

Melody → String Arrangement with Rule-guided Seq2Seq Transformer

Håkon Ganes Kornstad - FYS5429

The diagram illustrates a musical transformation process. On the left, a single melodic line for Violin Solo is shown in G major with a dynamic marking of **p**. An arrow points from this to the right side, where a six-part string arrangement is presented. The arrangement includes Violin Solo, Violin 1, Violin 2, Viola, Violoncello, and Contrabass. The Violin Solo part is marked *espressivo* and **p** con sordino. The Violin 1 and Violin 2 parts are marked **pp** con sordino. The Viola part is marked **pp** con sordino and includes a dynamic marking of **pp** pizz. con sordino. The Violoncello and Contrabass parts are marked **p** pizz. con sordino. A dynamic marking of **p** is also present at the bottom of the staff.

I want to use a Transformer Model to create an algorithm that takes a monophonic melody as input, and renders a multi-track string arrangement as output.

- Core challenge: Joint modelling of vertical harmony (**voicing**)
+ instrument-idiomatic, horizontal smooth lines (**voice-leading**).
- Melody instrument should be Violin, with the rest of the parts playing accompaniment. So called *divisi* can be used to split up into further sub-parts: {vln1a, vln1b, vln2a, vln2b, ...}

Data & Modelling

- **Dataset:** **PDMX** (large-scale public-domain MusicXML)
Train on *broad polyphonic textures* (strings, winds, choir, piano) to maximize data.
- **Supervision:** Extract **melody = top-line**. Remaining voices become **accompaniment**.
Optionally allow *divisi/doublings*.
- **Neural model:** Transformer encoder–decoder (Seq2Seq)
Transformer takes one sequence as input (the melody) and generates another sequence as output (the arrangement). The encoder-decoder structure reads and summarizes the melody into an internal representation, and generates the accompaniment step by step.
- **Must have-constraints:** instrument ranges, no voice crossing, avoid impossible jumps.
- **Nice-to-haves:** penalize awkward leaps; reward things like open strings on long notes or strong beats.
The idea is that even if the model learns “general polyphony” from lots of data, the final output is guided toward something that is idiomatic and playable for strings.

Evaluation and Timeline

- **Evaluation:** voice-leading metrics (leaps/range/crossing) + string-idiomaticity (open-string rate, low-register density) + qualitative MIDI/score examples. **Visual inspection by musicians.**
- **Timeline:**
 - **Feb:** Data inspection, parsing + filtering (strings / top-line extraction), dataset pipeline
 - **Mar:** baseline + metrics pipeline
 - **Apr:** Transformer training
 - **May:** Additional constrained decoding + evaluation/ablations
 - **Jun:** write-up + demos