Project Proposal: Sheet Music-to-Audio Generation via Deep Learning

Overview

We aim to build an end-to-end deep learning pipeline that takes sheet music as input and generates the corresponding musical output. The process is divided into two major phases:

1. **Optical Music Recognition (OMR)** — extracting symbolic music (e.g., notes, rhythms) from images or PDF sheet music
2. **Music Generation** — Using Transformer-based deep learning models to interpret symbolic tokens derived from input files and generate the corresponding musical audio.

This project blends Computer Vision, Sequence Modeling (NLP), and Symbolic Music Generation, providing an excellent opportunity to gain essential skills in Deep Learning and Transformer architectures.

Project Phases

Phase 1: Optical Music Recognition (OMR)

Goal: Parse sheet music (PDF or image) into a structured format such as MusicXML or MIDI.

Methods:
- Utilize open-source tools such as Audiveris to extract structured MusicXML from scanned PDFs or images.
- Use music21 to verify, inspect, and clean MusicXML outputs, ensuring accuracy and readiness for the next phase.
- To incorporate more computer vision, experiment with training a custom neural OMR system using deep learning models (e.g., CNNs or Vision Transformers) on datasets like DeepScoresV2 or MUSCIMA++.
- Optionally, implement a music symbol detection pipeline using object detection frameworks (YOLO, Detectron2) to localize and classify individual music notation elements.

Skills involved: Computer Vision, object detection/recognition, preprocessing
Note: Datasets like DeepScores are large, and training deep models without a GPU will significantly increase training time.

Phase 2: Music Generation via Transformer

Goal: Use a Music Transformer to generate music based on symbolic input (notes, chords, durations).

Methods:
- Use symbolic datasets like MAESTRO, JSB Chorales, or Lakh MIDI for training.
- Tokenize MIDI or MusicXML data into sequences of note events using music21,

pretty_midi, or Magenta-style tokenizers.
- Fine-tune a Transformer model (e.g., Magenta's Music Transformer or a custom PyTorch implementation).
- Generate music using the note_seq library and synthesize audio with a library such as FluidSynth for playback and evaluation.

Skills Involved: Sequence modeling, tokenization, deep learning with Transformers, audio synthesis.

Alternative Unified Approach: TrOMR

As an extension or alternate path, we can explore **TrOMR (Transformer for Optical Music Recognition)**, a research model that unifies both phases:

- Single end-to-end Transformer that takes in sheet music images and outputs MIDI token sequences.
- Uses a Vision Transformer (ViT) or CNN encoder + Transformer decoder.
- More research-intensive but allows for elegant, multimodal training.
- Suitable for GPU-enabled environments (e.g., Colab Pro, university clusters, Nvidia GDX).

Project Goals

- Build a multimodal pipeline combining computer vision and sequence modeling: Sheet music → MusicXML/MIDI → Generated music → Audio synthesis.
- Implement and evaluate both rule-based and deep learning-based OMR approaches.
- Train and fine-tune a Transformer model to generate accurate musical output from symbolic tokens.
- (Optional) Experiment with an end-to-end model like TrOMR to unify vision and generation phases.
- Output: musical MIDI/audio that corresponds to sheet input

Datasets

- **DeepScoresV2** (sheet music images + symbol labels)
- **MUSCIMA++** (handwritten OMR)
- **MAESTRO** (polyphonic piano MIDI)
- **JSB Chorales, Lakh MIDI,** or **GiantMIDI** (additional symbolic music datasets for training and evaluation)

Limitations & Challenges

- **Hardware Constraints**: Neural OMR and Transformer training is resource-intensive; access to GPUs or cloud-based solutions like Google Colab might be essential.
- **Data Complexity**: Working with image-based, symbolic, and audio data adds overhead in preprocessing and alignment.
- **Model Integration**: Combining computer vision, symbolic parsing, and generation requires careful modular design.

- **Learning curve**: The project spans multiple technical domains including vision, NLP, audio processing, and music theory.

Why This Is a Great Project

- **Multimodal Model**: Combines computer vision, symbolic music, and generative models
- **Real-world relevance**: Sheet music digitization and AI music generation are active research areas
- **Relevant skill growth**: Gain experience in PyTorch/TensorFlow, symbolic music, tokenization, Transformers, and audio synthesis.

Next Steps

1. Set up initial OMR pipeline using Audiveris + music21
2. Explore training a deep learning-based OMR model using DeepScoresV2 or MUSCIMA++.
3. Parse and tokenize MusicXML/MIDI files for input to the Transformer model.
4. Train and evaluate the Music Transformer using symbolic datasets.
5. Generate music sequences with libraries like note_seq and synthesize audio using FluidSynth.
6. Optional: experiment with vision-to-music (TrOMR) pipeline

Tools that can help us

To support the development, debugging, and tracking of our multimodal pipeline, we could use the following tools:

- **Weights & Biases (wandb)**: For experiment tracking, logging training progress, visualizing loss/accuracy metrics, and comparing model runs.
- **Google Colab**: For training the models on a cloud GPU.
- **Hugging Face**: Prebuilt model wrappers, tokenizers, and training utilities. We can port Music Transformer-style models to the Hugging Face ecosystem using PyTorch Lightning or Trainer API.
- **Kaggle**: For running experiments on free GPUs, finding datasets, hosting tokenized or symbolic datasets, and collaborating via cloud-based notebooks.
- **TensorBoard**: As an alternative for tracking training metrics locally.
- **MuseScore / ScoreCloud:** To download/generate sheet music for testing.

Pretrained Models we can use

To accelerate development and avoid training from scratch, we could use or fine-tune the following pretrained models. These include both music generation models and optical music recognition (OMR) models:

Music Generation:

- **Magenta's Music Transformer**: A transformer-based model trained on the MAESTRO dataset for polyphonic piano generation.
- **PopMAG**: A chord-conditioned melody generation model that can serve as an additional benchmark.

- **REMI-based Transformers:** Models using REMI-style tokenization formats supported by toolkits like miditok and muspy.
- **MusicGen (Meta)**: A text-to-music model capable of symbolic and audio generation, considered for future exploration.

Optical Music Recognition (OMR):

- **Deep Watershed Detector (DWD)**: A fully convolutional model that was state-of-the-art on the original DeepScores dataset. For DeepScoresV2, it was trained on full-resolution data using a crop-and-reassemble strategy. DWD does not support detection of overlapping objects sharing the same center, such as staff lines and ledger lines, which were disabled during its evaluation
- **Faster R-CNN with HRNet Backbone**: A novel combination introduced to the OMR field, using a standard Faster R-CNN framework enhanced with the HRNet backbone to preserve high-resolution feature representations. This model achieved very high Average Precision on common symbol classes and sets a new benchmark for object detection in typeset music.

These two pre-trained models and their benchmarks represent the primary contributions of the DeepScoresV2 paper beyond the dataset itself. They are publicly available along with code and instructions at: https://zenodo.org/record/4012193

Future Works

1. Extend the pipeline to generate sheet music from the generated audio or MIDI (closing the loop).

References and Learning Resources Works

The sound of AI YouTube Channel
Music Generation using Music Transformers
Tristan Behrens Music Transformer
DeepScoresV2 and MUSCIMA++ for OMR dataset exploration