CS7CS4/CSU44061 – Machine Learning
Academic Year 2024-2025
Trinity College Dublin

Final assignment 2024-25

- You must do this assignment entirely yourself - you must not discuss or collaborate on the assignment with other students in any way, you must write answers in your own words and write code entirely yourself. If you use any online or other external content in your report you should take care to cite the source. It is mandatory to complete the declaration that the work is entirely your own and you have not collaborated with anyone - the declaration form is available on Blackboard. All submissions will be checked for plagiarism.
- Include the source of code written for the assignment as an appendix in your submitted pdf report. Also include a separate zip file containing the executable code and any data files needed. Programs should be running code written in Python, and should load data etc when run so that we can unzip your submission and just directly run it to check that it works. Keep code brief and clean with meaningful variable names etc.
- Important: For each problem, your primary aim is to articulate that you understand what you're doing - not just running a program and quoting numbers it outputs. Generally most of the credit is given for the explanation/analysis as opposed to the code/numerical answer.
- If you use machine learning models not covered in the course then you must take care to show that you understand them and are not just running code in a "black box" fashion (so explain how predictions are generated from an input, what the cost function is, what the model parameters and hyperparameters are and how they affect the predictions etc).
- Reports should typically be about 5 pages, with 8 pages the upper limit (excluding appendix with code).
- The **submission** on Blackboard should include the report in PDF format, a selected generated output in MIDI format (only one), and the final Python code. Note that the PDF must be uploaded as a separate file (not as part of a zip file).

Part 1 [80 marks]

The assignment consists of repurposing the GPT script from week9 into a generative model for melodies. You are free to take whatever steps you might consider necessary or useful, as long as you explain your motivations. The exercise consists of repurposing that specific GPT code, so it is not acceptable to upload an entirely different code. Instead, you are asked to describe the steps to re-use that code to build a generative model for melodies, and then attempting to improve the model starting from that blueprint. The adjustments/improvement you should consider might involve reworking the model hyperparameters, model architecture, input dataset, tokenisation, vocabulary, evaluation metrics (appropriate justifications and evaluation are expected).

General instructions: Download the archives finalAssignment_GPT.zip and finalAssignment_musicDataset.zip from Blackboard. The former includes the GPT code (same as for the week9 assignment) that was built for processing/generating text data. The latter includes a music dataset, consisting the midi melodies for top music pieces from 1964 to 2018, as well as code for preprocessing those midi files into text files that can used (almost) immediately in the GPT code.

Task: Starting from that GPT script and melody dataset, build models for generating melodies:

1. Repurpose the GPT text model provided with minimal changes so that it can generate melodies. The music dataset provided is the file 'inputMelodiesAugmented.txt'. Indicate the

evaluation metrics you plan to use and explain how they are appropriate for this task. Your evaluation should include an appropriate baseline.

2. Improve the model from that original architecture. Explain your reasoning when deciding what could be changed for improving the outcome. Include appropriate objective evaluation of the improvement. If deemed appropriate, subjective evaluation may also be included (a script for playing the output is included). Changes should involve improvements on the dataset (e.g., augmentation), on the architecture (with appropriate justifications), and on the training hyperparameters.

Hint 1: Note that there are many ways to do this and angles to explore. Melodies involve at least two key dimensions i.e., time and pitch. You may decide to build your model one step at a time (e.g., so that it decoded pitch, ignoring time, or vice versa). Indeed, your evaluation will need to be adjusted appropriately.

Hint 2: Note that the dataset has been heavily simplified by restricting the music to a single melodic stream (rather than having multiple instrument), and by restricting the pitch to a single octave, thus reducing the vocabulary size.

Please find further details below on the preprocessing adopted in the finalAssignment_musicDataset.zip

[mark breakdown: (i) data preprocessing and feature engineering 20 marks, (ii) machine learning methodology 25 marks, (iii) evaluation 20 marks, (iv) report presentation 15 marks]

Part 2 [20 marks]

(i) What is an ROC curve? How can it be used to evaluate the performance of a classifier compared with a baseline classifier? Why would you use an ROC curve instead of a classification accuracy metric? [5 marks]

(ii) Give two examples of situations where a linear regression would give inaccurate predictions. Explain your reasoning and what possible solutions you would adopt in each situation. [5 marks]

(iii) The term ‘kernel’ has different meanings in SVM and CNN models. Explain the two different meanings. Discuss why and when the use of SVM kernels and CNN kernels is useful, as well as mentioning different types of kernels. [5 marks]

(iv) In k-fold cross-validation, a dataset is resampled multiple times. What is the idea behind this resampling i.e. why does resampling allow us to evaluate the generalisation performance of a machine learning model. Give a small example to illustrate. Discuss when it is and it is not appropriate to use k-fold cross-validation. [5 marks]

Further details on the music dataset preprocessing

You may decide to use the input music dataset in its current form ('inputMelodiesAugmented.txt'), or to augment it further. That is your choice. Whatever you decide to do, your choices must be described and motivated step-by-step in the report. If you decide to only work on the dataset provided, you don't need to go through the steps that follow. Nonetheless, it is useful that you understand what the original dataset was and how it was processed into the file 'inputMelodiesAugmented.txt'.

Please find the step-by-step description of how the music dataset was built.

- The '**BiMMuDa'** was downloaded and can be found in the finalAssignment_musicDataset.zip. The dataset contains melodies in .mid file (MIDI format) in the subfolder 'bimmuda_dataset'. Please find details, acknowledgments, and license inside the corresponding folder.
- **extractMelodies.py**: All files ending in '_full.mid' were copied to the 'musicDatasetOriginal' directory
- **midi2text.py**: All MIDI files in 'musicDatasetOriginal' are loaded, heavily simplified so that they only use notes within a single octave (vocabulary: C, C#, D, D#, E, F, F#, G, G#, A, A#, B, with the additional token 'R' representing a music rest), exported into a single text file 'inputMelodies.txt', where each line contains the melody in a given MIDI file. The simplified melodies were also converted back into MIDI files, so that you can hear what the simplified melodies actually sound like ('musicDatasetSimplified' folder).
- **augmentMidiTranslations.py**: The simplified input melodies in 'inputMelodies.txt' are augmented by applying pitch translations, for example by raising the pitch of every note by one step/semitone. (Note that it would also be possible to remove the spaces from the file as all tokens are single characters.) The output is stored in the file 'inputMelodiesAugmented.txt'.