Dokumentation

CoinCat

Version 1

16.01.2023

Tarik Glasmacher / Tim Bering

Inhalt

Übersicht

CoinCat ist eine in Matlab geschriebene Software, die dazu dient, kleinere Münzgeldmengen zusammenzuzählen und mit verschiedenen Kriterien zu filtern. Durch ein eigens trainiertes neurales Netzwerk werden Münzen erkannt und einem entsprechenden Wert zugewiesen. Sie wurde im Zuge des Moduls **Bildverarbeitung** an der FH Aachen im Wintersemester 2022/2023 als freiwilliges Abschlussprojekt entwickelt.

Features

- Unterstützung für das Einbinden von eigenen Modellen (serialzedNet.mat ersetzen)
- Klassifikation durch ein neurales Netzwerk
- Grafische Benutzerschnittstelle für Benutzer
- 4 anpassbare Filteroptionen

Konfiguration

Um alle Funktionalitäten der Software nutzen zu können, müssen folgende Abhängigkeiten installiert werden:

- Computer Vision Toolbox
  - Um Bildoperationen ausführen zu können.
- Deep Learning Toolbox
  - Zum Trainieren und Anwenden der neuralen Netzwerke.
- Parallel Computing Toolbox
  - Zur Beschleunigung des Netzwerk-Trainings über Parallel-Pooling.

Die Verlinkungen auf die Matlab-Artikel für weitere Hilfe bezüglich dieser Toolboxen finden sich in der README.md.

Startanleitung

1. Abarbeiten der **Konfiguration** (einmalig).
2. Öffnen der Datei GUI.mlapp.
3. Starten über den Start-Knopf von Matlab.

Aufbau des Projektes

Dataset

Hier sind in Unterordnern die Trainings- und Validierungsdaten gespeichert. Jede Münzklasse ist dabei mit einem einzigartigen Label versehen, welches aufsteigend mit dem Wert der jeweiligen Münzklasse notiert ist. Das Bild “orig.jpg“ in jedem Unterverzeichnis wird durch einfache Bildoperationen verändert, um mehr Trainings- und Validierungsdaten zu erzeugen. Eigene Daten können hier ebenfalls abgelegt werden, da das Programm mit imageDatastores arbeitet, die alle Dateien in den entsprechenden Pfaden auslesen.

Unterordner	Münzwert
0	0,01€
1	0,02€
2	0,05€
3	0,10€
4	0,20€
5	0,50€
6	1,00€
7	2,00€

Tab 1.: Unterordner-Münzwert-Zusammenhang

Dokumentation
Hier findet sich die Dokumentation zum Projekt.

Logo
Hier finden sich zwei Versionen des Logos von **CoinCat**, die ihren Nutzen in der Software und in der Taskleiste beim Ausführen der Software finden.

Münzen
[OBSOLET]
Bilder der europäischen Zentralbank von jeder Münzklasse. Werden in kopierter Form weiterverwendet, aber diese Dateien haben keinen Einfluss auf das Programm.

Programm
Wird das Programm erweitert, können Dateien hier abgelegt werden.

Präsentationen
Alle Präsentationen dieses Projekts sind hier zu finden. Auch die Abschlusspräsentation liegt hier.

Pythonprogramm
[OBSOLET]
Während der Entwicklung sind wir anfangs auf Fehler mit der Hough-Kreisdetektion gestoßen, die nach Fehlern von Matlab aussahen. Deshalb versuchten wir, das Projekt in Python neuzuschreiben. Dieser Ansatz wurde jedoch verworfen. Die Inhalte dieses Ordners haben keinen Einfluss auf das Programm.

readCoins
Hier werden die Teilbilder, welche aus dem Originalbild gewonnen werden, abgespeichert. Ein erneutes Laden eines Bildes löscht die alten Dateien aus diesem Ordner.

DOG_neural_network.mlx
[OBSOLET]
Ein alter Lösungsansatz für das neurale Netzwerk (FF-Netzwerk).

GUI.mlapp
Hier liegt die grafische Benutzerschnittstelle für die Software. Zum Starten der Software muss diese Datei geöffnet und durch das Klicken auf den grünen Start-Pfeil ausgeführt werden.
Durch UI-Callbacks wird hier auf die sich in den .m-Dateien befindlichen Methoden zugegriffen, um den Nutzerfluss zu erzeugen.

PostworkFilterType.m

Die verschiedenen Filtertypen liegen hier. Falls eine Erweiterung der Filterfunktionen erwünscht ist, sollte diese hier als Eintrag im Enum hinzugefügt werden.

SURFHarris.m

[OBSOLET]

Anfangs versuchten wir, die Klassifikation der Münzklassen mit Hilfe des SURF-Algorithmus durchzuführen. Wegen der vielen Feinheiten der Münzen war dies allerdings nicht erfolgreich.

coinClassification.asv

Siehe *coinClassification.m*.

coinClassification.m

Hier wird das in serializedNet.mat abgespeicherte neurale Netzwerk net eingesehen und zur Klassifikation von eingehenden Teilbildern verwendet. Außerdem wird der Wert der Teilbilder hier zusammengezählt. Die Datei GUI.mlapp greift darauf zu.

coinCounter.mlx

[OBSOLET]

Ein erster Versuch, die Münzen (damals noch anhand eines Referenzobjektes) innerhalb eines Bildes zu erkennen.

coinsCalibrateWhite.jpg

[OBSOLET]

Zugehörig zu coinCounter.mlx. Teil des dortigen Live-Skript-Versuches, die Münzen (damals noch anhand eines Referenzobjektes) innerhalb eines Bildes zu erkennen.

cut.m

Hier liegt die Logik, um ein eingegebenes Bild in Teilbilder zu unterteilen und in Zellen zu verstauen, in welchen sich weitere Information, wie die Position und der Radius einer Münze befinden. Die Wertzuweisung durch die Klassifikation findet in coinClassification.m statt. Die Datei GUI.mlapp greift darauf zu.

dingus_logo.png

[OBSOLET]

Das alte Platzhalterlogo für **CoinCat**.

ids.txt

[OBSOLET]

Diente zur Erinnerung an den Zusammenhang von Unterordner-Namen und dem entsprechenden Münzwert. Weiteres siehe Dataset.

image_classification.mlx

Das Live-Skript zum Testen der Klassifikation sowie dem Trainieren des neuralen Netzwerkes. Zum Trainieren des neuralen Netzwerkes müssen die entsprechenden Abschnitte des Live-Skripts ausgeführt werden.

main.mlx

[OBSOLET]

Das Live-Skript zum Testen der cut.m-Funktionalität. Weiteres siehe cut.m.

muenzen.jpg

[OBSOLET]

Test-Eingabedaten. Werden nicht mehr verwendet.

muenzenVerdeckung.jpg

[OBSOLET]

Test-Bild, was zur Veranschaulichung verwendet wird, da es die Kriterien für ein in der Anwendung aufgenommenes Bild erfüllt. Wird nicht verwendet.

neural_network_stacked_autoencoder.mlx

[OBSOLET]

Ein alter Lösungsansatz für das neurale Netzwerk (Stacked Autoencoder).

postwork.m

Hier liegt die Logik, um die Nachbereitung der Bilder auszuführen. So werden durch diesen Code die Überlagerungs-Kreise angezeigt und Filter angewandt. Die Datei GUI.mlapp greift darauf zu.

serializedNet.mat

Das neuronale Netzwerk wird aus dieser Datei mit dem Variablennamen *net* ausgelesen. Die Datei kann durch eigene neuronale Netzwerke ersetzt werden. Die Datei coinClassification.m greift auf diese Datei zu.

serializedNet_Backup_2023_01_05.mat

Ein Backup eines neuronalen Netzwerkes. Wurde manuell erstellt. Wird nicht verwendet.

test_tim.mlx

[OBSOLET]

Testdatei. Wird nicht verwendet.

Wie es funktioniert

1. Ein Bild wird eingelesen. Der Nutzer zieht eine Linie über den Durchmesser einer beliebigen Münze. Je besser die Münze zu sehen ist, umso besser das zu erwartene Ergebnis.
2. Das gesamte Eingangsbild wird klassifiziert. Die Ergebnisse werden in der grafischen Benutzerschnittstelle angezeigt.

Nutzen

Als für mobile Endgeräte kann **CoinCat** in Alltagssituationen helfen, so zum Beispiel, um genügend Geld für den Parkautomaten aus dem mitgeführten Kleingeld hervorzuheben.

Beispieldaten

Beispiele für einlesbare Bilder finden sich im Ordner Beispiele.

Projektpräsentation

Die Projektpräsentationen finden sich im Ordner Präsentationen.

Lizenz