Deep Learning Shared Task

For this assignment you will participate in a shared task. The goal is to assign speaker labels to spoken utterances.

You will need to submit your predictions, as well as a report describing your solutions and the code which you used to generate the predictions.

This assignment is worth 30% of your course grade. The assignment grade will be based on the quality of your work as judged by the instructor based on your report and code.

The quality of the report and the code will be evaluated according to following criteria:

- does the submission follow the specified requirements
- is the solution appropriate for the problem and does it feature any especially creative approaches
- are the experiments and results explained clearly and does the report match the code

Additionally, you will get a bonus based on your ranking on the leaderboard of the shared task.

Specifically:

- if your rank first, you will receive bonus 2 points;
- if your score is no better than the provided baseline your will receive no bonus;
- for intermediate ranks the bonus points will be linearly interpolated.

The performance of the baseline solution is shown on Codalab (submitted by account `gchrupal`.)

Report

Your report should be 2 page maximum, in PDF format and should include the following:

Page 1: Description of your computational learning experiments, including:

- Deep learning architecture(s) used,
- Training, hyperparameters and optimization
- Baselines (if applicable)
- Discussion of the performance of your solution.

Page 2:

- The name of the account under which you submitted your results to the competition on Codalab (see below)
- Detailed specification of the work done by group members
- References or appendices (if applicable)

Code

Your code should be a plain Python script (.py, not a notebook) which can be run to generate your predictions. You do not need to include the training data. You may divide you code into several modules if necessary, but make it clear which file needs to be run to reproduce your predictions.

Codalab submission

You will need to submit your the file with your predicted labels to the competition server. The team leader will need to get a codalab account (using a tiburguniversity.edu email), and will be responsible for submitting your solution. Indicate the name of this account in your report. For further details about the format of the prediction file, see section Submission to Codalab.

IN SUMMARY: submission consists of the following items: 1. Report (.pdf, Canvas) 2. Code (.py, Canvas) 3. Prediction file (.zip, Codalab)

Group work

Your report needs to contain a detailed description of who did what, so make sure to keep track of this information.

Note: it is not acceptable to just say *All members worked together and contributed equally.*

If there are any problems with collaboration, such as serious disagreements, a group member not contributing, or a group dissolving, make sure inform the course coordinator as soon as possible via email.

Code reuse rules

Remember this assignment is group work. You are not allowed to collaborate with students from other groups. If you want to discuss or share anything related to the task, only do this publicly via the Canvas discussion board.

Submissions will be checked for plagiarism.

If you are found breaking the above rules you will be reported to the Board of Examiners for fraud.

You are allowed to use:

- code examples provided by the instructor during the course, or as part of the competition;
- open source libraries available for Python;
- code found on Github, Stackoverflow or similar websites, as long as it is credited in your script with a link to the source.

Dataset

Spoken utterances labeled with speaker labels

Our task is to assign a speaker ID label to spoken utterances. Each utterances is provided as a separate audio file in the WAV format. For the training data, you are provided with the speaker ID labels for each of these files. For the test data, you only have the files, and need to generate the labels.

The data is available on canvas and organized as follows:

- `data.zip`: compressed folder with training files in the `train` subfolder and test data in the `test` subfolder.
- `train.json`: the mapping between file names and speaker ID labels
- `test.json`: test items which you need to provide the labels for.

You can read the files `train.json` and `test.json` using by importing the `json` library and using the function `json.load`. After loading you will have a Python dictionary.

Submission format

In order to prepare your the file with your predicted labels, you need to replaces the null values in the file `test.json` with the ID labels. Important:

- The predicted labels need to be strings, not integers.
- The names of all the input audio files need to be included in the submission file, otherwise your submission will fail.

The following illustrates the first few lines of an example submission file:


```
{
  "2976155358_b4dd4407cf_4.wav": "123",
  "3502459991_fdec2da131_3.wav": "234",
  "539493431_744eb1abaa_4.wav": "123",
  "2208055895_37cd8e1edf_4.wav": "567",
```


Evaluation metric

The evaluation metric for this task is mean error rate, or the percentage of incorrect labels. The following function will be used for evaluation:

```
import numpy as np
def evaluate(gold_path, pred_path):
    gold = json.load(open(gold_path))
    pred = json.load(open(pred_path))
    err = np.array([gold[key] != pred[key] for key in gold.keys()]).mean()
    return err
```

Method

There are three important restrictions on the method used:

- The method should be implemented as neural network model using the Pytorch library. You may use other Python libraries in addition to Pytorch, but the core model should be coded in Pytorch.
- It should be fully automatic, that is, by re-running your code it should be possible to re-create your submission. No manual processing is allowed.
- Every software component used should be open-source and possible to install locally. This means that you cannot, for example, access a web service to carry out any data processing.

Some hints:

- Use part of the provided training data as a validation set.
- Only submit to Codalab after validating your results on this validation data.
- Start development with a simple baseline.

Submission to Codalab

The competition is hosted on Codalab at the following URL: https://bit.ly/3sCIsgN

You can submit your results in the **Participate** link.

Over the course of the competition you can make 4 submissions.

Note that if you submission fails for some reason such as incorrect format, this may still count as one of the submissions.

After uploading your file, make sure it appears on the leaderboard (**Results** tab). Note that the error rate of your best-scoring submission should be *publicly visible* on the leaderboard *at all times*. Otherwise, the submission is not eligible for a bonus.

The submission file should be a .zip file with a file named `test.json` in it. Make sure there are not additional subdirectories in the zip file. Make sure the file is valid JSON.