Deep Learning for Human Activity Recognition
Extra Credit Assignment, CMSC 422

Due Dec 14, 2018

1 Overview

This is an open-ended, experimental question presented for those deeply interested in the subject matter of the class. It is purely optional, and you will be exploring on your own. If you take this on, be aware there is no "right answer". The objective of the assignment is to explore training a Convolutional Neural Network (CNN) to perform automatic recognition of physical activities, from on-body inertial sensor data. This is a mini-project for which you'll need to create your dataset, label it (or not, depending on your technique), identify appropriate network architecture, train it, do validation and document your results on an appropriate test dataset.

- Wikipedia: `https://en.wikipedia.org/wiki/Activity_recognition`

2 PyTorch

We recommend implementing and training your network with PyTorch, an open-source machine learning library for Python. You have gained experience with Python in the course, and PyTorch also tends to be easier to pick up than some of the other libraries, so we think a final project in PyTorch is appropriate.

- Documentation: `https://pytorch.org/docs/stable/index.html`

  (You may find `torch.nn` most useful)

3 Sensor data

You will need to acquire or generate a dataset. Consider the following:

- Generating smartphone data: Tools such as "gauges" in the Apple App store seem to be suitable but there may be many other options. If you have aptitude in mobile development, you could certainly write your own data generator as well. This might involve some movement on your part to generate data. You may need to keep a log or other documentation to properly label the states and transitions in the sensor data. You may work in teams to produce data sets.
- Existing datasets: You may also consider looking at some of these datasets: `https://en.wikipedia.org/wiki/List_of_datasets_for_machine_learning_research#Motion-tracking`

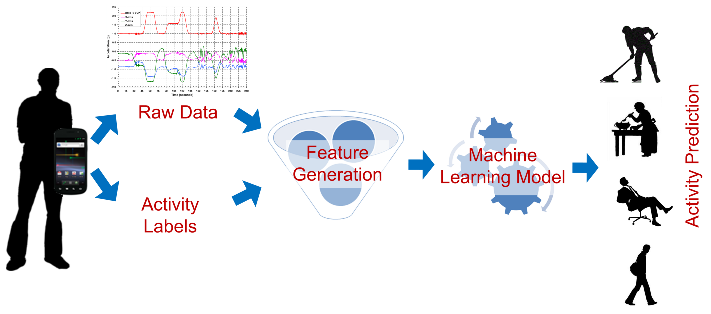

Figure 1: Illustration of activity recognition from smartphone sensor data

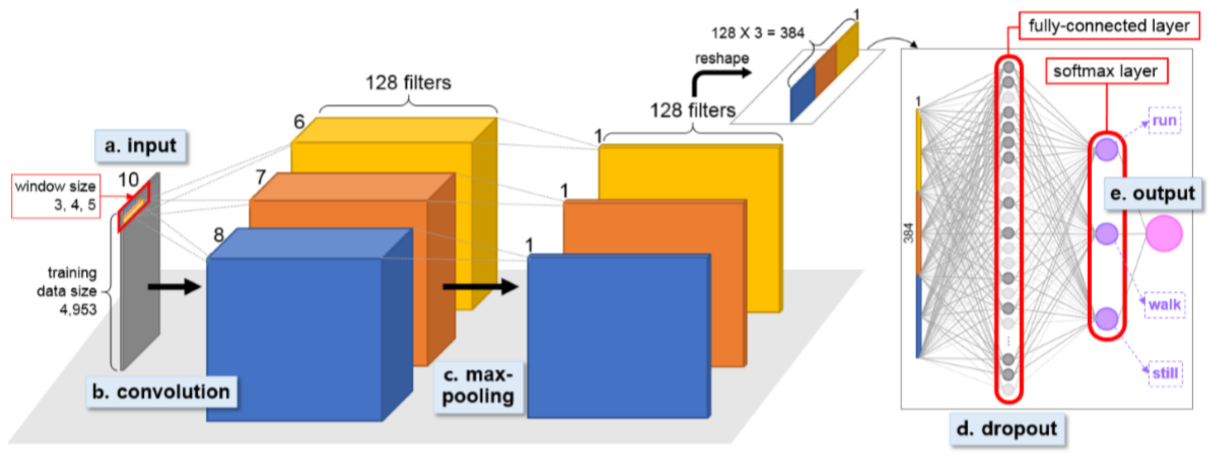

Figure 2: An example of a CNN, with 2-dimensional input, convolutional, max-pooling, dropout, fully-connected, and softmax layers.

3.1 Feature selection

For the purposes of this assignment, you might try to work with various subsets of the features. For example, if you have velocity, it should be pretty easy to tell when you are driving (simply based on the speed) in a manner that has low probability of false positives; but could you tell from just the accelerometer data? There are also quirks of smartphone sensors that you may want to think about (e.g., velocity may be computed from GPS or trilaterization of the cellular signals, which may diminish indoors, etc). Try to find the fewest sensors that provide best results.

3.2 Preprocessing

Although neural networks can learn their own representations, preprocessing can help training the network. Consider taking some steps to prepare your data.

- Normalization and mean subtraction
- Imputation

Scikit-learn documentation: `https://scikit-learn.org/stable/modules/preprocessing.html`

3.3 Activity states

What are the activity states and their labels? You might consider a discrete states (sitting/motionless, walking, running, stair climbing, driving); or start with something simple (moving and not moving). If we are aiming to duplicate “fitbit”-type activity, we probably want to distinguish between moving and not moving; and, if moving, between walk, jog, run and climb stairs. Consider using one-hot vector encodings of each class for your output labels.

- One-hot vector encoding: `https://en.wikipedia.org/wiki/One-hot`

4 Convolutional neural network

The rest of this document gives tips on implementing and training your CNN in PyTorch.

- Tutorial: `https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html`

4.1 Softmax layer

Assuming a one-hot vector encoding of outputs, each node in our output layer represents the probability of the corresponding class. We want our outputs to resemble a valid probability distribution, where the probabilities are all non-negative and sum to 1. Adding a softmax layer at the end of your network achieves this. See Figure 2 for an illustration. In PyTorch, you will use `torch.nn.Softmax`.

- Description:
  `https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax`
- Softmax function: `https://en.wikipedia.org/wiki/Softmax_function`

4.2 Parameter sweep

Neural networks have several hyperparameters, and the convergence rate is dependent on the choice of these parameters. You will need to identify the appropriate parameters for training on your particular dataset. One naive approach is performing an exhaustive grid search (or parameter sweep) through the parameter space. For example, you may perform a parameter sweep over all combinations of step size and batch size. You can use this approach, but it may take a while if your dataset is large. You may be able to identify the appropriate parameters on a smaller subset of your data, if your full dataset is too large.

- Wikipedia: `https://en.wikipedia.org/wiki/Hyperparameter_optimization#Grid_search`

4.3 Minimizing overfitting

You may find that your network overfits your training data. Here are a few things to consider:

- Make sure you have at least 2 training samples for every parameter in your network. You can compute the number of parameters in your network with

  ```
  sum(p.numel() for p in cnn.parameters() if p.requires_grad)
  ```

- Add a dropout layer after the convolutional layers (see Figure 2). Adding a dropout layer before or between convolutional layers is usually unnecessary.

  `https://pytorch.org/docs/stable/nn.html#dropout-layers`

5 Experimental evaluation and report

Document the results, provide commentary, and outline lessons learned. This is an open ended assignment– you may need to identify additional resources, citations or literature beyond those presented in class. You will also need to decide how to best present your experiment and results. A successful project should stand as an example of independent work that you could present as part of a portfolio to a prospective employer or as part of an application to graduate school.

There is no exact page requirement, and your report will be graded on quality. We don't expect anything more than 4 pages. Turn in your report on Gradescope by December 14.

6 Disclaimer

This is an open-ended, experimental question presented for those deeply interested in the subject matter of the class. It is purely optional, and you will be exploring on your own. If you take this on, be aware there is no "right answer".