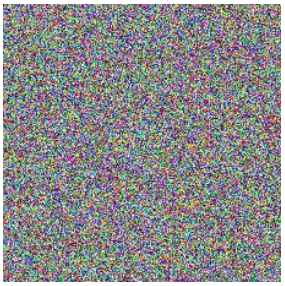
Irrbilder

Künstliche Intelligenz ist ein Teilgebiet der Informatik dass sich mit der Automatisierung intelligenten Verhaltens und dem maschinellen Lernen befasst. Bereits in den frühen 1940er Jahren setzte in diesem Bereich das Interesse für künstliche neuronale Netze (englisch *artificial neural network*) ein. Künstliche neuronale Netze folgen ihrem biologischen Vorbild und sind Netze aus künstlichen Neuronen. In jüngster Zeit wurden viele herausfordernde Anwendungen mit diesem Ansatz besser als mit konkurrierenden Lernverfahren gelöst. Insbesondere mit Deep Learning, künstlichen neuronalen Netzen mit zahlreichen Zwischenlagen, lassen sich Aufgaben lösen für die sich nur schwer mathematische Regeln formulieren lassen. Ein Beispiel dafür ist das Erkennen von Verkehrszeichen[1,2], eine der Kernaufgaben im Bereich des autonomen Fahrens[3].



Verkehrszeichen werden vom Menschen intuitiv erkannt und sicher von anderen Bildern unterschieden. Aktuelle Forschungsergebnisse[4,5] zeigen allerdings, dass sich, im Gegensatz zu einem menschlichen Betrachter, künstliche neuronale Netze überlisten lassen und in speziell konstruierten Irrbildern reale Objekte wie zum Beispiel Verkehrsschilder erkennen. Wird nun ein solches Irrbild zum Beispiel als Aufkleber auf der Rückseite eines LKWs angebracht kann dieses Bild, ohne von einem Menschen als Gefahr erkannt zu werden, Fehlverhalten von autonom fahrenden Fahrzeugen hervorrufen: Schwerste Unfälle können die Folge sein. Mit diesem aktuellen Problem beschäftigt sich der diesjährige informatiCup.

Aufgabe

Implementieren Sie eine Software die Bilder generiert mit denen sich ein gegebenes neuronales Netz für die Erkennung von Verkehrszeichen (siehe unten) überlisten lässt. Die von Ihrer Software generierten Bilder sollen intuitiv einem Verkehrszeichen möglichst unähnlich sein aber trotzdem mit einer hohen Konfidenz von dem künstlichen neuronalen Netz als solches erkannt werden.

---

[1] https://de.wikipedia.org/wiki/Verkehrszeichen
[2] https://e-archivo.uc3m.es/bitstream/handle/10016/7110/road_escalera_TIE_1997.pdf
[3] https://de.wikipedia.org/wiki/Autonomes_Fahren
[4] https://arxiv.org/abs/1412.1897
[5] http://dx.doi.org/10.13140/RG.2.1.1402.7760

Das neuronale Netz

Für diese Aufgabe wurde ein künstliches neuronales Netz mit dem German Traffic Sign Recognition Benchmark[6] trainiert. Dieser Datensatz umfasst über 50.000 Bilder von mehr als 40 verschiedenen Verkehrszeichen und steht frei zur Verfügung[7]. Das für diese Aufgabe mit diesem Datensatz trainierte neuronale Netz hingegen ist eine Black Box. Allerdings können Sie, wie im folgenden beschrieben, Anfragen an das neuronale Netz senden und erhalten als Antwort die Konfidenz mit der das eingereichte Bild als Verkehrszeichen klassifiziert wurde.

Ein- und Ausgabe

Klassifizierungsanfragen an das zu überlistende neuronale Netz können unter https://phinau.de/trasi gestellt werden. Mit Ihrer Anmeldung zum Wettbewerb erhalten Sie dafür einen API-Schlüssel. Erwartet wird ein HTTP POST-Request mit Encoding-Type `multipart/form-data` und den folgenden Parametern.

- `key`: der API-Schlüssel
- `image`: ein 64x64 PNG-Bild

Alternativ, insbesondere zu Testzwecken, kann die Web-Oberfläche genutzt werden. Es werden die folgenden HTTP-Statuscodes zurückgegeben.

- `200 OK`: In diesem Fall wird die Klassifizierung im Body der Response als JSON zurückgegeben. Beispiel:

  ```
  [ {"class": "Vorfahrt", "confidence": 0.7},
    {"class": "Gefahrenstelle", "confidence": 0.5} ]
  ```

- `400 Bad Request`: Bei fehlerhaften Anfragen wird im Body der Response eine kurze Begründung angegeben.
- `401 Unauthorized`: Der API-Schlüssel ist ungültig.
- `429 Too Many Requests`: Sie können pro API-Schlüssel maximal 60 Anfragen pro Minute stellen. Diese Grenze wurde überschritten.
- `503 Service Unavailable`: Der Server ist ausgelastet und kann keine Anfragen beantworten.

[6] http://dx.doi.org/10.1016/j.neunet.2012.02.016
[7] http://benchmark.ini.rub.de

Bewertung

Erwartet wird eine Software, die für mindestens fünf verschiedene Verkehrszeichen aus dem German Traffic Sign Recognition Benchmark Bilder ausgibt, die mit einer Konfidenz von mindestens 90% von dem neuronalen Netz (siehe oben) als Verkehrszeichen erkannt werden und dabei für einen menschlichen Betrachter dem erkannten Verkehrszeichen möglichst unähnlich erscheinen.

Neben der Software erstellen Sie bitte eine Ausarbeitung, welche die Installation und Bedienung Ihrer Software, sowie Ihren theoretischen Lösungsansatz beschreibt.

Ihre Einreichungen werden dann ganzheitlich bewertet. Neben der Güte der Lösung werden der theoretische Lösungsansatz, die Form der Ausarbeitung, die Softwarearchitektur und -qualität, mögliche Erweiterungen und, wenn Sie Ihre Lösung im Finale vorstellen dürfen, die Qualität der Präsentation bewertet.

Im Anhang finden Sie eine Checkliste der Bewertungskriterien. Nutzen Sie diese Liste um die Vollständigkeit Ihrer Lösung zu überprüfen.

Referenzplattform

Die Juroren werden Ihre Software aus den Quelltextdateien und den referenzierten Bibliotheken bauen und ausführen. Für diesen Wettbewerb sind die folgenden drei Zielplattformen zugelassen.

- Ubuntu 18.04 LTS AMD64
- Windows 10 x64, Release 1803 (10.0.17134.319) oder neuer
- macOS 10.13.6, Release 17G2208 / 15P6703 oder neuer

Liefern Sie zusammen mit Ihrer Software bitte eine vollständige Beschreibung, wie ein Juror ausgehend von den Auslieferungszuständen der obigen Referenzplattformen systematisch und wiederholbar eine gebaute Software erhält. Gerne können Sie auch ein Docker[8] Image bereitstellen.

Außerdem

Die FAQs zum laufenden Wettbewerb sowie Beispieleingaben und -ausgaben finden Sie in Kürze online auf https://github.com/InformatiCup/InformatiCup2019

[8] https://de.wikipedia.org/wiki/Docker_(Software)

Checkliste Bewertungskriterien

Theoretischer Ansatz

Der theoretische Ansatz muss in einer Ausarbeitung, die zusammen mit der Implementierung eingereicht wird, dargestellt werden. Bewertet werden sowohl der Inhalt als auch die Form.

Theoretische Ausarbeitung

Die theoretische Ausarbeitung soll die verwendete Theorie beschreiben.

- ❏ **Hintergrund**: Der Hintergrund des Lösung. Welche theoretischen Ansätze wurden verwendet? Warum wurden diese verwendet?
- ❏ **Auswertung**: Wie gut ist die Qualität der Lösung? Nach welchen (wissenschaftlichen) Kriterien wurde bewertet? Warum wurden diese Kriterien verwendet?
- ❏ **Diskussion**: Wie "gut" (auch außerhalb der reinen Qualität) ist die Lösung? Was für Probleme gibt es noch? Wie lässt sich die Lösung praktisch einsetzen?
- ❏ **Quellen**: Wurden (wissenschaftliche) Quellen richtig und angemessen verwendet?

Formalien

Eine gute Form ist entscheidend für die Lesbarkeit einer Ausarbeitung. Beachten Sie deshalb neben dem reinen Inhalt Ihrer Ausarbeitung auch einige Formalien.

- ❏ **Rechtschreibung**: Rechtschreibung und Grammatik sind korrekt.
- ❏ **Lesefluss**: Es gibt einen Lesefluss in der Ausarbeitung.
- ❏ **Layout**: Das Dokument hat ein einheitliches Layout. Dieses kann frei gewählt werden, darf aber nicht den Lesefluss stören.
- ❏ **Zitate**: Es wird richtig und einheitlich zitiert.
- ❏ **Quellenangaben**: Quellen sind richtig und einheitlich angegeben.

Softwarearchitektur und -qualität

Da eine etablierte Softwarearchitektur nur mit hohem Aufwand zu ändern ist, sollte sie besonders gründlich durchdacht und ausführlich begründet werden. Ausgewählte Aspekte der Softwarequalität sind für die Bewertung von besonderer Bedeutung. Gerne dürfen auch hier nicht genannte Aspekte aus den sehr weiten Feldern Softwarearchitektur und -qualität beleuchtet werden.

- **Architektur**: Beschreibung der Komponenten und deren Beziehungen
- **Software testing**[9]: Begründetes Konzept, Umsetzung
- **Coding conventions**[10]: Begründetes Konzept, Umsetzung
- **Wartbarkeit**: Mit welchem Aufwand kann das System angepasst werden?

Handbuch

Das Handbuch beschreibt den Installationsprozess der Lösung auf dem Referenzsystem, insbesondere die Abhängigkeiten. Zudem wird die Benutzung erläutert.

- **Installation**: Empfohlen wird die genaue Angabe der erforderlichen Befehle oder die Bereitstellung eines Installations-Skripts.
- **Betrieb**: Welche konkreten Befehle sind erforderlich, um Ergebnisse zu erhalten. Hier sei erneut auf die Referenzplattformen hingewiesen.

Qualität der Lösung

Die Qualität der Lösung wird an der Konfidenz des neuronalen Netzwerks und der Originalität der erzeugten Bilder bemessen. Zur Erinnerung: Die Konfidenz muss mindestens 0.75 (75%) betragen, sonst wird die Lösung nicht angenommen.

- **Konfidenz**: Wie hoch ist die Konfidenz für die generierten Bilder?
- **Originalität**: Wie kreativ sind die generierten Bilder?

Erweiterungen

Erweitern Sie Ihre Lösung über die Anforderungen der Aufgabenstellung hinaus. Seien Sie kreativ!

Präsentation

Im informatiCup-Finale werden die besten Lösungen vor einer Fachjury präsentiert.

- **Foliendesign**: Sind die Folien ansprechend? Lenken sie nicht vom Inhalt der Präsentation ab?
- **Vortragsstil**: Weckt der Vortragsstil Interesse an der Präsentation?
- **Verständlichkeit**: Ist der Vortrag verständlich? Wird der Hintergrund der Lösung in einem angemessenen Tempo erklärt? Wird nötiges Vorwissen geschaffen?
- **Reaktion auf Nachfragen**: Können Nachfragen beantwortet werden?

[9] https://en.wikipedia.org/wiki/Software_testing
[10] https://en.wikipedia.org/wiki/Coding_conventions