CS771 - Intro to ML (Autumn 2024): Mini-project 1

Due Date: October 17, 2024 (11:59pm)

Important: Before starting to work on this mini-project, all members of your group should read this document carefully for the instructions and to understand the datasets provided, and the deliverables we need from you for this mini-project.

1 Introduction

In this mini-project, you are provided 3 binary classification datasets. Each of these datasets represents the same machine learning task (which can be posed as a binary classification problem) and was generated from the *same* raw dataset (we will reveal the raw dataset and the task once the submission deadline is over). The 3 datasets (described in Section 2) only differ in terms of features being used to represent each input from the original raw dataset. Each of these 3 datasets further consists of a training set, a validation set, and a test set. The validation set can be used for selecting the best hyperparameters or any other analyses you may want to perform (some of which is mentioned in this document). While the validation set labels are provided to you, the test set labels are hidden (you will submit the test set predictions and we will compare them with the ground truth labels that we have hidden from you).

For this mini-project, your group has to work on two tasks as described below.

1.1 Task 1

The first task is to train some binary classification models (of your choice that you think would be appropriate) on each of the 3 datasets and pick the the best possible binary classification model for each of the 3 datasets. Our notion of "best" will be based on two aspects: (1) high accuracy of the trained model on validation/test set, and (2) low amount of training data used to train the model. The latter aspect is important since a model that requires a low amount of training data helps reduce the data-collection/training cost, while still generalizing well on the unseen test dataset. To assess this, please experiment with varying the number of training examples where you use the first {20%,40%,60%,80%,100%} training examples from the provided training set, and report how the trained model's accuracy varies on the *validation* set (note that you are also provided the labels of the validation set, so you can compute the validation set accuracy yourself) by showing a plot (X axis is the % of training examples and Y axis is the validation set accuracy). Include these plots in your final PDF report. The role of this analysis is to identify the best ML algorithm/model that

needs a small amount of training data and also generalizes well on the unseen (validation) set. Once you have identified your best models for each of the 3 datasets, use them to predict the labels for the corresponding test sets. You need to submit these predictions in 3 separate text files (see the "Deliverables" section at the end of this document).

1.2 Task 2

Once you have completed task 1, you will then investigate if using all the 3 training datasets together (recall that these 3 training datasets contain the same inputs but with different feature representations), you can learn an even better model that has a better accuracy than the models learned on each of the 3 datasets individually. Note that even though these 3 datasets are extracted from the same raw data, the feature representations used are different in each of them, so there may be an improvement in combining them (how you combine them is up to you and is left to your ML knowledge/creativity). For this task too, please experiment with varying the number of training examples where you use the first {20%,40%,60%,80%,100%} training examples, and report how the trained model's accuracy varies on the validation set (to identify the best model) and include the accuracy vs training set size plots in the final PDF report, and submit the predictions (in a text file) on the test set using your best model for this combined dataset similar to how you did for task 1 (see the "Deliverables" section at the end of this document).

2 Dataset Description

Note that each of the 3 datasets has the same number of training examples (recall that they represent the same raw training data but only different in the features used to represent each input). The 3 datasets you are given include:

- **Emoticons as Features Dataset**: The features of each input are given in form of 13 emoticons (see an example below). Note that since each emoticon is associated with a unicode value, you can also think of each input as consisting of 13 categorical (i.e., discrete-valued) features.
  - Format: CSV file with 13 columns containing `input_emoticons` and last column containing the binary `label`
  - Example: `input_emoticons: "`😟😪🙂🤔😷😌😠🤡😏👍🚴 `...", label: "0"`
  - `length of input_emoticon (number of features) = 13`
  - `label` $\in \{0, 1\}$
- **Deep Features Dataset**: The features of each input are extracted using a simple deep neural network. This resulted in each input being represented using a 13×786 matrix which basically consists of 786-dimensional embeddings of each of the 13 emoticon features (recall the description of the Emoticon dataset) of an input. You may think of ways to convert this matrix into a more suitable feature representation of that

input. One naïve way would be to average the 13 embeddings into a single embedding of size 786 which can represent the input, but you are free to think of better ways than just using the averaging. You are even allowed to apply other transformations on these features as well (although we already used a deep neural network based feature extractor to construct this dataset, you can even use another deep neural network to extract features from these features!)

  - Format: `.npz` file containing columns `features` (a list of floats) and `label`
  - Example: `features: [[0.1, -0.3, 0.5, ...], ...] label: "0"`
  - `input_features` $\in \mathbb{R}^{13\times 786}$
  - `label` $\in \{0, 1\}$

- **Text Sequence Dataset**: The features of each input are given in form of a length 50 string consisting of 50 digits (see an example below). Again, feel free to try out other transformations of these features if you think they would help. This dataset is perhaps the most challenging one among the 3 but, with regard to this dataset, please do keep in mind that the length 50 string-based feature representation of each input corresponds to emoticon and deep features based representations of the same input in the other two datasets. However, when learning the model for this dataset, you are only allowed to use this dataset (only feature transformations are allowed).

  - Format: CSV file with columns `input_str` and `label`
  - Example: `input_str: "02418121412265492664 61...", label_str: "0"`
  - `length of input_str = 50`
  - `label` $\in \{0, 1\}$

3 Some Experimental Protocols

ML models that you are allowed to use: For this mini-project, feel free to use existing implementations of ML models available in libraries such as scikit-learn or other publicly available code for the models you want to try, but please do mention the sources. You may use any ML model of your choice, and you are also welcome to use your own implementations of ML models. However, regardless of the methods you use (e.g., whether you use an off-the-shelf ML model or your own implementation, or a pre-defined or learned transformation of the given features), the only constraint is in terms of the model size, i.e., the number of *trainable* parameters of the model should not exceed 10,000. Here, the number of trainable parameters could refer, for example, to the total number of weights in a linear model ($\mathbf{w} \in \mathbb{R}^D$ means there are D trainable parameters), or the total number of trainable weights in the input, hidden, and output layers of a deep neural network. We are imposing this constraint to prevent usage of very complex deep learning architectures, and let you focus on simpler models instead, while still giving you some flexibility in terms of using a variety of options for the ML models including some simple deep learning models. **Important:** Note that it is not necessary that you use deep learning models for this mini-project. We will evaluate

you group's output based on NOT how fancy your models are, or whether your models are giving very high test accuracies, but in terms of the group's effort demonstrated even if you have experimented with rather simple models, but have provided a good analysis of whatever you did (but that being said, we do expect that your models would give respectable enough accuracies :-))

Feature extraction/engineering on the given features: You may choose to work with the given features for each dataset or can try some feature transformation if your model works better on another transformed feature representation (e.g., using a kernel, or using a deep neural net based feature extractor, or anything else that you think might work for the problem). If you use some pre-trained deep learning model as a feature extractor, we won't count the number of parameters of that pre-trained model towards the 10,000 parameters limit. The limit of 10,000 that we have specified is only for the trainable/learnable parameters of whatever model you are using. If you are using any pre-trained deep learning model as feature extractor, you must specify what that model is in your final report.

4 Guidelines and Deliverables

Task Summary/Guidelines

Recall that your first task is to develop a machine learning model (a binary classifier) for each dataset individually (so you will develop 3 models, one for each dataset) and report the predictions (binary labels) on a separate test dataset that we have provided (the test labels are not provided to you; we will compare your models' predictions with the ground truth labels in terms of the accuracy metric). Since you are working in groups, you may want to split this task such that 1-2 students from each group are working on each dataset (but you all may benefit from discussing among your group members while working on each dataset individually).

Recall that your second task is to try combining these 3 datasets to train a single model and see whether this model outperforms the models trained on the 3 datasets individually. How you combine the datasets is up to you (you can think of combining the features, doing some feature transformation, or using an ensemble, or any other approach that you think is suitable), and report the predictions on the same test dataset.

Note that, in addition to training and test datasets, you are also provided a validation set which you may use to tune the hyperparameters of the models, or any other analysis (like seeing how the performance of the model varies as we vary the amount of training data).

In summary, some of the guidelines/directions that you should follow:

1. Analyzing each of the 3 datasets in detail to see how to make the best use of the given features (either using them as it is, or by using some transformation) for that dataset to get the best performance.

2. Developing the best possible classification model for each of the 3 training datasets. As described earlier, when determining the "best", please keep in mind both amount of training data as well as accuracy.

3. Comparing/consolidating the findings from each dataset. Here, you should discuss among your team members your findings and insights obtained from working on individual datasets.

4. Assessing if you can find any correlations or patterns across the feature representations in each of the 3 datasets that you could possibly leverage to learn a better model by combining the 3 datasets.

Datasets

You can download the datasets and some helper code from the following URL:

`https://tinyurl.com/cs771-autumn24-mp-1-data`

The file `mini-project-1.zip` available at this link contains following files

- `train_text_seq.csv, train_emoticon.csv, train_feature.npy`: These contain both the training inputs and their corresponding true labels.
- `valid_text_seq.csv, valid_emoticon.csv, valid_feature.npy`: These contain both the inputs and their corresponding true labels. These files can be used to tune hyperparameters or test yours models, but should not be used as part of the training set.
- `test_text_seq.csv, test_emoticon.csv, test_feature.npy`. These contain only the test inputs on which the final predictions will be made using the trained models.
- `read_data.py` containing initial code to read above files and `your_group_no.py` dummy Python code.

Deliverables

Your group should submit the following in a single zip file named project-group-number.zip (we will assign a project number to each group).

- Detailed report (in PDF, maximum 6-8 pages in a LaTeX template available at `https://tinyurl.com/cs771mp-rep`) about your findings (e.g., ML methods tried and the rationale behind choosing the methods you tried) of training with these 3 datasets, and also your approach of learning the unified, improved model that combines the 3 datasets, and whether you see any improvements upon combining (it is okay if your best model is based on training on a single dataset and is better than the one that combines all 3 datasets; you may say so in the report if combining didn't yield any benefits). In your report, please clearly specify which model performed the best for each of the 3 datasets, and also for the combined dataset.

- Your code (Python files only). Your code must include README file or detailed comments. You may have multiple Python files but must have a file named `<your group no>.py`. After running this file, it should produce 4 predictions `.txt` files with each line containing the predicted label (0/1) of one test input. These four text files should be named `pred_emoticon.txt, pred_deepfeat.txt, pred_textseq.txt` and `pred_combined.txt`, respectively, and correspond to the *test set* predictions of the models trained on the 3 datasets individually, and of the model trained by combining the three datasets. We will compare your predictions with the hidden ground truth labels of the test data, and compute your models' test accuracies.

Each group should make only one submission (single zip file) on the following link: `https://www.dropbox.com/request/ywjwxKd9vRv8w6Y6GXch`