Dathoven

Introduction

Human inspiration has always been an elusive companion. Every artist has experienced at some point in time the lack of original ideas or the inability to develop a good idea from its birth to its destiny as a masterpiece.

Dathoven is a computer-aided music composition system that tries to help music composers to explore different melodic ideas faster and easier, even on those days when it seems like inspiration is nowhere to be found!

State of the art

There are lots of different apps for music composition and lots of projects of music generation based on machine learning algorithms.

MuseGAN

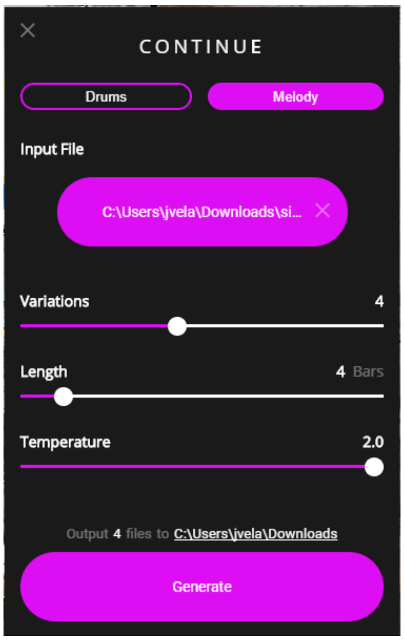
MuseGAN is a project on music generation. In a nutshell, we aim to generate polyphonic music of multiple tracks (instruments). The proposed models are able to generate music either from scratch, or by accompanying a track given a priori by the user.

MuseGAN is a project that's able to generate music with multiple instruments and tracks using GAN.

Magenta Studio

Magenta Studio is a collection of music plugins built on Magenta's open source tools and models. They use cutting-edge machine learning techniques for music generation.

Built on Tensorflow, Magenta Studio has plugins that provide different music generation functionalities. You can interpolate two different songs, generate new bars from example ones, etc.



Input File

C:\Users\jvela\Downloads\si...

Variations 4

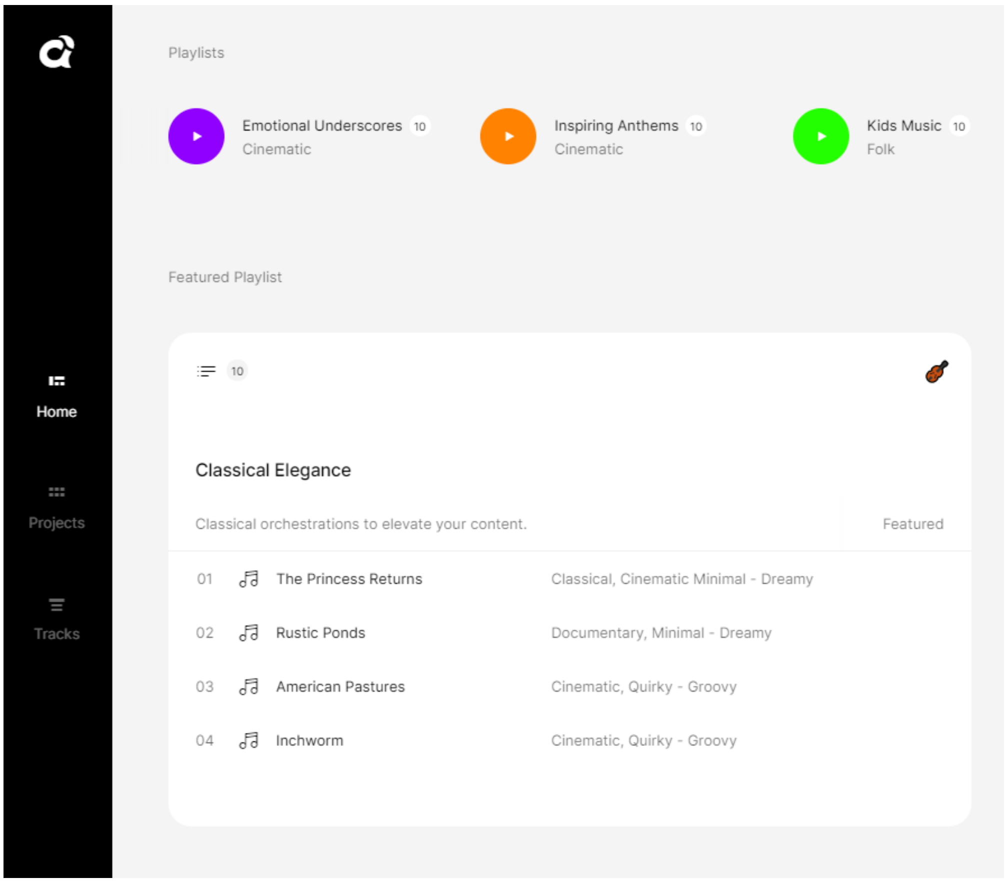
Length 4 Bars

Temperature 2.0

Output 4 files to C:\Users\jvela\Downloads

Amper Music

[Amper music]() is a web application that lets you create whole instrumental songs of very high quality establishing parameters such as music genre, tempo, etc. The resulting songs are truly amazing.



Playlists

Emotional Underscores 10 Cinematic

Inspiring Anthems 10 Cinematic

Kids Music 10 Folk

Featured Playlist

10

Classical Elegance

Classical orchestrations to elevate your content.

Featured

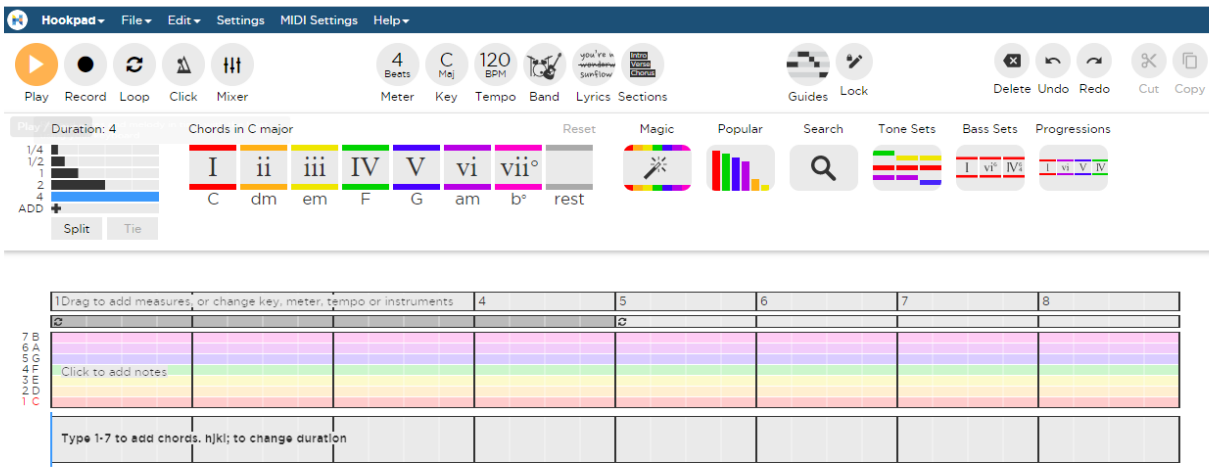
01	The Princess Returns	Classical, Cinematic Minimal - Dreamy
02	Rustic Ponds	Documentary, Minimal - Dreamy
03	American Pastures	Cinematic, Quirky - Groovy
04	Inchworm	Cinematic, Quirky - Groovy

Hooktheory

Hook theory is a company that creates digital tools for music education and composition, its product Hookpad can be described as:

A musical sketchpad with built-in music theory that helps you create beats, songs, and musical snippets

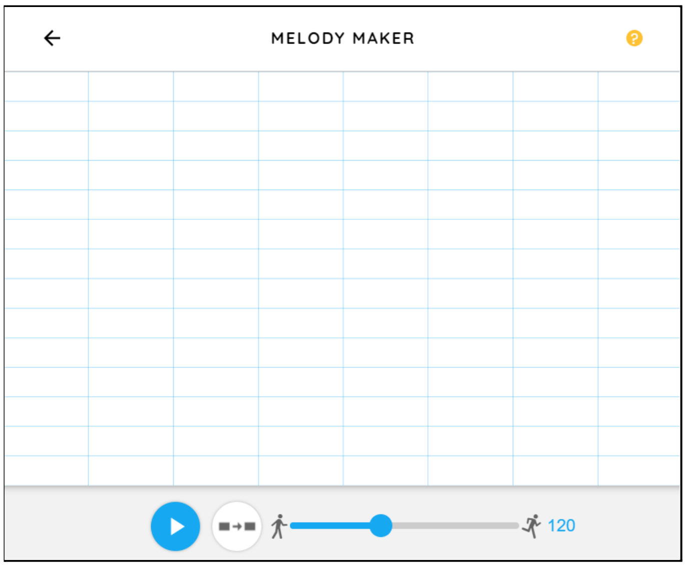
It is a really nice tool that helps the composer try different ideas faster and easier. This tool has a chord suggester algorithm integrated but it does not have the same tool for melody suggestions.



Melody Maker

Melody Maker is one of the mini apps included in Google's Chrome Music Lab project.

Chrome Music Lab is a website that makes learning music more accessible through fun, hands-on experiments.



Melody maker has a way of manipulating the melodies created but it’s suggestions only repeat the previous notes. It’s interface is really appealing and easy to use and it’s been the base for Dathoven’s current interface implementation.

Why Dathoven

Dathoven's goal is to make simple suggestions about melodic ideas in a very accessible way without the need of having previous musical knowledge.

It could be seen as the functionality that Hookpad is lacking, the suggestion of possible ways to continue a certain melody. It could also be seen as a combination of Chrome Music Lab’s Melody Maker and Google’s Magenta Studio bars generation plugin.

Data Acquisition

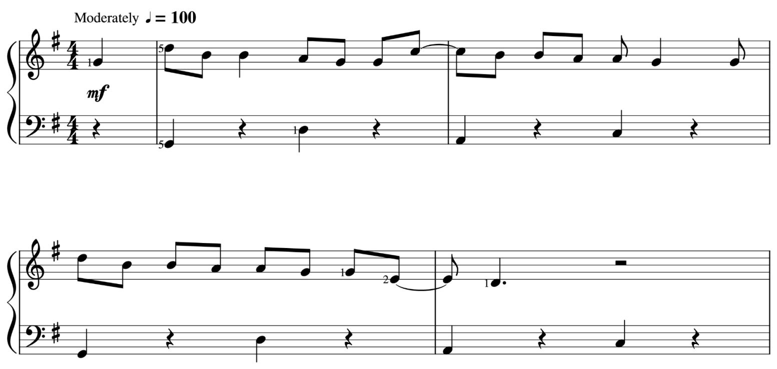
Exploring possible sources

In order to build a machine learning system that can make predictions of possible music melodies we need to gather data of real melodies that the system can learn from.

During this phase different possible sources of information were explored. The three main possibilities were:

- Sound files: extracting the actual notes of sound files is not a trivial task and could be the subject of a whole new data science project.

- Music sheets: there are lots of music sheets on the internet but they are not standardized and extracting the musical information would require some effort.



- MIDI: MIDI files are lightweight, standardized and there are tons of free MIDI files available on the internet.

MIDI

MIDI was the chosen option.

Gathering MIDI Files

There are lots of MIDI files freely available on the internet but they are not so easy to download and gather. In order to get enough MIDI files it was necessary to use Web Scraping techniques to extract the files from different websites.

The websites that were used as MIDI file sources are: Carlo's MIDI
The process of MIDI Web Scraping can be followed in this jupyter notebook:

The result of this Data Acquisition phase is a Dataset with hundreds of songs in midi format that can be found in this shared folder.

Data Extraction and Cleaning

MIDI Files

According to wikipedia *MIDI is a technical standard that describes a communications protocol, digital interface, and electrical connectors that connect a wide variety of electronic musical instruments, computers, and related audio devices for playing, editing and recording music.*

The MIDI specification of 1996 has more than 300 hundred pages, since then, there have been changes and updates to the original protocol. Despite it being quite an interesting topic we will not get in depth into the MIDI protocol in this document.

Complete MIDI 1.0 Detailed Specification v. 96.1 3rd Ed. (PDF)

1 / 334

The Complete MIDI 1.0
Detailed Specification
Incorporating all Recommended Practices
document version 96.1
third edition

For the purpose of this project what we need to know about MIDI are the following points:

- Standard MIDI files describe a sequence of MIDI events in time
- These MIDI events define every aspect of the sound and the song metadata
- “Note on” and “note off” are the events that define the notes that sound in each moment
- Note events have three attributes: Note number, Velocity and Channel
    - Note number: it gives us the pitch of the note. This means if it’s a C, a D or F# note.
    - Velocity: it can be thought of as the volume of the sound
    - Channel: MIDI files have up to 16 channels that can be routed to different sound modules (i.e. instruments).
- Channel 10 is reserved for percussion

MIDI in python: Music21

In order to extract the data from MIDI files and use it in our machine learning algorithms some different python libraries for MIDI were explored. After some trial and error (see relevant notebook) the chosen library was Music21:

music21: a toolkit for computer-aided musicology

Music21 is a set of tools for helping scholars and other active listeners answer questions about music quickly and simply.

Music21 is a library developed by MIT that has a lot of amazing tools for music analysis. One of these tools is its midi module which lets us process MIDI files and convert its musical information into the music classes defined in Music21. Objects in Music21 are well designed and let us manipulate and extract the information we need from them.

Extracting melodies

Using Music21 each of the channels in the MIDI files is converted to a sequence of notes at a certain point in time and with a certain duration.

For this purpose the steps taken are:

- Percussion events are discarded (channel 10 in MIDI files)
- Events are divided by channel, assuming melodic phrases are on the same instrument
- Chord events are simplified and reduced to their lowest note

After this processing the result is a dataset of melodies with different lengths extracted from each of the songs.

The detailed implementation of these steps can be followed in this notebook.

Analysing and cleaning the dataset

With the amount of data that we are dealing with, and having in mind that the information has been downloaded from public websites it is important to explore the data in order to find possible corrupted data that could mislead our machine learning algorithms in future steps in the process.

Analysing the information on our song dataset we can see that some of the tracks have pieces of melody that are too far apart from each other. By manipulating the data we are able to create compact pieces of melody that are better suited for machine learning training.

Some of the songs presented wrong information. There were notes with no time duration that were deleted from the dataset.

The detailed implementation of these steps can be followed in this notebook.

Once we have created our dataset we can start to develop machine learning algorithms to try and learn from the melodies we have in order to create new ones that can be used for Dathoven.

Machine Learning

Describing the problem

We have a dataset of sequences of notes that contain:

- Pitch: a number describing the pitch of the note in Music21
- Absolut offset: time since the start of the song
- Duration: time that the note is audible

Our goal is to be able to develop an algorithm that receives a note sequence and can continue this sequence in a way that the resulting sequence is similar to the sequences present on the dataset.

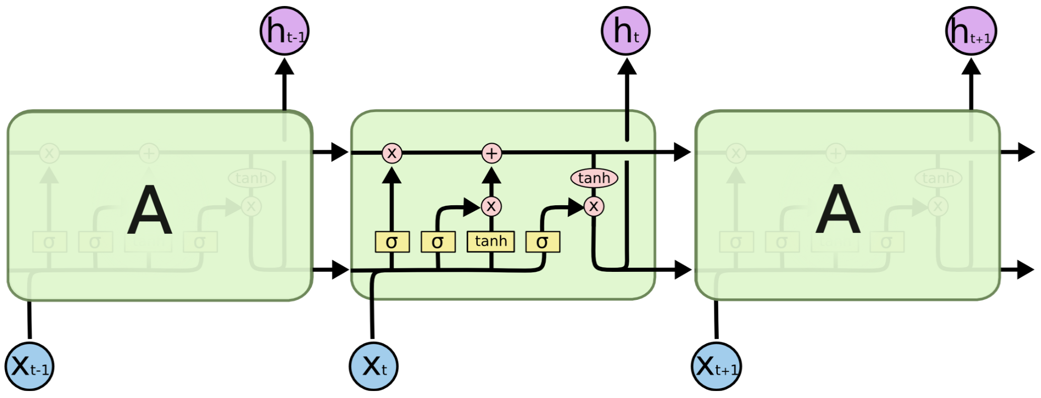
For this purpose the approach is to create an algorithm that is able to make a recommendation for the next possible note given a seed sequence. Applying this algorithm recursively on the sequence we can generate melodies of the desired length.

This problem is related to time series problems and recommendation systems.

Machine learning model

We need a machine learning algorithm that can manage multiple features, sequences and large amounts of data. One of the most popular choices for this kind of machine learning problem are LSTMs:

Long Short Term Memory networks – usually just called "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in following work.1 They work tremendously well on a large variety of problems, and are now widely used.



Feature engineering

There have been two main paths that have been explored to tackle this melody generation problem:

- Absolute approach: the data passed to the neural network are the pitch, offset and duration of the note:
    - Pitch: a number describing the pitch of the note in Music21
    - Absolut offset: time since the start of the song
    - Duration: time that the note is audible

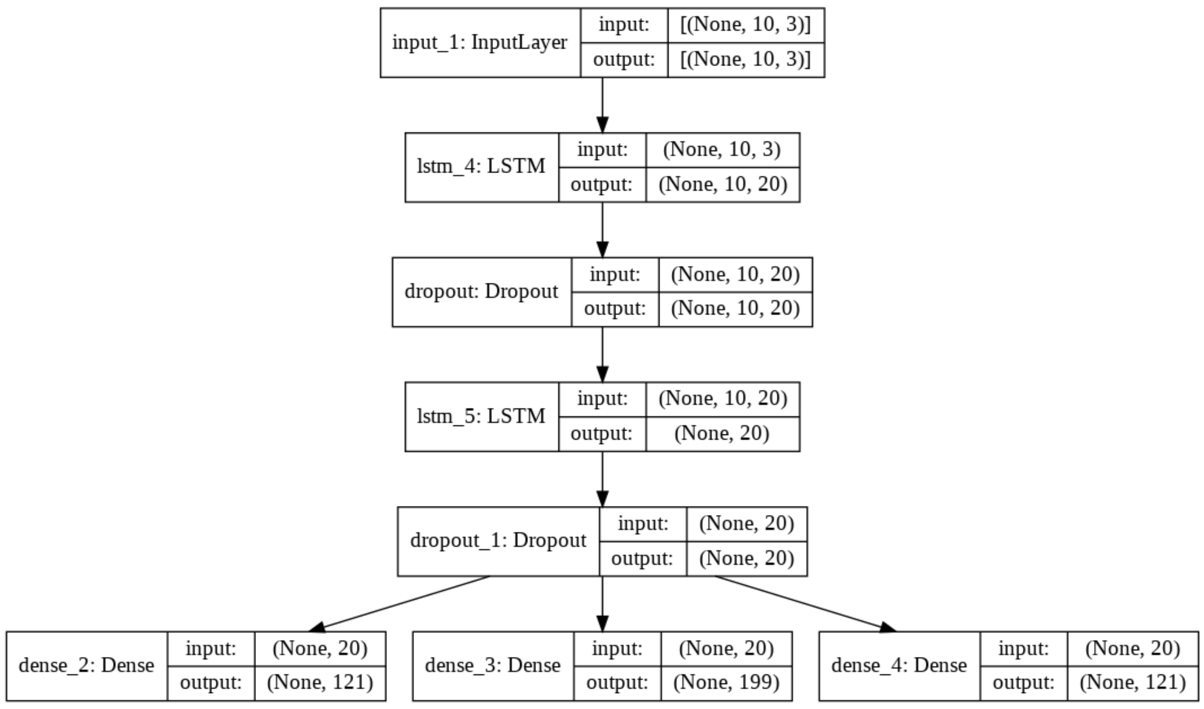
- Incremental approach: the data passed to the neural network are the increments between different notes. The data are the increments between the different notes:
    - Interval: difference between last note pitch and current note pitch

- Offset: time passed since the beginning of the previous note
- Duration: same value as in absolute approach

After some trial and error, the incremental approach seemed to get better results and it was the chosen approach for the final model.

Network Architecture

The last architecture of the network is formed by two lstm layers, two dropout layers and 3 dense output layers for each of the dimensions to predict.



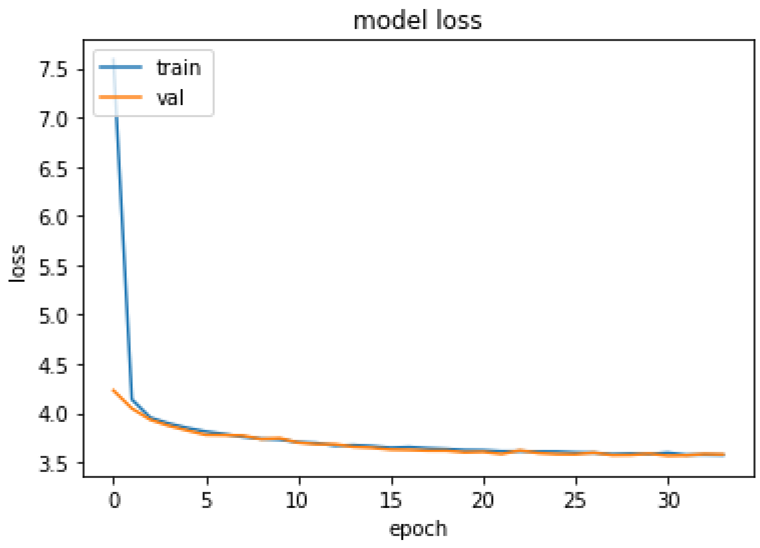
Evaluation function

The evaluation function used to train the model was the Sparse Top K Categorical Accuracy with a K of 5. In the best case scenario our model will be able to situate at the top 5 positions of its recommendation the actual next value of the melody.

In music there’s not a unique right answer and that’s why the model is trained to give a few good possibilities to follow the melody sequence.

Results

This is the notebook where the last model tried can be followed.



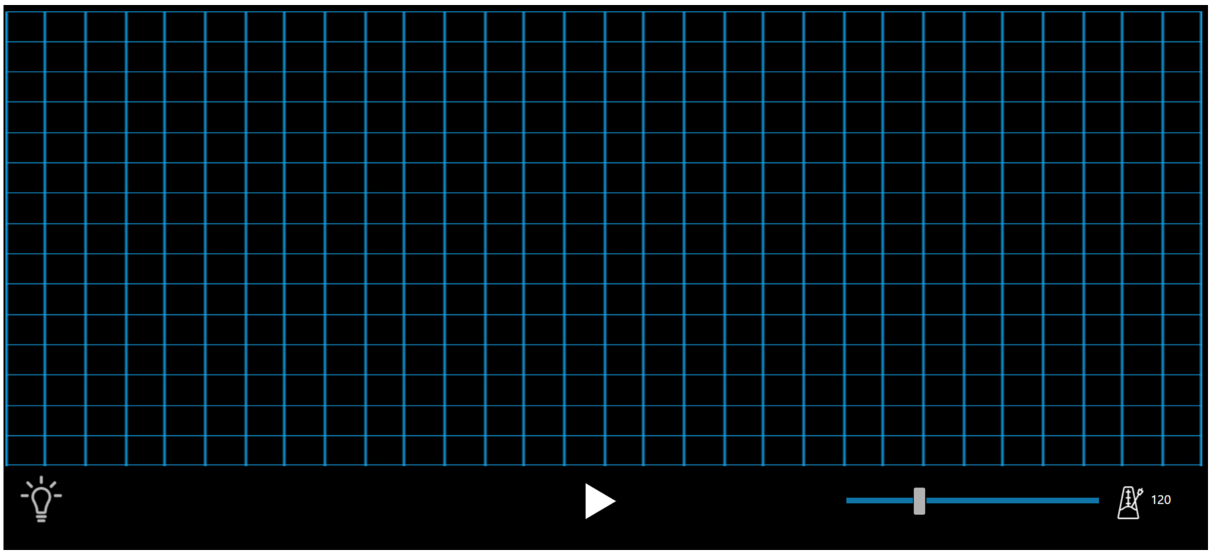
The model seems to be able to learn from the data and improves its performance on the validation set until the 30th epoch.

Front End

The front end is a crucial part of this project. It makes the machine learning algorithm useful and accessible.

Inspired by the ideas behind Hookpad and Melody Maker of Chrome Music Lab the approach for Dathoven was to create an interactive web app that could enable the users to create original melodies easily.

One of the best things about Chrome Music Lab is that it is open source and all its code is public. Melody Maker was developed 5 years ago in vanilla javascript. Starting from the source code of Melody Maker app the app has been transformed into a React application with the specific needs for the Dathoven project.



React

React is a *JavaScript library for building user interfaces.* It is one of the main trends in web development.

The repository with code for the react application is here.

Tone js

Tone.js is a Web Audio framework for creating interactive music in the browser. The architecture of Tone.js aims to be familiar to both musicians and audio programmers creating web-based audio applications.

Tone js is the javascript library used in the app to make it sound in the browser.

TensorFlow js

In order to use on the web app the models created in the jupyter-notebooks it was necessary to use TensorFlow js.

TensorFlow.js is a JavaScript Library for training and deploying machine learning models in the browser and in Node.js.

The steps taken to get the model into the web app were:

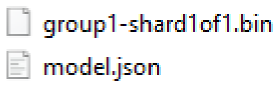
- Create a new conda environment, without an installation of tensorflow
- Install tensorflow on the clean environment:

```
pip install tensorflowjs
```

- Execute the converter on the model saved form python:

```
tensorflowjs_converter --input_format keras .\Model.h5
.\Model.h5js
```

- The result is a folder with the js model:



group1-shard1of1.bin
model.json

- In order to use it from the web app the model has to be loaded using a request from a url. In this app the web app and the model are hosted on S3 in Amazon Web Services.

Amazon Web Services

When looking for a way to make the app available for anyone, different options were evaluated. I chose AWS because I thought it could be a nice opportunity to learn a little bit about one of the main cloud options out there.

The S3 service is pretty intuitive and easy to use. The web app is hosted there.

In order to be able to deploy easily the application to S3 the package json configuration of the react app file was modified:

```
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject",
    "deploy": "aws s3 sync build/ s3://dathoven"
  },
```

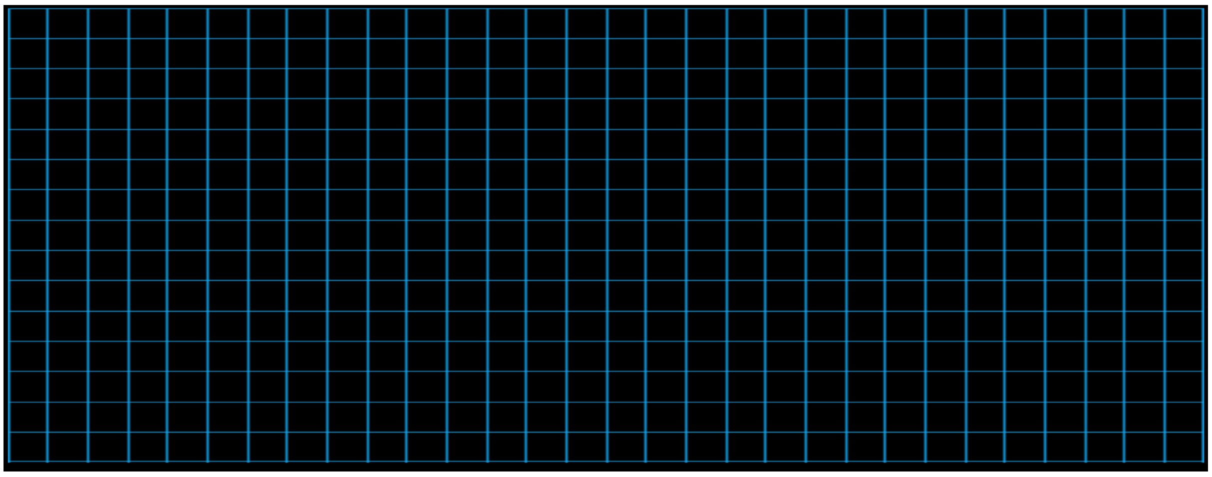
Using the aws cli with a single line the app is built and deployed to s3:

```
npm run build && npm run deploy
```

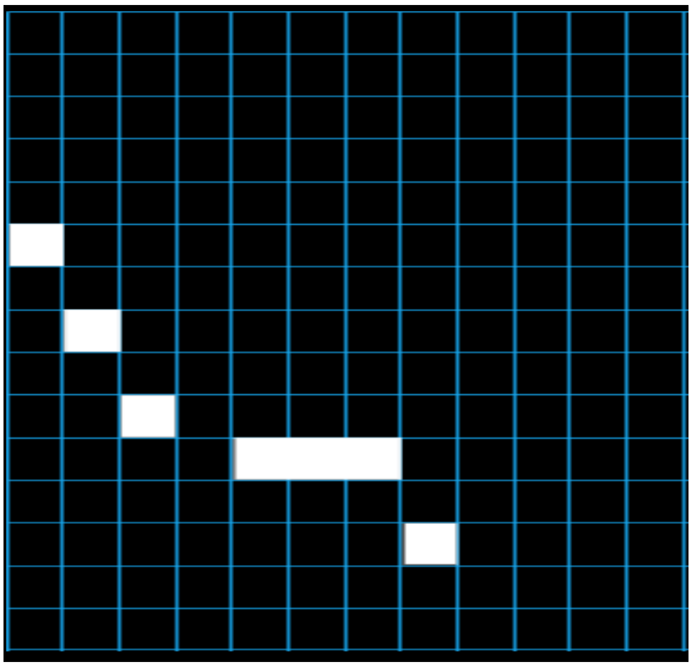
Here is the web application.

How to use the frontend

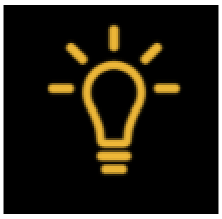
- On the central part of the app we have the composition grid.



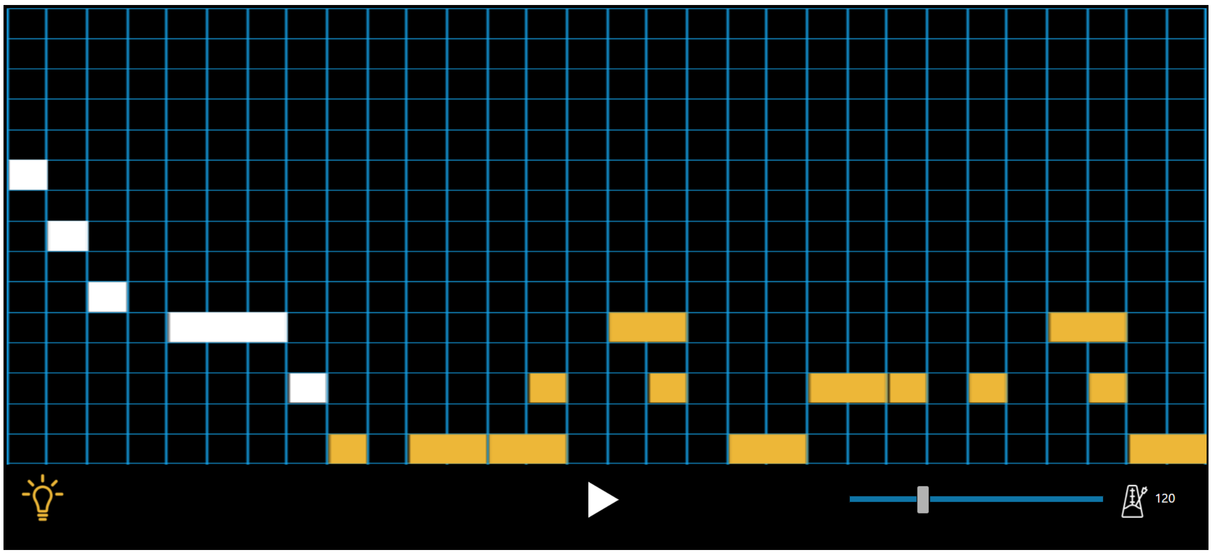
- We can add notes by clicking on the squares, if we click and drag we can create notes of length bigger than one time step



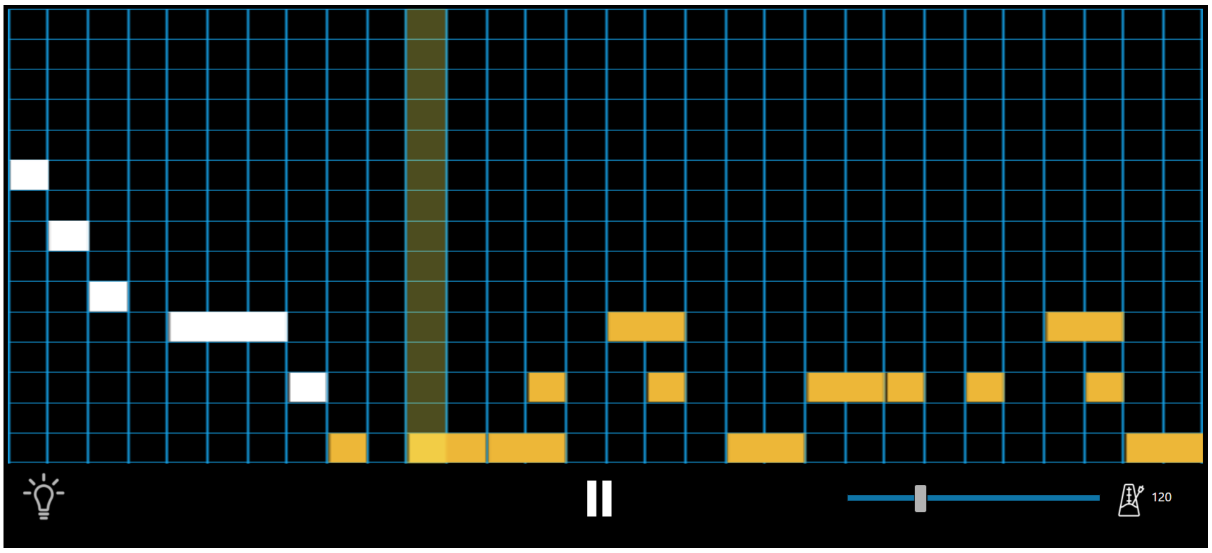
- Whenever we have a melody of two or more notes the lightbulb on the left-down will light up. Dathoven is ready to create a suggestion:



- If we click on the lightbulb a suggestion will be made:



- We can click on the play/pause button to start or stop the song playing:



- On the right-down corner of the app we can control the metronome, i.e. the speed with which the song will be played

Conclusions

The main goals of the project were accomplished. All the phases of a Data Science project were executed and it was a great opportunity to learn.

It is hard to evaluate the resulting melodies created by Dathoven, sometimes they seem to be more human, sometimes they seem to be more random. This problem of music generation is still a problem that's pretty hard to solve, but despite the current limitations an app that could be useful to inspire composers was created.

References

MIDI

- MIDI Specification
  https://www.midi.org/specifications-old/item/the-midi-1-0-specification

- Dr. Jason Freeman Course about MIDI
  https://www.youtube.com/watch?v=VTxjODp5oYs

Sequence Modeling with Neural Networks

- MIT class
  https://www.youtube.com/watch?v=CznICCPa63Q

- Deep learning book chapter
  https://www.deeplearningbook.org/contents/rnn.html

- Understanding LSTMs
  http://colah.github.io/posts/2015-08-Understanding-LSTMs/

- Keras Guide for RNNs
  https://www.tensorflow.org/guide/keras/rnn

- Understanding Keras LSTMs
  https://stackoverflow.com/questions/38714959/understanding-keras-lstms/38737941#38737941

Similar projects

- How to Generate Music using a LSTM Neural Network in Keras
  https://towardsdatascience.com/how-to-generate-music-using-a-lstm-neural-network-in-keras-68786834d4c5

- Chord Suggester
  https://medium.com/@huanlui/chordsuggester-i-3a1261d4ea9e

- Generating Music using an LSTM Network
  https://arxiv.org/ftp/arxiv/papers/1804/1804.07300.pdf

- A geometrical distance measure for determining the similarity of musical harmony
  https://link.springer.com/article/10.1007/s13735-013-0036-6

- Kaggle project: MIDI music data extraction using Music21
  https://www.kaggle.com/wfaria/midi-music-data-extraction-using-music21

- Zeldic musical RNN
  https://colab.research.google.com/github/cpmpercussion/creative-prediction/blob/master/notebooks/3-zeldic-musical-RNN.ipynb#scrollTo=gW812cV36ajE

Frontend

- Example using magenta
  https://github.com/notwaldorf/example-magenta-in-ts

- Tone js in react

https://react.christmas/2020/15

- Tensorflow js in react
  https://towardsdatascience.com/how-to-use-tensorflow-js-in-react-js-sentimental-analysis-59c538c07256

- CORS policy
  https://www.youtube.com/watch?v=dRHqGTOI3Ik

- Keras to Tensorflow js
  https://www.tensorflow.org/js/guide/layers_for_keras_users

- AWS CLI config
  https://aws.amazon.com/es/premiumsupport/knowledge-center/s3-locate-credentials-error/