Driving Behaviour Analysis

A journey thgrough convolutional neural networks

Dinu Ion George

1. Context

Aggressive driving behavior is the leading factor of road traffic accidents. As reported by the AAA Foundation for Traffic Safety, 106,727 fatal crashes – 55.7 percent of the total – during a recent four-year period involved drivers who committed one or more aggressive driving actions. Therefore, how to predict dangerous driving behavior quickly and accurately?

2. Solution Approach

Aggressive driving includes speeding, sudden breaks and sudden left or right turns. All these events are reflected on accelerometer and gyroscope data. Therefore, knowing that almost everyone owns a smartphone nowadays which has a wide variety of sensors, we've designed a data collector application for android based on the accelerometer and gyroscope sensors.

The collected data can be used to train a machine learning algorithm to predict the given driving behaviour.

3. Dataset Analysis

The data consist of the following:

- Acceleartion (X, Y, Z axis in meters per second squared (m/s2))
- Rotation X, Y, Z axis in degrees per second (degree/s))
- Classification label (SLOW, NORMAL, AGGRESSIVE)
- Timestamp (time in seconds)

The driving behaviour can be labeld as:

- SLOW
- NORMAL
- AGGRESSIVE

The first part of the data analysis consists in analyzing the given data from the perspective of a human user.

For example, we can select 1 row from the dataset consisng of a slow, normal and aggressive:

AccX	AccY	AccZ	GyroX	GyroY	GyroZ	Class	Timestamp
0.7384782	-0.228455	0.66773224	0.069791354	-0.0299323	0.054901514	NORMAL	3581631
-2.79920	-0.6846703	-0.6576443	0.12843442	-0.10323623	0.15263996	AGGRESSIVE	3582407
-1.27812	-1.6581681	-1.579102	-0.038331	0.09896017	0.03596469	SLOW	3583306

We can see that from the given data the human mind can no longer find a pattern to label the data. From this point the problem gets a lot harder, the reason beeing that we can have no intuition if the predictions of the implemented machine learning model are trully correct or not.

Because the data is too abstract to work with, the first idea would be creating new features from the given data and eliminating the ones that are of no use.

We notice that Timestamp would be of no use in the context of a randomized dataset for testing and so it can be removed. In the context of a nonrandomized dataset, it can create a stange behaviour where the model will learn the pattern of the Timestamp and so the acceleration and gyroscope informations would be set asside. This behaviour can be demonstrated using a PCA algorithm. From the given hypothesis we can eliminate Timestamp from the training dataset.

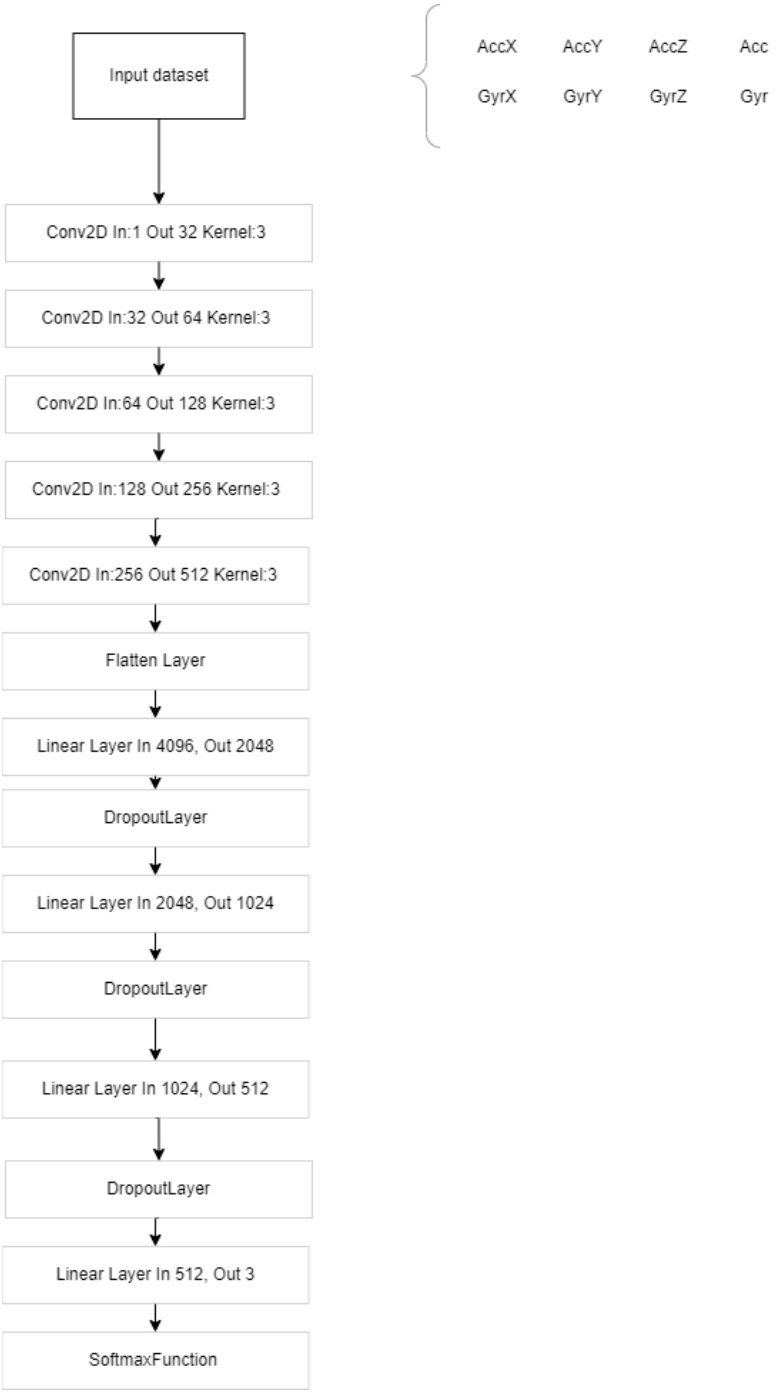
With the given Acceleration and Gyroscope axis values we can calculate the angular velocity and acceleration of the car, and so 2 more features will be created.

Acc = sqrt(ax^2 + ay^2 + az^2)

AngualrVelocity = sqrt(gx^2 + gy^2 + gz^2)

With two more features added and one eliminated we can not start creating a strategy to train a model on the given data.

4. Proposed Architecture - Convolutional Neural Networks



The input of the network would be a matrix of (2, 4).

Implementation

Used framework: Pytorch

Programming Language: Python 3.10

From the architecture above, the kernel size has been choosen as 3, and some padding rows have been added to the input matrix. The reason of that is that we want to reduce the matrix to a liniar vector, so that the flatten layer would not alter the quality of the last layer output. More complex explanations can be found in the source code.

For training the following optimizer and loss functions were choosen:

1. AdamW as optimizer, with a learning rate of 2e-5
2. CrossEntropyLoss as loss function

For each dropout layer, was choosen a dropout rate of 0.15

For each Conv2D layer, was choosen a kernal_size of 3, stride 1 and padding 1.

Experiments

After multiple experiments it was observed that with a batch size of 32 the model generalized better.

The number of epoches in wich the model was trained is equal to 5. Even tho after multiple experiments it was observed that the accuracy of the model improved after 7 or 8 epoches. Having a small dataset can produce the phenomenon of overfitting.

The proposed arhitecture achived a score of

Analisyng the predictions of the model, we can observe that no NORMAL behaviour was predicted, but a very good proportion of data referencing the SLOW and AGGRESSIVE behaviour has been predicted right.

Intuition: creating a dataset formed by SLOW and AGGRESSIVE behaviours can achive a very good accuaracy score on predicitons.

Points of improvement:

1. Adding multiple layers can improve the generality of the model.
2. Using a kernel of different dimensions
3. Modifying the dropout rate of the layers
4. Concatenating the output of the layers
5. Modifying the learning rate of the optimizer
6. Creating new features may improve the accuracy of the model.

5. Deliverables

- DrivingBehaviourModel.pth – Trained model
- DrivingBehaviour.ipynb – source code of the proposed architecture
- DrivingBehaviour - Kernel2.ipynb – source code of a similar architecture using a kernel of size 2
- submision.csv – predicted driving behaviour of the model
- Documentation