Technische Universität Dortmund

Fakultät Informatik
Design Automation for Embedded Systems Group

TODO Titel

Fachprojekt

Jack Diep, Florian Köhler, Yannick Naumann

Betreuung:
M.Sc. Mikail Yayla

27. August 2021

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

Inhalt dieses ersten Kapitels: Motivation, Hintergrund, Aufbau der Arbeit etc.

1.1 weitere Unterkapitel

⋮

Kapitel 2

Das Netzwerk

2.1 Aktivierungsfunktion

2.2 Binärer Linear-Layer

2.3 BatchNorm

2.4 Auswertung

Um am Ende das Ergebnis des Netzwerkes auszuwerten, müssen die Ergebnisse des letzten *Linear-Layers* normalisiert werden. Diese normalisierung entscheidet, ob ein Neuron feuert oder nicht. Da bei der Anwendung des MNIST-Datensatzes immer nur das Neuron ausgewählt werden sollte, da immer nur eine Zahl auf einem Bild ist, wird dieses am ende über die $argmax(x)$ Funktion. Damit ist die erkannte Zahl immer das aktivste Neuron.
Für die Normalisierung der Daten wird hier die *logsoftmax(x)* Funktion verwendet.

2.4.1 logsoftmax vs. softmax

2.5 Verluste durch binäre Linear-Layer

Durch die binarisierung der *Linear-Layer* ist zu vermuten, dass diese, im Vergleich zu normalen *Linear-Layer*, etwas schlechter performen. Dies ist der Fall, da die Anzahl der möglichen Kantengewichte stark, auf Null und Eins, eingeschränkt ist.

Durchgang	binarär	normal
1		

Trainiert wurde hier das gleiche Netzwerk, ein mal mit binären *Linear-Layern*, das andere mal mit normalen *Linear-Layer*. Jedes Netzwerk wurde für 50 Epochen trainiert, bevor die Genauigkeit ausgewertet wurde. Um sicher zu gehen, ob die Genauigkeit gegen diesen Wert konvergiert, wurde jede Messung fünf mal wiederholt.

Kapitel 3

Binarisierung

Kapitel 4

Export

Nachdem das Netzwerk nun trainiert wurde, müssen die Ergebnisse, die Kantengewichte und Schwellwerte der Neuronen, nun exportiert werden. Im Folgenden sollen diese dann in den BNN-Beschleuniger Baustein importiert und verwendet werden. Da der Import in VHDL stattfindet, eignen sich hier simple Formate, sprich eine einfache Textdatei. Diese kann dann, Zeichen nach Zeichen, von dem Import-Buffer eingelesen und in einer Matrix gespeichert werden.

4.1 Export der Kantengewichte

Da es sich bei unserem Netzwerk um ein *FullyConnected Neural Network* handelt, ist insbesondere jedes Neuron mit jedem Neuron der Folgenden Schicht verbunden. Bei unserem BNN ergibt sich also folgende Kantenanzahl

$$784 \cdot 500 + 500 \cdot 1024 + 1024 \cdot 1024 = 1.952.576$$

Diese Gewichte müssen alle, mit möglichst wenig Mehrkosten, in die Datei geschrieben werden. Da es sich bei den Gewichten lediglich um binäre Werte, Einsen und Nullen, handelt, ist kein Trennzeichen zwischen den Gewichten notwendig. Die Gewichte sind außerdem, trivialer Weise, präfixfrei und können fortlaufend in die Datei geschrieben werden.
Um die Gewichte zu extrahieren, wird zuerst über jeden *Layer* iteriert. In jedem *Layer* wird nun jedes Neuron abgelaufen. Jedes dieser Neuronen hat nun jeweils eine Kante

zu jedem Neuron in der nachfolgenden Schicht. Hier wird ebenfalls über alle Kanten iteriert und das jeweilige Gewicht wird hinten an eine Variabel an gehangen. Ist nun ein *Layer* fertig, wird der Inhalt der Variable, welche als Zwischenspeicher dient, in die Datei *weights.txt* geschrieben. So wird für jede Schicht ein IO-Zugriff gemacht.

4.2 Export der Schwellwerte

Literaturverzeichnis

[1] Beutelspacher, Albrecht: Das ist o.B.d.A. trivial!. Tipps und Tricks zur Formulierung mathematischer Gedanken. Vieweg Verlag, Braunschweig und Wiesbaden, 2004.