Montagsmaler mit KI

Grenzen des Image captioning?

In dieser Dokumentation wird die Frage “Grenzen des Image captioning?” behandelt.

Anhand mehrerer Versionen eines interaktiven Projektes (“Montagsmaler mit KI”) wird untersucht, welche Einschränkungen bei den automatischen Bildbeschreibungen auftreten und wie sich unterschiedliche Modellansätze auf diese Grenzen auswirken.

Das Konzept jeder Version ist gleich:

Ein Benutzer malt auf einer Online-Leinwand ein vorgegebener Begriff und die KI errät, was gezeichnet wurde.

Das Konzept ähnelt dem Spiel Montagsmaler.

Insgesamt wurden vier verschiedene Projekte mit ähnlichen Herangehensweisen erstellt.

Die Projekte wurden mir Python und einfacher Webentwicklung umgesetzt.

MMAI – Erster Versuch

https://github.com/AlyssaSmt/MMAI

Die erste Version des Montagsmalers mit KI (MMAI) ist ein Zeichnspiel, bei dem ein Benutzer auf einer Leinwand malt und die KI erraten soll, was gezeichnet wurde. Es beinhaltet Echtzeit vorhersagen und erlaubt es den Nutzer die Zeichnungen lokal zu speichern.

Die Funktionen beinhalten:

- Malen auf einer Leinwand
- Echtzeit KI vorhersagen
- Konfidenz und Top 3 vorhersagen
- Speichern und Galerie

Benutzung

Die Benutzung des Modells ist in der README-Datei beschrieben.

Tools und Bibliotheken:

- TensorFlow/Keras: Wird verwendet um ein Convolutional Neural Network (CNN) zu bauen und trainieren, das Zeichnungen erkennen kann.
- FastAPI: Ein einfaches Python-Webframework, das die Backend-REST-API für die Modellvorhersagen bereitstellt. Es betreibt einen HTTP-Server (über Uvicorn) zur Bearbeitung der Anfragen.
- Python (mit PIL, NumPy, etc.): Zur Bildbearbeitung und Datenbehandlung.
- HTML//CSS/JavaScript: Zum Erstellen einer einfachen Weboberfläche.

Modelarchitektur

Das in MMAI verwendete KI-Modell ist ein einfaches CNN, das Skizzen in einen festen Satz von Kategorien einordnet.
Für die erste Version des Projekts wurden zehn verschiedene Kategorien verwendet.

Datensatz und Training:

Der Datensatz stammt aus “Quick Draw!” von Google. Er beinhaltet eine Sammlung von über 50 Million Zeichnungen aus 345 Kategorien.

Der Quick Draw! Datensatz stellt Zeichnungen als Sequenz von Stiftstrichen in NDJSON-Dateien bereit.

Diese NDJSON Dateien werden des Python Programm “convert_ndjson_to_png.py” vorverarbeitet und in 64x64 Graustufenbilder umgewandelt. Das Programm liest für jede Zeichnung die Sequenz der Stiftstrichen aus und rendert diese zu einem Bild. Die Bilder werden anschließend auf den gezeichneten Bereich zugeschnitten und auf die Größe von 64x64 Pixeln skaliert. Das Programm ist begrenzt auf ein Maximum von 1500 Bildern pro Kategorie. Die erzeugten Bilder werden für weitere Verwendung in einem separaten Ordner gespeichert.

Das Trainieren des Modells erfolgt über das Python Skript “train_model.py”. Das Skript nimmt die zuvor konvertierten Bilder und teilt sie in Trainings- und Validierungsdaten (80/20-Aufteilung). Zusätzlich wird eine Normalisierung angewendet, um die Pixelwerte auf den Bereich [0,1] zu skalieren. Das Modell wird anschließend für 25 Epochen trainiert. Danach wird das besttrainierte Modell gespeichert. Das Ergebnis ist ein trainiertes CNN, welches Zeichnungen ihren jeweiligen Kategorien zuordnen kann.

Backend

Die Datei “main.py” lädt beim Start das trainierte CNN-Modell sowie die Klassenzuordnungen. Wenn eine Vorhersagenanfrage vom Frontend eingeht, wird die gleiche Vorverarbeitung durchgeführt wie im Training:

Das Bild wird dekodiert, in ein Graustufenbild umgewandelt und normalisiert. Anschließend sagt das CNN die Klassenwahrscheinlichkeiten für das Bild voraus. Anhand der Vorhersagewerte identifiziert das Backend daraus die wahrscheinlichsten Klassen.

Frontend

Im frontend steuert die JavaScript Datei “script.js” die Benutzerinteraktion und sendet die Zeichnung regelmäßig zur Vorhersage an das Backend. Wenn der Benutzer auf der Zeichenfläche zeichnet, erfasst die Anwendung die Stiche Bewegungen. Alle paar hundert Millisekunden wird der Inhalt der Zeichenfläche erfasst und an den API-Endpunkt gesendet.

Das Vorhersagen Ergebnis wird anschließend genutzt, um das UI zu aktualisieren. Dabei werden die Top-Vorhersage mit Konfidenz sowie die Top-3-Vorhersagen angezeigt.

Die Anwendung nutzt ein zufälliger Begriff der vorgegebenen Klassen zu Beginn jeder Runde. Wenn die Zeichnung mit der Modellvorhersage übereinstimmt, wird sie als korrekt erkannt. Anschließend kann der Benutzer das Bild speichern, welches rechts in einer Galerie angezeigt wird.

Erkenntnisse

Während der Erstellung des CNNs wurden einige Probleme erkannt. Zu Beginn hat die KI immer wieder das gleiche erraten. Dies lag daran, dass ursprünglich der Raw-Datensatz von Quick Draw! genutzt wurde. Dieser Datensatz enthielt viele ungeeignete Zeichnungen, welche zum größtenteils nur weißen Bildern waren oder nur einzelne Striche hatten. Besonders oft wurde die Klasse “String Bean” erraten, welche im Schwarz-Weiß-Format nur aus Strichen besteht. Dadurch wurde das Modell so trainiert, dass ein weißes Bild mit Strichen immer ein “String Bean” war.

Der erste Lösungsansatz war, die Klasse “String Bean” zu entfernen. Jedoch wurde danach häufig die Klasse “Sun” erraten, da diese ebenfalls nur aus einem Kreis mit Strichen besteht und vielen anderen Klassen ähnelt.

Die Lösung des Problems war die Verwendung der " Simplified Drawing Files” von Quick Draw!. In diesem Datensatz werden Zeichnungen an die linke obere Ecke des Bildes ausgerichtet, einheitlich auf 256x256 Pixel skaliert, auf 1-Pixel-Abstand resampled und mit dem Ramer–Douglas–Peucker-Algorithmus vereinfacht. Durch die Verwendung des neuen Datensatzes konnte das Modell deutlich besser trainiert und genutzt werden.

Es ist ebenfalls anzumerken, dass nicht alle Klassen für das Trainieren eines kleinen CNNs tatsächlich nützlich sind. Zum Beispiel Klassen wie “String Bean”, “Apple”, oder

“Circle” sind sehr allgemein und bestehen aus einfachen Strichen oder Kreisen. Dies führt dazu, dass vieles falsch eingeordnet wird.

MMAI2 - Einführung von CLIP zur Zero-Shot Recognition

https://github.com/AlyssaSmt/MMAI2

Die zweite Version des Montagmalers mit KI baut direkt auf der ersten Version auf. Das gleiche CNN-Modell wird weiterhin verwendet, jedoch wird zusätzlich ein zweites KI-Modell verwendet.

In dieser Version wurde das CLIP-Modell von OpenAI eingebunden, um Vorhersagen mit offenem Vokabular (Zero-Shot Recognition) zu ermöglichen. Das bedeutet, dass das System versucht, Skizzen außerhalb des begrenzten Klassensatzes zu erkennen, mit dem das CNN trainiert wurde.

MMAI2 führt für jede Zeichnung zwei Modelle parallel aus:

Zum einen das ursprüngliche CNN zur Klassifizierung innerhalb seiner trainierten Klassen, zum anderen ein vortrainiertes CLIP-Modell, welches die Zeichnungen mit beliebigen Textbeschreibungen vergleichen kann. Die Ergebnisse beider Modelle werden dem Benutzer angezeigt.

Model Architektur und Daten

Das CNN in MMAI2 ist im Wesentlichen dasselbe aus der ersten Version. Der einzige Unterschied besteht darin, dass es mit 29 Klassen statt nur 10 Klassen trainiert wurde.

Ursprünglich wurde diese Version genutzt, um den Unterschied zwischen nur 10 Klassen und 29 Klassen zu zeigen.

Im Nachhinein wurde zusätzlich ein weiteres Modell integriert, um den Unterschied zwischen zwei verschiedenen Modellansätzen (supervised CNN vs. Zero-Shot-Modell) zu zeigen.

Das CLIP-Modell ist vortrainiert, genauer gesagt verwendet es OpenAIs CLIP mit der ViT-B/32 Vision-Transformer-Architektur.

CLIP ist ein neuronales Netzwerk, das auf rund 400 Millionen Bild-Text-Paaren trainiert wurde und dadurch die Ähnlichkeit zwischen einem Bild und einer Textbeschreibung messen kann.

MMAI2 nutzt CLIP, um eine Bildbeschreibung für eine Zeichnung zu ermitteln, ohne dass das Modell explizit auf diese Zeichnungsklassen trainiert wurde.

Da CLIP prinzipiell jede Textbeschreibung bewerten kann, wurde eine feste Liste an Bildunterschriften definiert, aus der CLIP frei wählen kann. Zusätzlich wurden Bildunterschriften hinzugefügt, die nicht im ursprünglichen CNN-Datensatzes verwendet worden sind.

Dadurch ist es möglich, auch Objekte zu zeichnen, die nicht im CNN-Datensatz enthalten sind und dennoch korrekt vom CLIP-Modell erkannt werden (Bsp.: "a simple sketch of a person").

API und Backend

Die API stellt nun zwei Endpunkte bereit, zum einen das ursprüngliche CNN-Modell und zum anderen das neue CLIP-Modell.

Das gezeichnete Bild wird an CLIP übergeben, und das Modell beantwortet im Wesentlichen die Frage: “Wie ähnlich ist diese Zeichnung den einzelnen Bildunterschriften in der Liste?”.

CLIP berechnet einen Ähnlichkeitswert für jede Bildunterschrift, der mithilfe einer Softmax-Funktion in einen Konfidenzwert umgerechnet wird. Anschließend wird die Bildunterschrift mit der höchsten Übereinstimmung sowie die Top-3-Bildunterschriften mit ihren jeweiligen Konfidenzwerten ausgegeben.

Das Backend lädt beim Start das CLIP-Modell und tokenisiert alle potenziellen Bildunterschriften vorab. Dadurch können Text- und Bild-Embeddings schnell berechnen werden, ohne Textbeschreibungen erneut zu tokenisieren. Bild und Text werden zur Vergleichbarkeit in denselben Embedding-Raum kodiert. Der Code ermittelt anschließend die Kosinusähnlichkeit zwischen dem Bild und jedem Text-Embedding und verwendet diese, um Überreinstimmungen zu bestimmen.

Frontend

Das Frontend ist ähnelt ebenfalls die erste Version des MMAI. Der Benutzer zeichnet auf einer Leinwand und die KI versucht zu erraten, was gezeichnet wurde.

Das Ursprüngliche CNN gibt eine Vorhersage aus den zuvor definierten Klassen zurück. Darunter werden die Vorhersagen des CLIP-Modells mit den vordefinierten Bildunterschriften angezeigt. Beim Speichern einer Zeichnung werden sowohl die Top-Übereinstimmung des CNNs als auch die Top-Übereinstimmung des CLIP-Modells angezeigt.

Es wird weiterhin ein vordefinierter Begriff aus der Liste des CNNs vorgegeben. Zusätzlich kann der Benutzer jedoch auch Begriffe zeichnen, die nur für das CLIP-

Modell genutzt wurden. In diesem Fall wird nicht mehr bewertet, ob das gezeichnete mit dem vorgegebenen Begriff übereinstimmt bzw. “richtig” ist.

Erkenntnisse

Das Hinzufügen von weiteren Klassen des CNNs hat das Modell nicht stark verändert. Es wurde meistens das richtige vorhergesagt, jedoch musste oft mehr gezeichnet werden, um eine höhere Übereinstimmung zu bekommen.

Die Erkennung des CLIP-Modells war ebenfalls oft erfolgreich, allerdings ist CLIP nicht speziell auf schwarz-weiß Zeichnungen trainiert. Daher musste in vielen Fällen detaillierter gezeichnet werden, damit das Modell das Objekt korrekt erkennen konnte.

Im direkten Vergleich hat das selbst trainierte CNN-Modell schneller das gezeichnete erkannt als das CLIP-Modell.

MMAI3 – Erweiterte Klassen Kategorien

https://github.com/AlyssaSmt/MMAI3

Die Dritte Version des Modells erweitert primär die Anzahl der CLIP-Bildbeschreibungen. Es wurde getestet, wie gut CLIP mit einer deutlich größeren Anzahl an vorgegebenen Bildbeschreibungen umgehen kann. Insgesamt wurden etwa 140 weiter Bildbeschreibungen hinzugefügt. Diese Bildbeschreibungen wurden in einer “captions.txt” Datei ergänzt.

Die Anwendung MMAI3 ist eine minimale Erweiterung von MMAI2.

Erkenntnisse:

Während das CNN weiterhin stabile Ergebnisse liefert, zeigt CLIP bei größerem Vokabular mehr Unsicherheit und Genauigkeit. Außerdem weist das CLIP-Modell eine größere Verzögerung bei der Vorhersage auf als in MMAI2.

MMAI4 – Open Vocabulary Erkennung mit OpenCLIP

https://github.com/AlyssaSmt/MMAI4

In der letzten Version des Projektes wurde ausschließlich mit CLIP gearbeitet.

Das selbst trainierte CNN wurde vollständig entfernt, wodurch das Projekt zu einer Art Open-Vocabulary-KI weiterentwickelt wurde. Es benutzt ein vortrainierte CLIP-Modell, welches bereits in MMAI2 und MMAI3 verwendet wurde.

Wesentliche Veränderungen

Da ausschließlich mit CLIP gearbeitet wurde, muss das Modell nicht mehr trainiert werden. Stattdessen vergleicht das Modell die Skizzen mit einer größeren Liste an Wörtern.

Damit es nur Wörter verwendet werden, welche zeichenbar sind, wurden die Kategorien des Quick Draw!-Datensatzes verwendet. Insgesamt stehen dem Modell 345 verschiedene Wörter zur Auswahl. Dieser Datensatz kann jedoch mit einem anderen Datensatz von Wörtern ersetzt werden.

Ebenfalls wurde das UI minimal angepasst. Die Live-Vorhersage kann nun ein- und ausgeschaltet werden. Dies ermöglicht es dem Nutzer ein Bild zu zeichnen, ohne von den Vorhersagen abgelenkt zu werden. Außerdem werden statt der Top-3-Ergebnissen jetzt die Top-5-Ergebnisse angezeigt. Dies ist zur Visualisierung der Genauigkeit des Modells gedacht.

Erkenntnisse

Da diese Version mit einer deutlich größeren Anzahl an Wörtern arbeitet, hat das Modell größere Schwierigkeiten, die korrekte Vorhersage zu treffen.

Ursprünglich wurden 10.000 Wörter aus dem englischen Vokabular verwendet. Diese erwiesen sich jedoch als ungeeignet für das Modell, da diese Wörter zum größtenteils nicht Skizzierbar waren. Dies führte dazu, dass die Vorhersagen kaum aussagekräftig waren.

Anschließend wurde eine KI-generierte Liste an Wörtern mit rund 700 skizzierbaren Wörtern verwendet. Mit dieser Liste an Wörtern war es möglich, ein aussagekräftiges Bild zu zeichnen und eine sinnvolle Vorhersage zu bekommen. Die Ergebnisse waren jedoch häufig ungenau, insbesondere wenn die Zeichnung sehr abstrakt war oder wenn sich Begriffe stark ähnelten (Bsp.: “Soccer Ball” und “Baseball”).

Zuletzt wurde die Liste der Wörter des Quick Draw!-Datensatzes verwendet. Diese Liste enthält eine große Anzahl an skizzierbaren Wörtern und war somit besonders geeignet für die Anwendung. Da sie auf 345 Wörter begrenzt ist, waren die Vorhersagen des Modells deutlich stabiler und genauer.

Gesamtergebnis

Das Projekt untersucht die Grenzen des Image Captionings mithilfe eines Montagmaler-Spiels mit KI. Hierfür wurde Vier verschiedenen Versionen entwickelt und getestet.

Beschränkung der Captions:

In CNN-basieren Modellen kann eine Vorhersage nur innerhalb trainierter Klassen gemacht werden. Zeichnungen außerhalb der vordefinierten Bildunterschriften erhalten jedoch trotzdem eine Caption, die meistens nicht korrekt ist. Dies verdeutlicht die “Closed-World”-Annahme klassischer, supervised Captioning-Systeme.

Visuelle Abstraktheit:

Oft sind Zeichnungen sehr abstrakt und können mehrdeutig sein. Alle Versionen der Anwendung zeigen oft Unsicherheit bei visuell ähnlichen Klassen (Bsp.: “Ball” und “Apple”).

Dies zwingt die Modelle eine Vorhersage zu treffen, obwohl diese mit hoher Wahrscheinlichkeit nicht korrekt ist.

Empfindlichkeit bei der Vorverarbeitung:

Das Ergebnis der Vorhersage der CNN-Modelle hängen stark von der Vorverarbeitung und des Datensatzes ab. Durch die Verwendung eines ungenauen Datensatzes kann das Modell nicht richtig trainiert werden und die Vorhersagen sind ungenau.

“Open Vocabulary” beseitigt Unsicherheiten nicht:

MMAI4 reduziert zwar die Klassenbeschränkungen, jedoch bleibt das gewählte Vokabular eingeschränkt. Je mehr Klassen verwendet werden, desto geringer ist die Genauigkeit der Vorhersage.

Fazit:

Das Projekt zeigt, dass Image Captioning nicht nur durch die Modellwahl, sondern auch durch den Datensatz, die Vorverarbeitung und auch das genutzte Vokabular abhängig ist. Verschiedene Modelle erhöhen zwar die Flexibilität, jedoch bleiben alle Modelle zu einem gewissen Grad beschränkt.

Quellen

https://github.com/AlyssaSmt/MMAI

https://github.com/AlyssaSmt/MMAI2

https://github.com/AlyssaSmt/MMAI3

https://github.com/AlyssaSmt/MMAI4

https://quickdraw.withgoogle.com/

https://github.com/googlecreativelab/quickdraw-dataset

https://github.com/first20hours/google-10000-english/blob/master/google-10000-english-no-swears.txt

https://github.com/openai/CLIP