University of Waterloo

ECE 657A:

Data and Knowledge Modeling and Analysis

Spring 2021

Assignment 3:

Classification with Deep Learning on Covid-19 and FashionMNIST Datasets

Due: August 11, 2021 at 11:59pm EST

1 Overview

Broad objectives:

- Set up two types of Fully Connected Deep Neural Networks (one standard, one your own choice) on the same COVID-19 dataset from assignment 2. Results posted to Kaggle.
- Set up and train a small Convolutional Neural Network in python tensorflow for classification on the FashionMNIST dataset. No Kaggle for this.
- Shorter, concise report than previous assignments. The focus on design of your model and results, rather than data preprocessing or feature engineering.

Collaboration:

- You can do your work alone or in pairs.
- You can collaborate on the right tools to use and setting up your programming environment, but each team must produce their own code, report and kaggle submission.
- If you are working in a pair you will need to join a "group" on Crowdmark. Even if you are working alone, you need to sign up for a group on LEARN to get access to a dropbox for your code.

Hand in:

- One report per team, via the CROWDMARK site in PDF format. You will need to divide the PDF up into multiple files and drag and drop each onto the relevant `[CM#]` questions. You should receive an invite to crowdmark by email.
- Your report should be as concise as possible, show the main results and analysis without showing repeated versions of the same output. You should include small pieces of code in your report to demonstrate the core aspects of what you did.

- You will also submit the code/scripts needed to reproduce your work as a python jupyter notebook to the LEARN dropbox.

Tools: You can use any libraries available in python, tensorflow, keras, scikitlearn for this project. You need to mention explicitly which libraries you are using, any blogs or papers you used to figure out how to carry out your calculations.

2 Dataset 1 : DKMA-Covid19-Jan-States *Again!*

NOTE: *You already know this dataset well from Assignment 2, the data and the task are exactly the same, the only difference now is that you will use Deep Neural Networks to build your models. This time around you also don't need to spend so much time explaining data pre-processing, just summarize what you did and get to the DNN models you tried and their results.*

This dataset uses statistics on COVID-19 cases in US states in January of 2021. The original data comes from the following two sources:

- John Hopkins University CSSE COVID-19: `https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data`
- US 2020 Census

The datapoints describe various information about the population and Covid pandemic situation in **50 US States** and over each day of the month of **January 2021**. Each record gives Covid information for that *day* and demographic data for the *whole state* as well. Note the demographic data is duplicated for each state since for all days since it doesn't change that quickly and is based on data from the last census. Also note that the data was combined together from multiple sources and not processed for uniform number formats and was not normalized. As far as we are aware there is no missing data, but there are certainly some states which more extreme values than others, so consideration of outliers is reasonable. All of this will be initial pre-processing you will need to carry out.

File Descriptions:

- `dkmacovid_train.csv` - main training data with input features and three binary labels.
- `dkmacovid_kaggletest_features.csv` - feature for a test set, with "Id" column for Kaggle competition.
- `dkmacovid_kaggletest_labels.csv` - labels for the kaggle test set...oh, you don't get that one :)

Fields Descriptions:

- **Day**: Date in January 2021 ranging from Jan 2 to Jan 31.
- **State ID**: Arbitrary ID number for each state, based on alphabetical order. Note there are 51 states since the District of Columbia is also included.
- **State**: Name of the US State.
- **Lat**: Latitude for the geographic centre of the state.
- **Long_**: Longitude for the geographic centre of the state.
- **Active**: Number of active, tracked COVID-19 cases that day in that state.
- **Incident_Rate**: see original data source
- **Total_Test_Results**: see original data source
- **Case_Fatality_Ratio**: see original data source
- **Testing_Rate**: see original data source
- **Resident Population 2021 Census**: see original data source
- **Population Density 2021 Census**: see original data source
- **Density Rank 2021 Census**: see original data source
- **SexRatio**: see original data source

Label Description:

Each row of this data dataset represents a particular **day** in January, 2021 in a particular **US State** and has three binary labels which are:

- `True` if is the corresponding count went **up** from the previous day.
- `False` if is the corresponding count went **down** from the previous day.

The three binary labels are:

- **Confirmed**: Was there a daily increase in the **confirmed total cases** of COVID-19 in that state on that day?
- **Deaths**: Was there a daily increase in the **number of deaths** from COVID-19 in that state on that day?
- **Recovered**: Was there a daily increase in the number **of people recovered** from COVID-19 in that state on that day?

Note that the labels are fairly **unbalanced**, so there are quite a few more `True` cases than there are `False` cases. **Recovered** is the most balanced, so it should be easier to train. **Deaths** is next and **Confirmed** is the least balanced.

2.1 Data Pre-processing and Preparation

`[CM1]` **(10 points)** (1 page maximum) Report a very brief summary of the pre-processing steps you applied to the data to make it usable for analysis. Since you already did this for Asg 2, you do not need to repeat those details. You should just describe at a high level what manipulations you made to the data and any additional things you did differentlyl from asg2. This could inclue features which were added from other datasources, Representation Learning methods, etc.

2.2 Deep Neural Networks

In your implementations for deep learning, please use *Tensorflow* or *Pytorch*. For the data format, since we have three seperate labels for each datapoint you have two ways you use them:

- you can use the data as is, with three networks making binary classification decisions for each datapoint
- Or you can transform the data so that each datapoint has one of eight labels, one for each combination of `True/False` triples.

Whichever you choose, make it clear in the report and in your network design.

Default Network

`[CM2]` **(15 points)** Classify the data using a Fully Connected Deep Neural Network with the following setup.

- Two hidden layers with 20 hidden units each
- ReLU activation functions
- Softmax output layer for the classification decision

Your Own Network

`[CM3]` **(15 points)** Classify the data using any other DNN architecture that you want to try. **Describe the design of your network using text and figures.** This includes details necessary to reproduce it such as network architecture, optimizers, activation functions, regularization methods, design choices, numbers of parameters.

You can also consider exploring questions about what type of network architecture would work well on this data. (eg. can the use of *Resnet* learn a better classifier than a simple fully-connected network? or Does a combination of fully-connected and *RNN/LSTM* perform better?) Be sure to **cite any sources** you used to research your approach: libraries, as well as papers or blogs.

2.2.1 Results Analysis

`[CM4]` **(10 points)** Briefly report on the following:

- Runtime performance for training and testing on both networks for all three labels (either as three seperate models or combined).
- Comparison of the different algorithms and parameters you tried.
- You can use any plots to explain the performance of your approach. But at the very least **produce two plots**, one of **training loss vs. training epoch** and one of **classification accuracy vs. training epoch** on both your training and test set.

2.3 Kaggle

The **Kaggle Competition** for this assignment can be found here:

`https://www.kaggle.com/c/ece657as21-asg3`

`[CM5]` **(10 points)** In this part of your report on Crowdmark simply enter your group name and url on kaggle, so we can find your results.

The kaggle setup for asg3 is exactly the same as asg2 so that your results are comparable. You can use any method from the assignment to produce your answer submission. The description on Kaggle will tell you which label(s) you need to predict and the format. Data from five states *Washington, Tennessee, Iowa, Texas, Illinois* has been held out from the `train` set and will be used for testing your trained algorithms on Kaggle and is available on LEARN (and Kaggle) as `dkmacovidkaggletestfeatures.csv`.

Assessment

Similarly to assignment 2, your score on the final public and private datasets on Kaggle will be used to calculate a grade for a portion of the assignment. The target scores will be announced later but any submitted model that runs and beats the provided baseline solution will already receive half of the score.

3 Data Set 2 - Fashion MNIST ... with a Twist

An image dataset based on ”Fashion MNIST”. The input features will be the same but the **label** you will be using will be new, something we have computed and created based on the data. Note: there is *no Kaggle* competition for this dataset.

This is an image dataset based on publicly available dataset Fashion MNIST:

`https://github.com/zalandoresearch/fashion-mnist`

It is composed of small (28x28 pixels), grey-scale images of clothing items such as shoes, coats, pants, etc. The dataset was create to serve as a direct drop-in replacement for the well known MNIST dataset (`http://yann.lecun.com/exdb/mnist/`) dataset for benchmarking machine learning algorithms. It shares the same image size and structure. The 'target' field was originally assigned a label for clothing type to each image as an integer from 0-9. The training set contains 60,000 examples and the test set 10,000 examples.

For this assignment, we are providing you the same input features, but the **labels** you will be using will be **new**. We have created a new label based on the data and calculated to be associated with each entry. The target is the modified label for the images obtained based on a formulation to categorize the fashion MNIST images to a new set of categories. The new categories are numbered 1-5, however, the detail regarding the categories is held out on purpose.

The training dataset that you download from LEARN under Assignment 3 will have the "FashionMNIST with a Twist" dataset using these new labels.

The classification label is the feature `target`.

File Descriptions:

- `trainx.csv` - the training set for your model. The training file contains vectors of size 785 representing pixel vaues of a 28x28 image.
- `trainy.csv` - is the "target", a numeric target obtained using our formulation.
- `testx.csv` - are the features for the test set to use for final evaluation.
- `testy.csv` - the target feature for the text data.

3.1 Classification with Convolutional Neural Networks

`[CM6]` **(20 points)** Explanation of your model (algorithms, network architecture, optimizers, regularization, design choices, numbers of parameters)

- You can use any CNN-based approach to solve the classification problem. You can explore any architecture of the network you like.
- You should also consider exploring other ML methods and Deep Neural Network variants to solve the problem, cite any sources you used : library, as well as papers or blogs.
- Whatever you choose, provide a concise and clear description of your architecture sufficient to implement it, some justification for your choices and reference any materials you used along the way.
- Show some of the important code blocks to implement your model. We will also consult your full code on LEARN, so this is your chance to guide us to understand your code and how you achieved your result.

3.2 Results Analysis

`[CM7]` **(20 points)** Briefly report on the following:

- Runtime performance for training and testing.
- Comparison of the different parameters or designs you tried.
- You can use any plots to explain the performance of your approach. But at the very least **produce two plots**, one of **training loss vs. training epoch** and one of **classification accuracy vs. training epoch** on both your training and test set.