Ernst-Abbe Hochschule Jena
Fachbereich Elektrotechnik und Informationstechnik

Dr. Jürgen Hess

Aufgabenblatt – Semantische Segmentierung auf Embedded-Systemen

Anwendungsanalyse, Modellbewertung und Deployment-Einschätzung

Zielsetzung

In dieser Übung analysieren Sie verschiedene neuronale Netze zur semantischen Segmentierung hinsichtlich Laufzeit, Speicherbedarf, Genauigkeit und Architekturtauglichkeit für Edge-Geräte (z.,B. Coral, Raspberry Pi). Der Fokus liegt auf der praxisnahen Modellwahl für reale Anwendungen.

Hinweis: Die zugehörigen Modelle und den Source Code finden Sie im Repository:
https://github.com/juehess/lecturematerial-semantics

Installation und Vorbereitung

Folgen Sie den Anweisungen im Repository, um die `eah_segmentation`-Umgebung auf Ihrem Raspberry Pi zu installieren. Starten Sie anschließend `jupyter-lab`:

```
$ cd lecturematerial-semantics
$ conda activate eah_segmentation
$ jupyter-lab --no-browser --ip=0.0.0.0
```

Hinweis: Notieren Sie sich die IP-Adresse des Raspberry Pi (z.,B. durch `hostname -I`). *Beispiel:* Geben Sie im Browser Ihres Rechners z.B. ein: `http://192.168.1.42:8888` und führen Sie dort das Notebook aus.

Teil A – Messung und Vergleich

A1. Führen Sie die Modelle auf dem Raspberry Pi (und ggf. Coral) aus. Dazu laden Sie das Notebook *segmentation_ example.ipynb*. Führen Sie alle Modelle mit dem Modelltyp *tflite* und dem Device *cpu* aus. Das DeepLabV3 Modell lässt sich auch mit dem Device *coral* ausführen. Erfassen Sie folgende Werte und tragen Sie diese in die Tabelle ein:

- Inferenzzeit pro Bild (ms)
- RAM-Verbrauch zur Laufzeit (MB)
- Dateigröße der TFLite-Modelle (MB)
- Subjektive Bildqualität (Skala 1–5)

Modell	**Zeit [ms]**	**RAM [MB]**	**Größe [MB]**	**Qualität (1–5)**
SegFormer B0				
DeepLabV3+ (ADE20K)				

Fortsetzung auf der nächsten Seite

Modell	Zeit [ms]	RAM [MB]	Größe [MB]	Qualität (1–5)
Mosaic				
DeepLabV3 (Cityscapes, CPU)				
DeepLabV3 (Cityscapes, coral)				

A2. Wie verhält sich die Modellgröße zur RAM-Nutzung? Entspricht das Ergebnis ihren Erwartungen? Erklären Sie mögliche Gründe für Diskrepanzen.

Teil B – Anwendungsszenarien

Wählen Sie für jedes Szenario ein Modell aus und begründen Sie Ihre Entscheidung.

B1. **XR-Brille:** Muss die Umgebung live segmentieren, um virtuelle Inhalte stabil zu verankern. Läuft auf einem Smartphone mit Edge-Beschleuniger.

B2. **Indoor-Service-Roboter:** Muss Möbel und Personen erkennen, läuft auf Jetson Nano. 1 Hz reicht.

B3. **Staubsaugroboter:** Muss Bodenarten unterscheiden, läuft auf Raspberry Pi 4 mit <50 MB RAM.

B4. Bonus: Welche Rolle spielt der Trainingsdatensatz (Cityscapes vs. ADE20K) für die Generalisierbarkeit?

Teil C – Architekturkompatibilität und Quantisierung

C1. **Netron-Analyse:** Öffnen Sie SegFormer in `https://netron.app`. Welche Layer erkennen Sie, die eine Quantisierung auf Coral verhindern? Nennen Sie zwei konkrete Beispiele.
Hinweise:

- Achten Sie auf typische Operatoren, die nicht in INT8 repräsentierbar sind oder für Coral nicht unterstützt werden.
- Beispiele: `MatMul` (Matrixmultiplikation, z. B. in Attention), `Softmax`, `LayerNorm`, `Cast`, `Exp`, `Log`, `Transpose`.

C2. Warum sind Layer wie `Softmax`, `LayerNorm` oder `MatMul` problematisch für INT8-Quantisierung auf EdgeTPU?

C3. **Bonus:** Welche TFLite-Operatoren muss ein Modell ausschließlich verwenden, um für Coral EdgeTPU geeignet zu sein?

Hinweis: Schauen Sie in die Dokumentation von Coral oder führen Sie einen Test mit dem `edgetpu_compiler` durch:
`https://coral.ai/docs/edgetpu/compiler/`
`https://colab.research.google.com/github/google-coral/tutorials/blob/master/compile_for_edgetpu.ipynb`

Viel Erfolg und viel Spaß bei der Analyse und Bewertung!