INITIAL PLAN

REVERSE ENGINEERING AUDIO SYNTHESISER SOUNDS

CM3203 One Semester Individual Project - 40 Credits

AUTHOR

Harrison Taylor

TaylorH23@cardiff.ac.uk

SUPERVISOR

Prof. David Marshall

MarshallAD@cardiff.ac.uk

PROJECT DESCRIPTION

Traditional instruments such as a concert grand piano or an acoustic guitar have a fixed property called the timbre which allows the listener to instantly distinguish which note is coming from which instrument. Audio synthesisers are a class of instrument that produce sound by manipulating the output of an audio source through a series of effects, which are controlled via a set of parameters. The type of manipulation available to the instrument player differs based on the model of synthesiser, and the timbral information for the instrument can be described by a configuration of these parameters.

A configuration of synthesiser parameters is referred to as a patch, and is desirable to record as it is often difficult to recall the exact configuration of parameters for a certain sound. Collections of patches are often sold by sound designers who have spent a lot of time learning the characteristics of a certain model, and know how to reproduce a given sound.

The goal of this project is to use machine learning techniques to reverse engineer the effect that each parameter has a synthesiser's output sound. The hypothesis is that if a well designed ANN (Artificial Neural Network) is sufficiently trained on a set of patches with corresponding sample sounds, the network can then predict a patch that results in a reasonably close reproduction of some arbitrary input sound. The accuracy of a predicted patch relies on the capability of the audio synthesiser to produce the sound, and the quality of the ANN model.

A series of experiments will be conducted to determine the best type of data representation and neural network configuration as a general architecture for solving this problem. Although many types of synthesis methods exist, Subtractive and FM based synthesisers will be the main focus of this project.

Virtual Studio Technology is a standard interface for audio plugins and software instruments, often used with Digital Audio Workstations for producing music. For this project, synthesisers in the form of VST Instruments (VST**i**) will be used in experiments for convenience due to fast offline processing and easy manipulation via the VSTi API.

ETHICS

This project does not require any ethical approval as any data that is assessed is synthesised for the experiments.

AIMS & OBJECTIVES

PRIMARY OBJECTIVES

- **VSTi patch rendering**
  Produce or find an interface for a VST host that will return an audio clip in WAV format given a set of parameters and note information.

- **Neural Network Development - Stage 1**
  Develop a neural network architecture that can reliably predict, with a good degree of accuracy, an unseen audio clip from a simple VSTi that it has been trained on.

- **Neural Network Development - Stage 2**
  Improve upon initial neural network architecture by developing a general architecture that is able to produce models that reliably predict patches for a small range of different VSTi's. This stage is important to ensure that the network architecture performs well generally, rather than for just one type of VSTi.

- **Neural Network Development - Stage 3**
  Improve upon the developed neural network architecture to accommodate changes in note pitch. Synthesiser sounds are rarely identical in timbre across their possible range, so experiments need to be developed to ensure that the network can predict an accurate patch for a given pitch.

- **Neural Network Architecture Comparison Experiments**
  Perform experiments for a range of audio data representations and neural network architectures in order to determine the most effective configuration for the general task of reverse engineering audio synthesiser sounds

SECONDARY OBJECTIVES

- **Reliable selection of patches for timbral features of a VSTi**
  Each VSTi can produce a limited set of sounds based on it's parameters. In order for a neural network to identify the best patch for a given patch, a good representation of the effect that a parameter makes on the output sound is essential. There is often a "sweet spot" in which small parameter changes in a certain patch produce a wide array of sounds. Ideally there should be more testing patches around these "sweet spot" parameter configurations, and less testing patches around areas that contribute a smaller amount of change in the output sound.

- **Investigation into temporal changes of parameters**
  It is very rare in musical performance that a patch remains unmodified for the entirety of a song - often the patch is seen as a starting point followed by a change in parameters as the instrument is played. Taking these performance characteristics into account, investigation into developing a network architecture that can recognise these temporal changes would be an appropriate area of consideration.

- **Patch Preview application**
  Using a trained model for a given VSTi, produce a small application that takes as input an audio file to predict an appropriate patch for reproducing the audio file's sound using a selected neural network model. Upon processing, launch a visual version of the VSTi that accepts MIDI messages for a user to preview the patch.

- **Investigation into appropriate fitness functions**
  An investigation into what the most appropriate fitness function for classifying sounds may yield better performance in predicting patches. For example, a potential fitness function may select a predicted patch that is on average 60% correct for the duration of the sound over a predicted patch that is 90% correct for the majority for the sound but with a large amount of error for the remaining 10%. An investigation into different fitness functions may prove beneficial to the overall performance of an architecture.

WORK PLAN

All deliverables for the final project are due 11/5/17, giving me 15 weeks to fully complete the outlined objectives. The main implementation should ideally be completed by Easter Break, allowing for writing of the report and final experiments to be conducted in the final weeks of term.

A quick breakdown of scheduled objectives is defined below:

Week 1 - 29 Jan

- Research of previous work and familiarisation with VST tools

Week 2 - 5 Feb

- Research and learn about potential neural network architectures & data representations
- Start initial implementation of basic neural network architecture
- Investigation into patch selection for VST instruments

Week 3 - 12 Feb

- Further learning of neural network architectures & data representations
- Initial experiments and refinements of basic neural network architecture
- Development of framework for visualising patch differences using t-SNE

Week 4 - 19 Feb

Milestone 1: Successful patch prediction for simple synthesiser

- Investigation into generalising simple neural network for other synthesiser models
- Finalisation of planned experiments for architecture configurations
- Research fitness functions and construct experiments for ideal fitness functions

Week 5 - 26 Feb

- Initial experiments for different architecture configurations
- Refinement of architecture configurations
- Development of fitness function for optimal prediction

Week 6 - 5 Mar

- Continued experiments for comparisons of different architecture configurations
- Development of automatic selection of patches for VST instruments

Week 7 - 12 Mar

Milestone 2: Successful overall improvements in patch prediction

- Continued experiments for comparisons of different architecture configurations
- Begin considerations of temporal features and changes in played note for performanceWeek 8 - 19 Mar
- Investigate comparisons between network performance for different types of synthesisers - does one type of architecture perform better with a certain type of synth for example?
- Continued experiments and improvements to network layers & architectures

Week 9 - 26 Mar

Milestone 3: Main implementation Successfully completed

- Begin compilation of data and analysis of network performances
- Begin write up of final report
- Investigate strongest models for solving general problem for main focus

Week 10 - 2 April

- Begin experiments with sounds not generated by target synthesisers - how well can synthesisers emulate sounds of other instruments & synthesisers with help from the developed neural network?
- Continue write up of final report

Week 11 - 9 April

Milestone 4: Final implementation

- Perform tuning for better efficiency and performance of the network
- Continue write up of final report
- Repeat experiments as necessary

Week 12 - 16 April

- Final report writeup

Week 13 - 23 April

Milestone 5: Final Report Draft

- Deliver final report for review

Week 14 - 30 April

- Improve final report
- Repeat experiments if necessary
- Develop VST Host Preview application

Week 15 - 7 May

- Deliver Final Report draft for review
- Deliver Project code
- Submit Final Report & Project Code