Automatic Music Generation

Overview

This project is a software system that automatically generates musical compositions using machine learning techniques. It processes MIDI input files and trains a Recurrent Neural Network (RNN) model to learn musical patterns and structures. The trained model can then generate new musical sequences that align with the style of the input training data. The project uses Python, developed in the PyCharm IDE, and integrates various machine learning libraries.

Motivation

The project was developed as part of a professional certification exam, blending my passion for music with artificial intelligence. By training neural networks, specifically Long Short-Term Memory (LSTM) networks, I aimed to create a system that automatically composes music. Initially, I didn't plan on adding a user interface, but as the project evolved, I decided to make the application more user-friendly by adding a GUI.

Key Features

- Generates music compositions from trained neural network models.
- Reads and processes MIDI files for training.
- Implements Long Short-Term Memory (LSTM) neural networks.
- Provides an intuitive GUI for users to interact with the application and generate music.
- Supports compositions based on well-known classical composers.

Project Structure

- **PyCharm**: The project is developed in PyCharm, which provides an integrated development environment for Python with features for debugging and project management.
- **Neural Network**: A Recurrent Neural Network (RNN) with LSTM layers is used for music generation.
- **GUI**: A Tkinter-based graphical user interface enables users to easily interact with the system.

Libraries and Tools Used

1. **music21**: For parsing and analyzing MIDI files.
2. **glob**: To handle file operations and directories.
3. **tqdm**: Displays progress bars during training.
4. **numpy**: For efficient data manipulation.
5. **random**: For generating random inputs during testing.

6. **tensorflow.keras**: Used to build and define the LSTM model.
7. **sklearn**: For splitting data into training and test sets.
8. **tkinter**: For creating the GUI.
9. **PIL**: For image processing in the GUI.

How it Works

1. MIDI File Processing

The system reads MIDI files from a specified directory and extracts musical notes and chords. These are then converted into a numerical format suitable for input into the neural network.

2. LSTM Model

An LSTM-based neural network is trained on the processed MIDI data. This model learns to predict the next note in a sequence based on previous notes.

3. Music Generation

Once trained, the system can generate new sequences of music based on the patterns it has learned. The output is a MIDI file containing the generated composition.

4. User Interface

The application provides a graphical interface that allows users to:

- Select the folder with the MIDI files of their favorite composers.
- Start the music generation process.
- Save the generated music as a MIDI file.

Future Improvements

- **Improved User Interface**: Making the GUI more interactive and visually appealing.
- **More Composer Choices**: Adding more composers and expanding the training data.
- **Customizable Parameters**: Allow users to adjust training parameters like batch size, epochs, and model architecture.

References

- **Music21 Documentation**: https://web.mit.edu/music21/doc/
- **TensorFlow Documentation**: https://www.tensorflow.org/
- **MIDI Standard**: https://www.midi.org/