Bildverarbeitung Praktikum

Übung 8, Abgabe 18.06.2020

Dr. Christina Gillmann

June 10, 2020

1 Kleiner Disclaimer

Bitte beachten sie, dass es in der aktuellen Vorlesung nicht darum geht, die Algorithmen wie beispielsweise die Backprpagation selbst zu implementieren. Um das zu lernen gibt es eigene Vorlesungen, deren Inhalt den Rahmen unserer Vorlesung sprengen würden. Ich möchte sie an dieser Stelle explizit auffordern eine solche Vorlesung zu besuchen, wenn es ihr Studienplan zulässt. Neurale Netze sind eine zukunfsweisende Technologie und sie können für ihre persönliche Karriere (egal ob akademisch oder in der Industrie) von diesem Wissen massiv provitieren. Aufgrund der Popularität von neuralen Netzwerken und der guten Anwendbarkeit in der Bildverarbeitung möchte ich ihnen jedoch nicht vorenthalten wie sie schnell und einfach ein neurales Netzwerk programmiert werden kann. Ich möchte ihnen mit dieser Aufgabe ein Beispiel geben wie sie sehr gute Bibliotheken (vor allem im Bezug auf Performance und Generalität) nutzen können um Neurale Netzwerke schnell zu implementieren wowie ihnen ein Gefühl zu geben was bei der Nutzung von neuralen Netzwerken wichtig ist.

2 Die eigentliche Aufgabe: CNN Programmieren

Sie sollen ein CNN implementieren. Hierzu müssen sie tensorflow und keras herunterladen. Diese Bibliotheken stellen alle Funktionen bereit, die sie für das Trainieren und Testen von Neuralen Netzwerken benötigen.

```
conda create -n tensorflow_env tensorflow
conda activate tensorflow_env

conda install -c conda-forge keras
conda install -c conda-forge/label/broken keras
conda install -c conda-forge/label/cf201901 keras
conda install -c conda-forge/label/cf202003 keras
```

In Tensorflow gibt es einen Datensatz namens MNIST. Es handelt sich hierbei um einen sehr berühmten Datensatz, der handgeschriebene Zahlen von 0 bis 9 enthält. Dabei soll richtig erkannt werden welche Zahl geschrieben wurde.

Mit folgendem Befehl können sie den Datensatz laden. Er enthält 60000 Trainingsdatensätze und 10000 Testdatensätze.

```
import tensorflow as tf
(x_train, y_train), (x_test, y_test) =
tf.keras.datasets.mnist.load_data()
```

a) Finden sie einen Repräsentanten jeder Klasse (0-9) und zeigen sie ihn in einem Plot an.

Bevor sie das neurale Netzwerk trainieren können müssen sie die Daten normalisieren und in die richtige Form (Werte zwischen 0 und 1) bringen. Das erreichen sie durch:

```
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
input_shape = (28, 28, 1)

x_train = x_train.astype('float32')
x_test = x_test.astype('float32')

x_train /= 255
x_test /= 255
```

Nun können sie das neurale Netzwerk designen. Ein Vorschlag wäre:

```
model = Sequential()
model.add(Conv2D(28, kernel_size=(3,3), input_shape=
   input_shape))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation=tf.nn.relu))
model.add(Dropout(0.2))
model.add(Dense(10,activation=tf.nn.softmax))
```

Dieser Aufabau entscheidet über die Performance ihres Neuralen Netzwerkes. Die 10 in "Dense" muss so beibehalten werden, da sie am Ende entscheiden möchten, welche der Klassen 0-9 nun am besten getroffen wurde. (Hierzu ein kleiner Hinweis: wir werden uns in der Vorlesung noch damit beschäftigen was es bedeutet, wenn man diese Klasse nicht einwandfrei identifizieren kann).

Trainieren können sie ihr Netzwerk durch

```
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
model.fit(x=x_train,y=y_train, epochs=10)
```

```
model.evaluate(x_test, y_test)
```

Die Funktion "evaluate" wird ihnen die Genauigkeit ihres Models ausgeben.

b) Hier können sie verschiedene Parameter für die Kernel Größe wählen. Testen sie 3-5.

c) Auch das Pooling kann angepasst werden (2-5). Testen sie diese Parameter.

Jedes mal, wenn sie einen der Parameter aus b) und c) ändern, wird ihr Model eine andere Präzision bei der Vorhersage haben. Dokumentieren sie diese.

d) Es gibt im MNIST Datensatz Bilder die einfach zu klassifizieren sind und welche bei denen es schwierig ist. Ein Beispiel ist der Datensatz mit der Nummer 4444. Es handelt sich dabei um eine 9, die einen zweiten Bogen enthält. Bei allen Experimenten die sie durchführen sollten sie beobachten, ob dieser Datensatz richtig klassifiziert wird. Teilen sie mir ihre Ergebnisse abhängig von den gewählten Parametern mit. Folgend finden sie den Code, der für den Datensatz die Vorhersage trifft.

```
image_index = 4444
plt.imshow(x_test[image_index].reshape(28, 28),cmap='Greys')
pred = model.predict(x_test[image_index].reshape(1, img_rows
    , img_cols, 1))
print(pred.argmax())
```

Generell sollten sie diese Aufgabe als einen Versuchsaufbau sehen bei dem sie viele Parameter haben die sie manipulieren können. Neurale Netze sind an der Stelle eine Art Blackbox in der sie etwas reinfüttern und ein Ergebnis bekommen, jedoch können sie nicht nachvollziehen wie diese Berechnung genau funktioniert (das ist tatsächlich ein fundamentaler Forschungsfunkt). Die Qualität ihres Ergebnisses hängt von den gewählten Parametern ab und es gibt noch keine ausreichenden Methoden um dieses "Irgendwas" mathematisch zu fassen. Gerne diskutieren wir ihre Ergebnisse dann in der Abnahme (soweit ich das nachvollziehen kann, denn wie gesagt, man kann es manchmal nicht richtig nachvollziehen).