbetriebnahme

Das Modell soll eingesetzt werden, um die emotionale Wirksamkeit von Werbemittel-Alternativen zu vergleichen. Eine API wurde bisher nicht realisiert, die Inbetriebnahme könnte wie folgt aussehen: Testpersonen, die über das Vorgehen aufgeklärt wurden und ausdrücklich eingewilligt haben, sollen nacheinander auf ihrem eigenen Gerät zwei Werbemittel (z. B. Standbild-Anzeigen vs. kurze Videos) gezeigt werden. Währenddessen soll über eine Webanwendung im Browser (hierzu könnte ein Link per E-Mail verschickt werden) mithilfe der eigenen eingeschalteten Kamera das Gesicht in automatischen Intervallen fotografiert werden. Hier müssen die Testpersonen einmalig per Browserdialog für den Kamerazugriff und das Versenden der Bilder einwilligen. Die Fotos würden dann automatisch per http POST an einen FastAPI-Endpunkt (/predict) auf dem Server der Testenden gesendet. Die Bilder würden im JPEG versendet und empfangen werden. Die Übertragung sollte über das Internet erfolgen. FastAPI würde die Bilder entgegennehmen und sie temporär im Arbeitsspeicher speichern. Sie würden dann in ein standardisiertes Format konvertiert und auf die Eingabegröße des Machine-Learning-Modells skaliert. Dabei könnte direkt sichergestellt werden, dass das Farbschema mit dem Modell kompatibel ist. Die so vorverarbeiteten Bilder würden dem in FastAPI geladenen Modell übergeben, sodass es Wahrscheinlichkeiten für die Emotionen berechnen kann. Die Vorhersagen sollen nicht an die Testperson zurückgesendet werden, die Antwort auf den POST-request könnte standardisiert ein http-Status 204 sein. Die Vorhersagen sollten stattdessen dann inkl. Testperson- und Bild-ID sowie weiteren Metadaten in eine angebundene Datenbank für die spätere Auswertung gespeichert werden. Ob die Bilder ebenfalls separat gespeichert werden können, müsste noch datenschutzrechtlich überprüft werden.

Ergebnisse

Zur Überwachung des Trainings wurden Accuracy und Loss genutzt. Hier zeigt sich, dass das Transfer Learning Modell mit 5 Klassen mit einer Accuracy von 0.5 über fast den gesamten Lernprozess und einem zwar sinkenden Loss, aber deutlich höheren Validation Loss von minimal 1.2 nur rät und zusätzlich auf die Trainingsdaten overfitted. Das CNN mit 5 Klassen schneidet schon etwas besser ab, aber auch hier zeigt sich Overfitting auf die Trainingsdaten mit höherem Validation Loss als Trainings Loss sowie eine recht niedrige Accuracy von 0.6. (siehe Visualisierungen und Report in den Notebooks 02_cnn_v1 und 03_transfer_v1).

Um die Performance zu verbessern wurde dann mit weniger Klassen und der vierfachen Bildmenge trainiert. Es wurden zwei Emotionen ausgewählt, die recht unterschiedliche Ausprägungen im Gesicht zeigen: bei „Happy“ sind häufig lächelnde Münder mit Zähnen zu sehen, bei „Sad“ dagegen häufig geschlossene Münder, weshalb ich annehme, dass zumindest in diesen Bild-Bereichen

regelmäßig Ähnlichkeiten innerhalb der Klassen bestehen, die diese leichter unterscheid- und lernbar machen. Sicher wird aber auch die erhöhte Trainingsmenge anteilig für bessere Ergebnisse gesorgt haben.

Im Folgenden werden nur die CNN-Ergebnisse bildlich gezeigt, da die Ergebnisse der Transfer Learning-Modelle mit unruhigen Trainingsverläufen, höherer Validation Accuracy als Trainings Accuracy sowie etwas schlechterer Performance auf die Testdaten auffallen. Hier könnte noch weiter experimentiert werden (vergleiche Visualisierung im Notebook 03_transfer_v2). Die CNN-Modelle lernen alle erfolgreich, wobei Batch 16 geringere Schwankungen aufweist. Alle zeigen die übliche etwas höhere Trainings Accuracy als Validation Accuracy, sowie einen etwas höheren Validation als Trainings Loss – der Unterschied bleibt aber so niedrig, dass nicht von starkem Overfitting auszugehen ist.

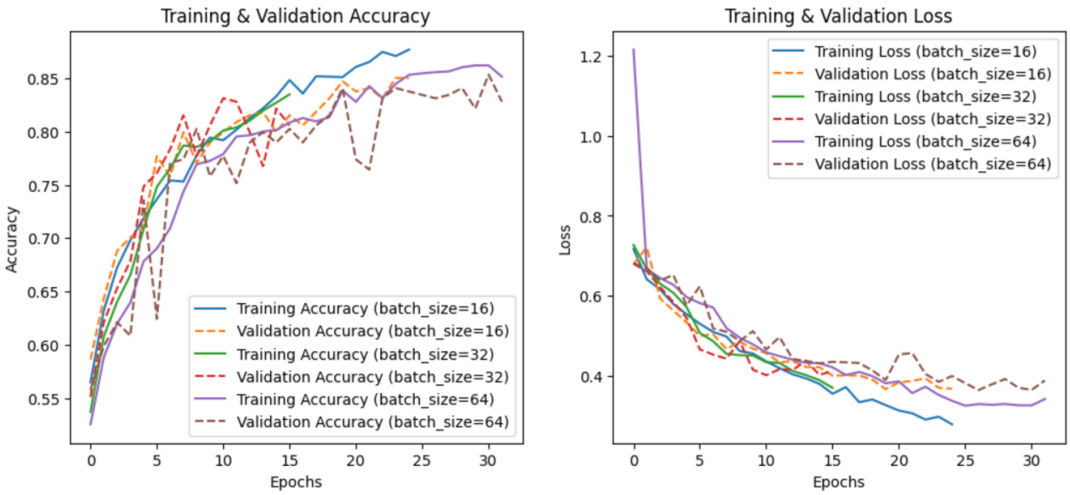

Abbildung 2: Training und Validation Accuracy und Loss für CNN-Modelle

Es zeigt sich auf die Testdaten für das CNN-Modell mit Batchgröße 16 die beste Performance. Die Klassen Happy und Sad werden mit 330 von 388 bzw. 362 von 400 Bildern korrekt vorhergesagt (siehe Abb. 2), was einer Precision von 0.9 bzw. 0.86 entspricht. Die Accuracy liegt bei 88% (siehe Abb. 3)

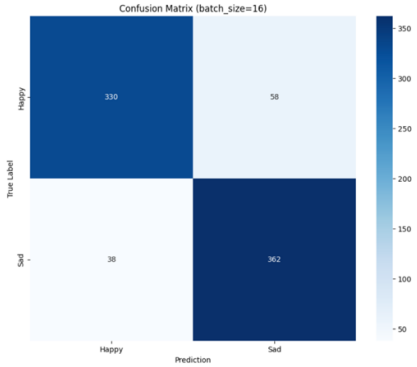

Abb. 3: Heatmap CNN, 2 Klassen (Batch 16)

	precision	recall	f1-score	support
Happy	0.90	0.85	0.87	388
Sad	0.86	0.91	0.88	400
accuracy			0.88	788
macro avg	0.88	0.88	0.88	788
weighted avg	0.88	0.88	0.88	788

Abb. 4: Klassifikationsreport CNN, 2 Klassen (Batch 16)

Die CNN-Modelle brauchten zwischen 16 bis 32 Epochen, die Transfer Learning Modelle zwischen 18 bis 25, waren aber dennoch etwas langsamer im Training. Bei den CNNs zeigt die Batchgröße deutlichere Auswirkungen auf die Performance als bei Transfer Learning. Sodass sich, wie bereits oben begründet, das CNN-Modell mit Batchgröße 16 recht deutlich als das vielversprechendste identifizieren lässt. Möchte man auch beim Transfer Learning ein „Bestes" benennen, wäre das mit der Batchgröße 32 mit im Vergleich etwas niedrigeren Loss-Werten auszuwählen – wobei das Transfer Learning Modell mit Batchgröße 64 im Training das stabilste war.

Schlussfolgerungen

Es stellt sich heraus, dass mit einfachen Methoden und einem kleinen Datensatz nicht so leicht ein gutes Modell trainiert werden kann. Die Reduzierung der Klassen und Erhöhung des Bildmaterials haben eine deutliche Verbesserung gebracht. In weiteren Experimenten sollte geprüft werden, ob es möglich ist, auch für mehr Klassen durch eine mindestens vierfache Vergrößerung des Datensatzes ein performantes Modell zu trainieren. Zusätzlich könnte getestet werden, ob Bilder mit Annotationen wie bounding boxes z.B. um Mund und Augen dabei helfen könnten, eine bessere Performance bei mehr zu lernenden Klassen zu erreichen, weil das Modell so auf die zur Unterscheidung wichtigen Bildausschnitte fokussiert werden kann.

Falls ohne Annotationen weiter experimentiert wird, wäre alternativ zu testen, ob Vorhersagen robuster werden, wenn Bilder im richtigen Format zentriert werden – das Model könnte dafür ggf. um einen resizing_layer erweitert werden, der crop_to_aspect_ratio nutzt.

Dass zwei deutlich zu unterscheidende Emotionen in den Experimenten für performantere Modelle gesorgt haben, lässt vermuten, dass es leichter ist, aus Bildern mit eindeutigen Gesichtszügen etwas vorherzusagen. Somit ist darauf zu achten, dass sowohl gutes Bildmaterial geliefert wird und ggf. zusätzlich eine Strategie entwickelt wird, um uneindeutige Bilder auszusortieren.

Eine Strategie zur Auswertung könnte sein, dass Bilder ab einer bestimmten hohen Vorhersagewahrscheinlichkeit zu einer Anzahl für die jeweilige Emotion zusammengezählt werden, sodass in Kombination mit einer Gesamtzahl an Bildern eine Ratio der jeweiligen Emotion über alle Bilder berechnet werden könnte.

Da Emotionen sich vielschichtig ausdrücken, sollte zur Bewertung idealerweise nicht allein das Ergebnis des Modells herangezogen werden, sondern dieses kombiniert werden mit z.B. zusätzlichem Feedback in Form einer Bewertungsskala oder einem kurzen Kommentar – beides ebenfalls automatisiert auswertbar.

Literaturverzeichnis

Bhagat D., Vagilb, A., Guptac, R.& Kumard, A. (2023). Facial Emotion Recognition (FER) using Convolutional Neural Network (CNN). *Procedia Computer Science, (Heft 235),* Seiten 2079-2089. Doi: https://doi.org/10.1016/j.procs.2024.04.197

Geron, A. (2020). Praxiseinstieg Machine Learning mit Scikit-Learn, Keras und TensorFlow - Konzepte, Tools und Techniken für intelligente Systeme. Heidelberg: O`Reilly.

Speck, A. (2017). Stärkere Werbewirkung durch emotionale Botschaften. Abgerufen am 10.05.2025 von https://www.springerprofessional.de/werbewirkungsforschung/konsumforschung/staerkere-werbewirkung-durch-emotionale-botschaften/12451504

Datensatz

Yadav. H., Facial Emotion Dataset. Abgerufen am 08.05.2025 von https://www.kaggle.com/datasets/himanshuydv11/facial-emotion-dataset/data