Machine Learning for Computer Vision
Abschlussprojekt: Rindenklassifizierung

Paul Walker
Department of Computer Science
DHBW Stuttgart
inf20045@dhbw-stuttgart.de

Tom Hofer
Department of Computer Science
DHBW Stuttgart
inf20173@lehre-dhbw-stuttgart.de

Abstract

Diese Arbeit befasst sich mit der Entwicklung eines Machine Learning Modells zur Klassifikation von Rinde zu bestimmten Baumarten. Das Modell könnte in der Praxis beispielsweise in einer App zur Bestimmung der Baumart anhand eines Fotos der Baumrinde eingesetzt werden. Im Rahmen der Arbeit werden vier Modelle basierend auf den vortrainierten Modellen InceptionV3, MobileNetV2, VGG16 und VGG19 erstellt, getestet und gegenübergestellt. Für das Training der Modelle kommen ein Trainingsset mit Daten aus dem Internet und ein Validationset mit selbst erhobenen Daten zum Einsatz.

1 Einleitung

Unser Abschlussprojekt befasst sich mit der Entwicklung eines Machine Learning Modells zur Bestimmung der Baumart anhand eines Fotos der Baumrinde. Die Rinde eignet sich sehr gut für die Bestimmung der Baumart, da diese, anders als zum Beispiel Blätter oder Früchte, unabhängig von der Jahreszeit ist. Die Eingabe für unseren Algorithmus ist also ein Bild. Zur Klassifizierung der Baumart verwenden wir ein vortrainiertes Convolutional Neural Network, welches wir für unseren Anwendungsfall anpassen. Die Ausgabe, beziehungsweise Vorhersage unseres Algorithmus ist die Art des Baumes, dessen Rinde auf dem Foto abgebildet ist.

In der Praxis könnte ein solches Modell zum Beispiel in einer App eingesetzt werden, die Baumarten bestimmen soll. Das Erkennen der Baumart anhand der Rinde ist ein schwieriges Problem, da es teilweise große Unterschiede innerhalb einer Klasse gibt, während zwischen Bäumen verschiedener Klassen manchmal nur sehr kleine Unterschiede erkennbar sind. Abbildung 1 zeigt beispielhaft eine Kiefer, die sehr ähnlich zu einer Lärche aussieht und eine weitere Lärche, die sich von den anderen beiden Bäumen relativ stark unterscheidet. Diese starken Unterschiede liegen unter anderem am Alter des Baumes.

2 Stand der Technik

Die Rindenklassifikation ist kein neues Problem und es wurden bereits Ansätze sowohl mit klassischer Computer Vision [5], als auch mit Hilfe von Deep Learning entwickelt [4]. In einigen Ansätzen wird ein radial radial basis probabilistic network (RBPNN) anstatt einem convolutional neural network (CNN) verwendet [3]. Viele Paper, in denen ein CNN genutzt wird, verwenden ein vortrainiertes CNN [4], dass dann nur noch auf das Problem spezialisiert wird. Dieser Ansatz wird auch in dieser Arbeit verfolgt.

Kiefer Lärche Lärche

Abbildung 1: Schwierigkeit des Problems

3 Daten und Features

Wir nutzen 21491 Bilder aus dem Internet bestehend aus 8 verschiedenen Klassen. Die Daten stammen aus dem BARK-KR [6] Datenset, dem Tree Species Dataset [1] und dem BarkNet 1.0 [2] Datenset. Einige Beispiele dieser Daten sind in Abbildung 2 abgebildet. Neben diesen Daten nutzen wir auch 333 selbst gelabelte Bilder bestehend aus denselben 8 Klassen. Einige Beispiele sind in Abbildung 3 abgebildet.

Diese Daten werden aufgeteilt in ein Trainingsset bestehend aus 21169 Bildern aus den Internetdaten und zwei verschiedene Entwicklungsdatensätze, einem bestehend aus 322 Bildern aus den Internetdaten (dev-set 1) und einem bestehend aus den 333 selbst gelabelten Bildern (dev-set 2) Ein Testset haben wir nicht. Wir können also keine genaue Aussage über die tatsächliche Feldperformance unseres Modells treffen. Es werden zwei verschiedene Entwicklungsdatensätze genutzt, da die Daten aus dem Internet und die selbst gelabelten Daten stark verschieden sind. Das sorgt dafür, dass die Genauigkeit des Models auf den selbst gelabelten Daten durchgängig schlecht ist. Der Entwicklungsdatensatz aus dem Internet vereinfacht die Entwicklung, da auf diesem das Model eine höhere Genauigkeit hat und Overfitting damit schneller erkennbar ist.

Alle Bilder werden auf eine Größe von 512*512 Pixeln gebracht. Um Verzerrungen zu vermeiden werden die Grafiken zunächst durch Zuschneiden auf das richtige Seitenverhältnis gebracht. In unseren Datensätzen haben wir acht Klassen, beziehungsweise acht Baumarten. Die Datensätze aus dem Internet bieten zwar noch weitaus mehr Klassen, allerdings sind für diese keine selbst gelabelten Daten vorhanden. Zusätzlich mussten die Labels der Daten aus dem Internet angepasst werden, da diese die Bäume noch in genaue Unterarten einteilen. Für diesen Rinden-klassifikator ist allerdings nur die Überart relevant. So wird beispielsweise das Label Örientalische Weiß-Eiche - Quercus alienaïn unserem Datensatz einfach als Oak, also Eiche, gelabelt. Die vertretenen Baumarten sind Esche, Buche, Birke, Tanne, Lärche, Eiche, Kiefer und Fichte.

4 Methoden

Zur Erstellung unseres Modells verwenden wir Transfer learning. Das bedeutet wir verwenden ein bereits für Bilddaten Trainiertes CNN aus dem Internet und passen dieses auf unseren Anwendungsfall an. Die Anpassung erfolgt in den Ersten und letzten Schichten des Modells.

Die erste Schicht ist die Eingabeschicht, die ein Bild mit 512*512 Pixeln und drei Kanälen annimmt. Dazu wird eine Schicht hinzugefügt, die die Bilder von einem Wertebereich von 0 bis 255 auf einen Wertebereich von 0 bis 1 transformiert. Für die Ausgabe des Modells werden vier Schichten am Ende hinzugefügt. Zunächst erfolgt ein zweidimensionales global average pooling, gefolgt von einer flatten und dense Schicht. Die dense Schicht verwendet eine Relu als Aktivierungsfunktion. In der letzten Schicht wird die Ausgabe des Modells nochmal auf die Anzahl an Klassen, also in unserem Fall acht, heruntergebrochen. Die Letzte Schicht ist ein Softmax-Layer. Diese Schichten, die am Ende zu dem Model hinzugefügt wurden sind in der Abbildung 4 rot markiert, diejenigen, die am Anfang des Models hinzugefügt wurden sind in der Abbildung 4 grün markiert. Um unser Model zu trainieren, verwenden wir sparse categorial crossentropy loss.

Abbildung 2: Beispiele aus den Trainingsdaten

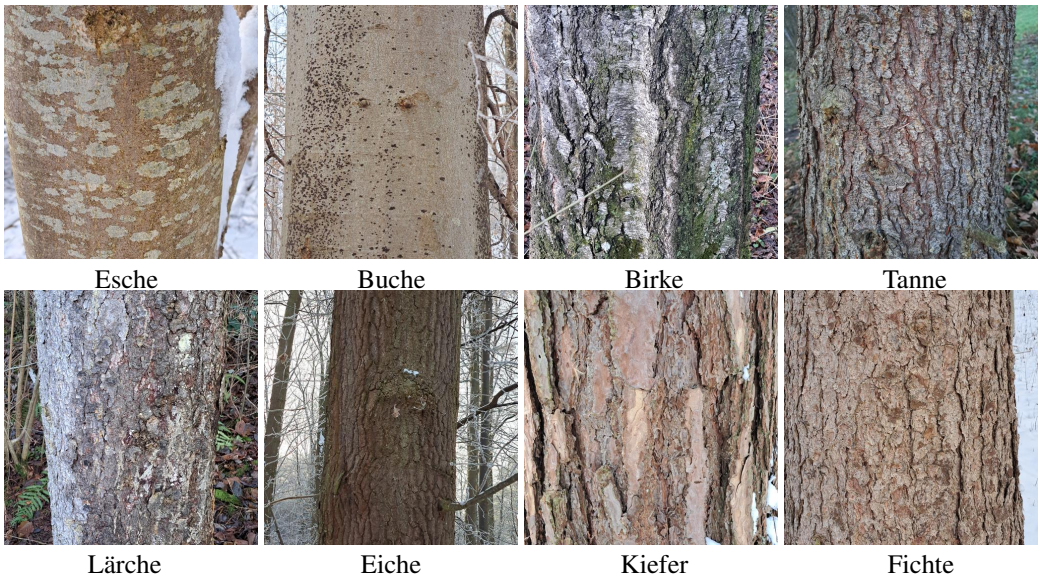

Abbildung 3: Baspiele aus dem Validationset

Die Abbildung zeigt die in diesem Paper gemachten Ergänzungen an einem VGG16 Model. Neben diesem Model wurden diese Ergänzungen noch an einem InceptionV3 model, einem MobileNetV2 model und einem VGG19 model gemacht und die Ergebnisse verglichen.

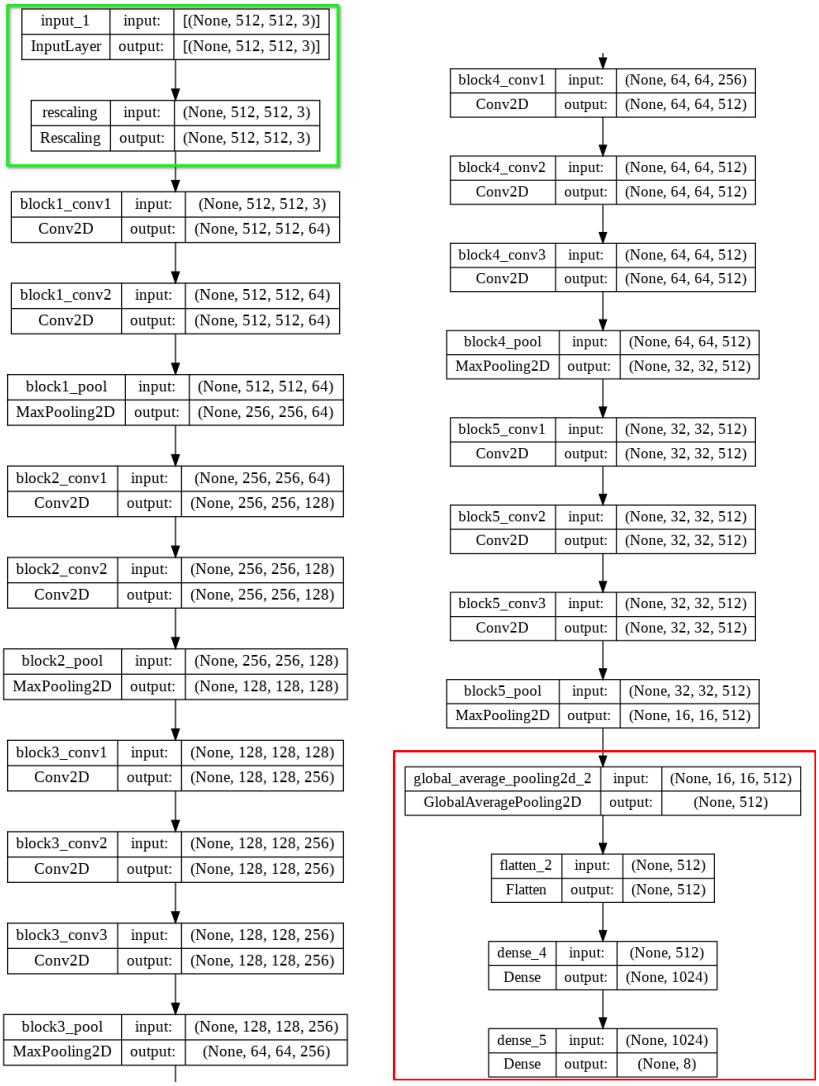

Abbildung 4: Darstellung des auf VGG16 basierenden Modells

5 Experimente/Ergebnisse/Diskussion

Wir haben verschiedene vortrainierte Basismodelle ausprobiert und deren Performance gegenübergestellt. Die betrachteten vortrainierten Modelle sind InceptionV3, MobileNetV2, VGG16 und VGG19. Die Ergebnisse der Gegenüberstellung sind in Abbildung 5 dargestellt. Die höchste Performance erzielt MobileNetV2 mit einer Trefferate von 52,85% auf dem Entwicklungsdatensatz dev-set 2 mit unseren selbst gelabelten Daten. Bei acht Klassen würde man mit zufälliger Auswahl auf eine Trefferate von 12,5% kommen, das Modell ist also deutlich besser als zufällige Auswahl. Da alle Modelle jedoch auf unseren selbst gelabelten Daten eine sehr viel geringere Trefferchance haben als beim Taining, ist es wichtig auch den Entwicklungsdatensatz dev-set 1 mit den Daten aus dem Internet zu betrachten. Auf diesen Daten weisen die Modelle eine ähnliche Trefferrate wie beim Training auf. Unsere Modelle können also nicht gut von den Trainingsdaten auf die selbst gelabelten Daten generalisieren. Ursache dafür können regional verschiedenen Unterarten sein oder auch die verschiedenen Licht- und Wetterverhältnisse, bei denen die Bilder gemacht wurden. Die selbst gelabelten Daten wurden im Winter bei Schnee fotografiert, während die meisten der Daten aus dem Internet bei guten Lichtverhältnissen und nicht mit Schnee im Hintergrund fotografiert wurden. Um ein besseres Ergebnis zu erzielen, müssten mehr regionale Daten bei verschiedenen Licht- und Wetterverhältnissen erhoben werden, die dem Trainingsset hinzugefügt werden.

Um die Performance weiter zu verbessern wurde das beste Model (MobileNetV2) ausgewählt und auf diesem ein Fine-Tuning schritt durchgeführt. Dafür werden die unteren 15 Layer des Models mit einer niedrigen Learningrate nachtrainiert. Das sorgt für eine Performanceverbesserung auf den Trainingsdaten und den Entwicklungsdaten aus dem Internet (dev-set 1), die Performance auf den selbst gelabelten Entwicklungsdaten sinkt allerdings etwas (siehe Abbildung 6).

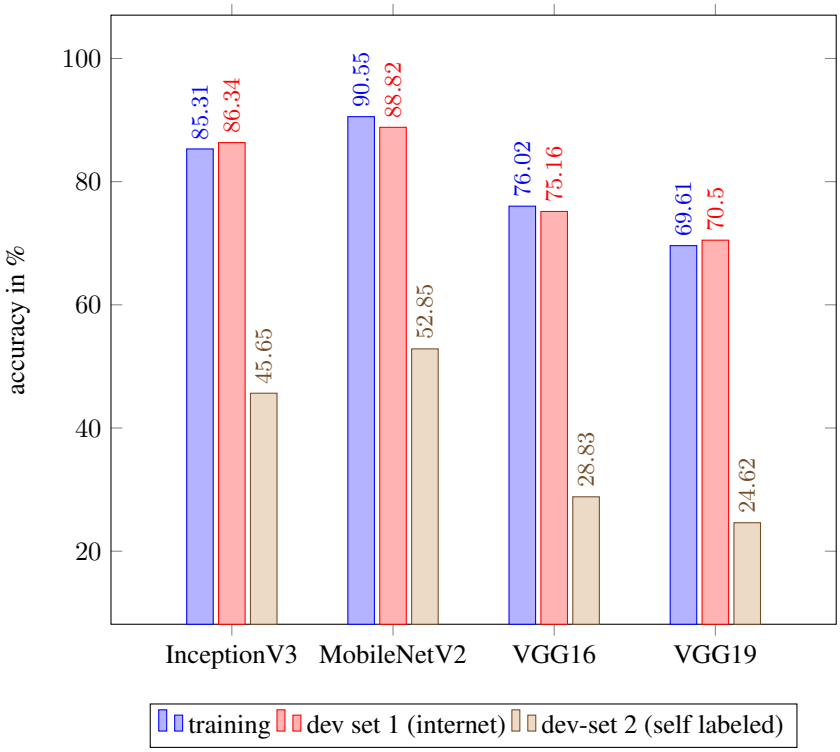

Abbildung 5: Vergleich der verschiedenen Modelle

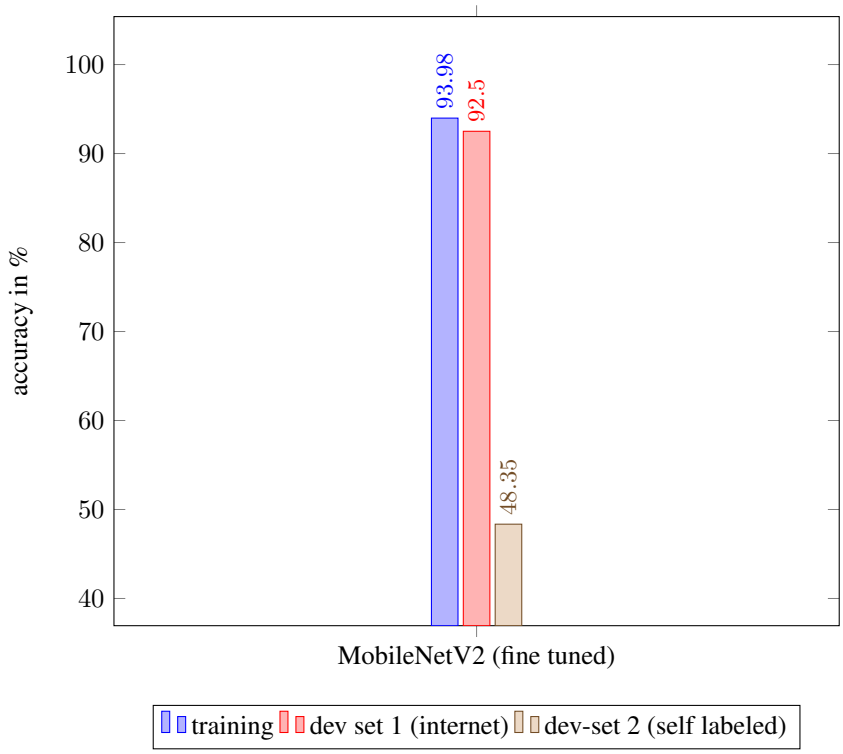

Abbildung 6: fine tuned MobileNetV2

6 Fazit und Ausblick

In dieser Arbeit wurden vier neuronale Netze mittels Transfer Learning aus vier vortrainierten Modellen erstellt. Diese wurden mit Bildern von Rinde aus dem Internet trainiert, mit dem Ziel der Klassifikation zu einer von acht Baumarten. Zusätzlich wurden händisch Daten erhoben und gelabelt, die ein Entwicklungsdatensatz (dev-set 2) bildeten. Ein zweiter Entwicklungsdatensatz (dev-set 1) wurde aus einem Teil der Daten aus dem Internet erstellt. Die vier entwickelten Modelle wurden nach ihrer Treffwahrscheinlichkeit gegenübergestellt. Außerdem wurde eine schlechte Generalisierungsfähigkeit der Modelle von den externen auf die selbst erstellten Daten festgestellt. Da die Performance auf dem Entwicklungsdatensatz aus dem Internet (dev-set 1) sehr nahe an der Performance auf den Trainingsdaten liegt, kann davon ausgegangen werden, dass bei einem Training des Modells mit regionalen Daten eine ähnlich hohe Performance erreicht werden kann.

Referenzen

Verwendete Softwarebibliotheken

- Pillow-SIMD
- google.colab (drive)
- Tensorflow
- os
- Keras
- matplotlib

[1] Stefan Fiel und Robert Sablatnig. *Tree Species Dataset consisting of Images of the Bark, Leaves or Needles*. Zenodo, Mai 2010. DOI: 10.5281/zenodo.4446955. URL: https://doi.org/10.5281/zenodo.4446955.

[2] Philippe Giguère. *BarkNet 1.0 (Part 1 of 4)*. Mendeley, 2019. DOI: 10.17632/ZGR7R2R4NT. URL: https://data.mendeley.com/datasets/zgr7r2r4nt.

[3] Zhi-Kai Huang. "Bark classification using RBPNN based on both color and texture feature". In: *International Journal of Computer Science and Network Security* 6.10 (2006), S. 100–103.

[4] Debaleena Misra, Carlos Crispim-Junior und Laure Tougne. "Patch-based CNN evaluation for bark classification". In: *European Conference on Computer Vision*. Springer. 2020, S. 197–212.

[5] Jiatao Song u. a. "Bark classification by combining grayscale and binary texture features". In: *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2004.* IEEE. 2004, S. 450–453.

[6] Kim Tae Kyung, Baek Gyu Heon und Hyun Seok Kim. *Tree bark identification dataset (BARK-KR)*. This work was supported by Korea Forest Service (Korea Forestry Promotion Institute), grant number 2018113B10-2020-BB01 and 2020185D10-2122-AA02. Zenodo, Mai 2021. DOI: 10.5281/zenodo.4872489. URL: https://doi.org/10.5281/zenodo.4872489.