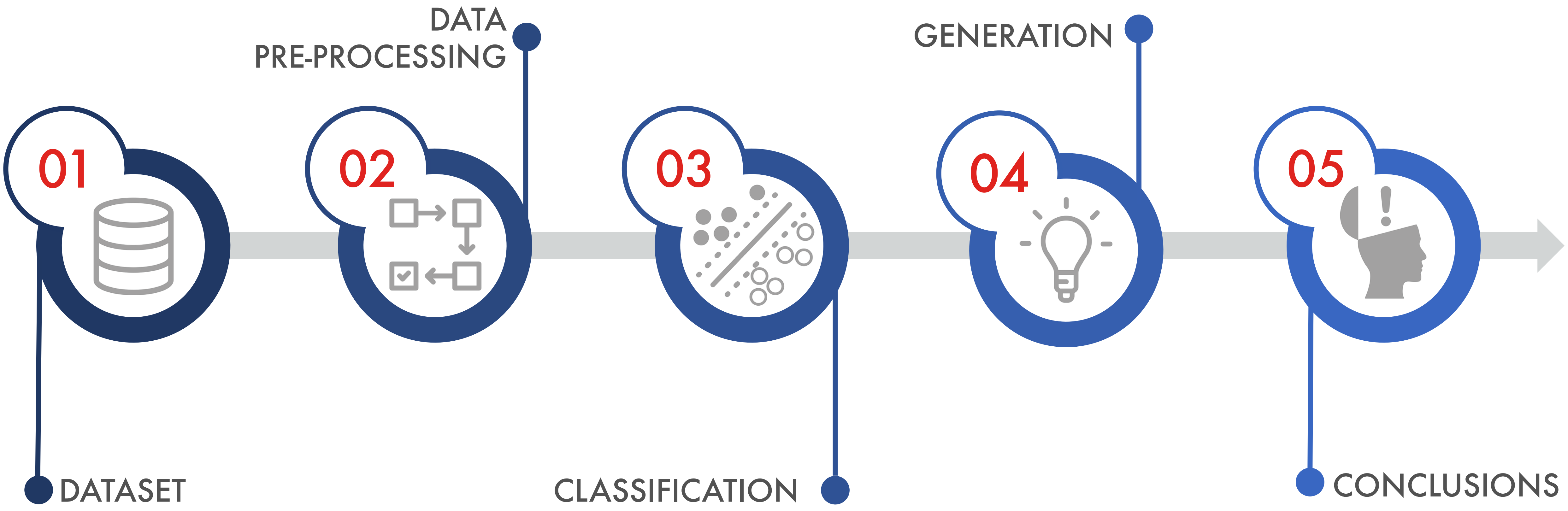
Classical Music Dataset

Honey Badgers Team

Agenda


01
DATASET
02
DATA
PRE-PROCESSING
03
CLASSIFICATION
04
GENERATION
05
CONCLUSIONS

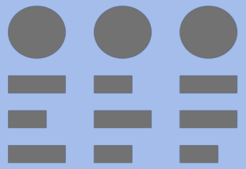
Dataset

Why This Dataset

- Available Data
- Interesting and Unusual Task
- ML Friendly Format
- Structured but Unpredictable

Explaining Dataset

- Collection of classical polyphonic pieces.
- 7134 unique pieces



- Contains 3 columns:

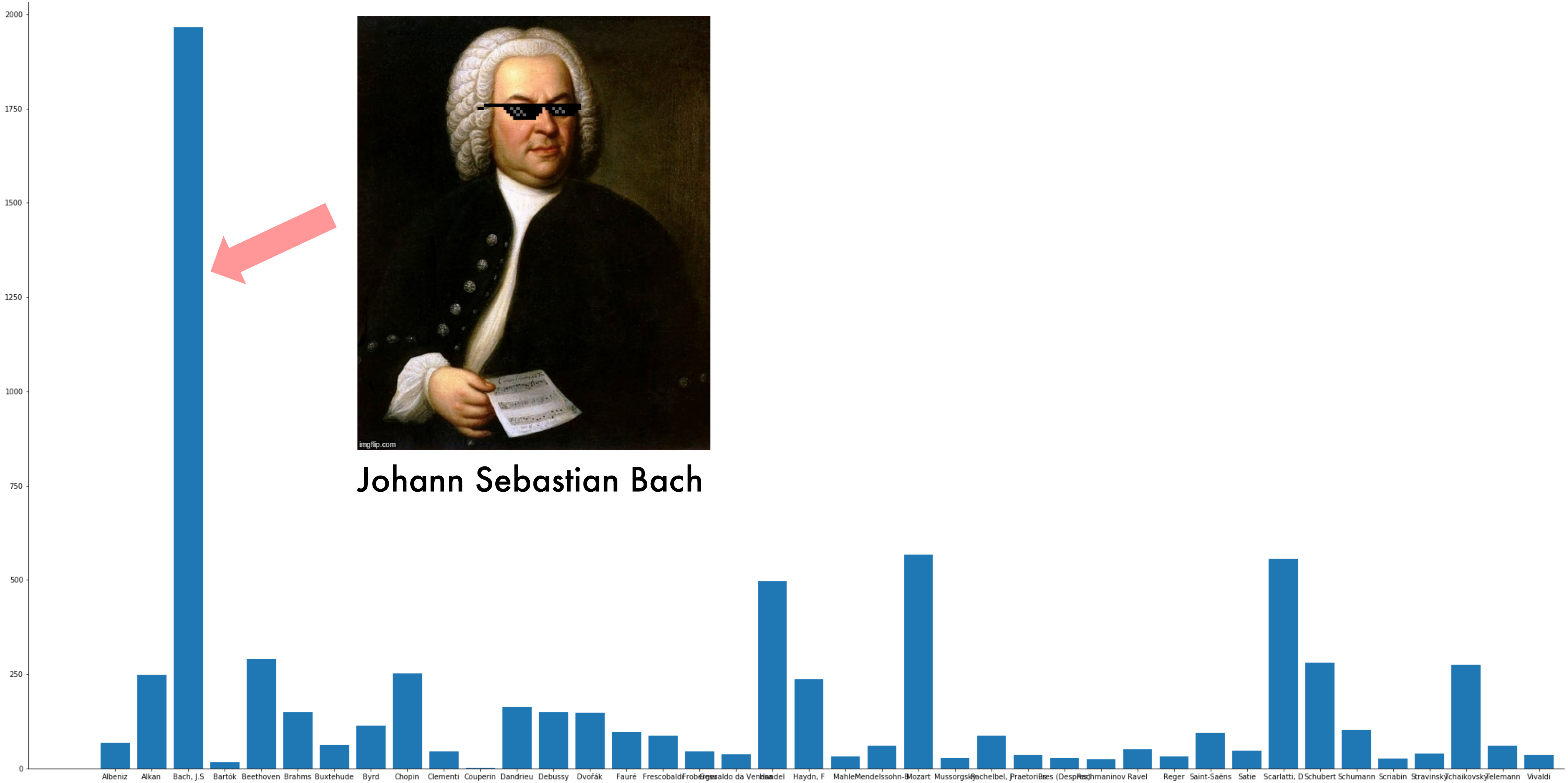
_name: link to music archive

_composer: surname of a composer

_piece: sparse matrix with stored elements in COOrdinate format

- 40 Composers

Pieces Distribution



Johann Sebastian Bach

Data Representation

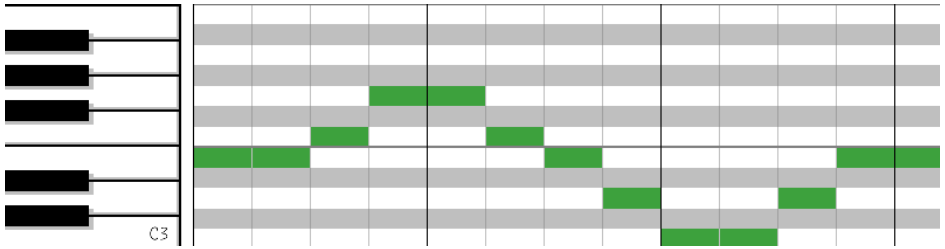
MIDI Format

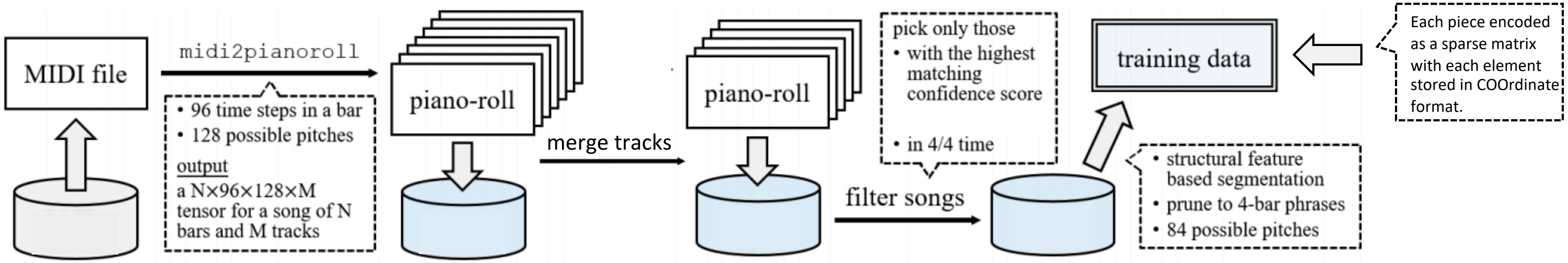
Features: Note on, Note off, Relative metrical time, Absolute time.
Drawback: It does not effectively preserve the notion of multiple notes.

```
2,  96, Note_on,  0, 60, 90
2, 192, Note_off, 0, 60,  0
2, 192, Note_on,  0, 62, 90
2, 288, Note_off, 0, 62,  0
2, 288, Note_on,  0, 64, 90
2, 384, Note_off, 0, 64,  0
```

Piano Roll

Features: Note Control Information, Length, Localization.
Drawback: There is no note off information.





Details

Characteristics

- Temporal structure is critical.
- Pieces of variable length.
- Total number of timesteps is more than 70M

Shortcomings

- The lack of musical info as note values, key signature and tempo.
- Most important: no info on voicing and relating sounds or multiple instruments and tracks.

Goals

1. Classification Task

- ✓ Classify authors by extracted sample from a piece with two different approaches.

2. Generation Task

- ✓ Learn musical content and generate new musical pieces based on this.
- ✓ Find an approximation of the distribution of these pieces.

Data Pre-Processing

Data Preprocessing

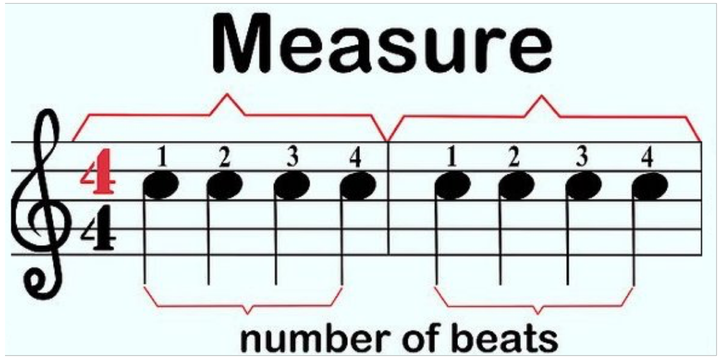
Each matrix has been normalised, so to take intensity/loudness between 0 and 1.

Since not all the available notes are played by the dataset pieces, the notes range have been shrunk to 70.

Data has been randomly split in train (80%), test (20%) and validation set.

Data Slicing

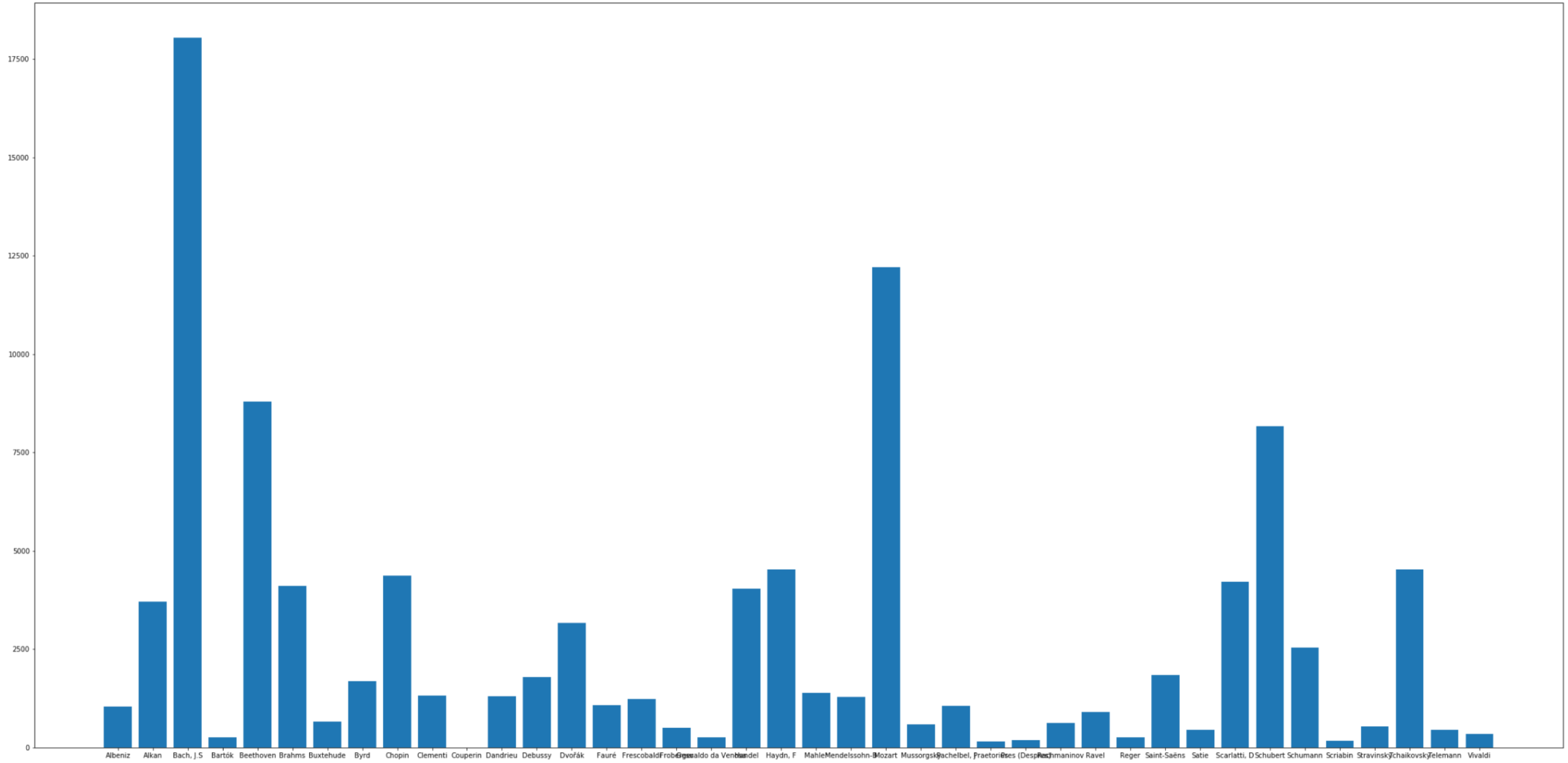
Hypothesis: time signature is 4/4 → 4 beats X measure



24 time steps correspond to one beat → 96 time steps X measure

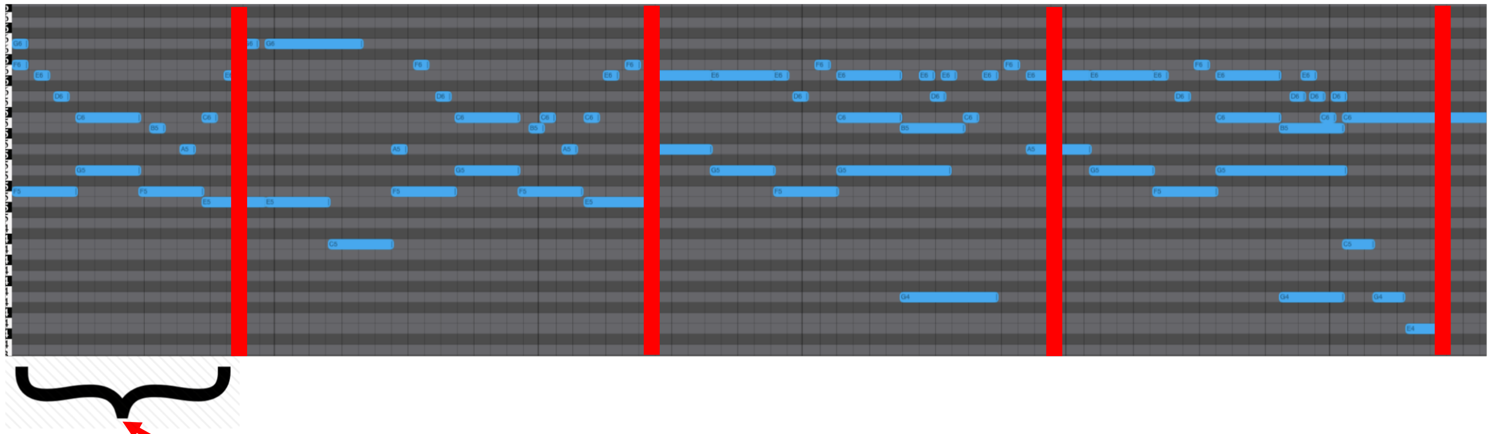
Our choice is to slice the pieces in chunks of 8 measures → each slice consists of 768 time steps.

Data Slicing



Data Augmentation

We augmented the dataset exploiting (randomly) shifted slices.



Random initial shift

Now it’s possible to sample any possible slice.

(Additionally, a vertical shift could aim at tonality invariance)

Classification

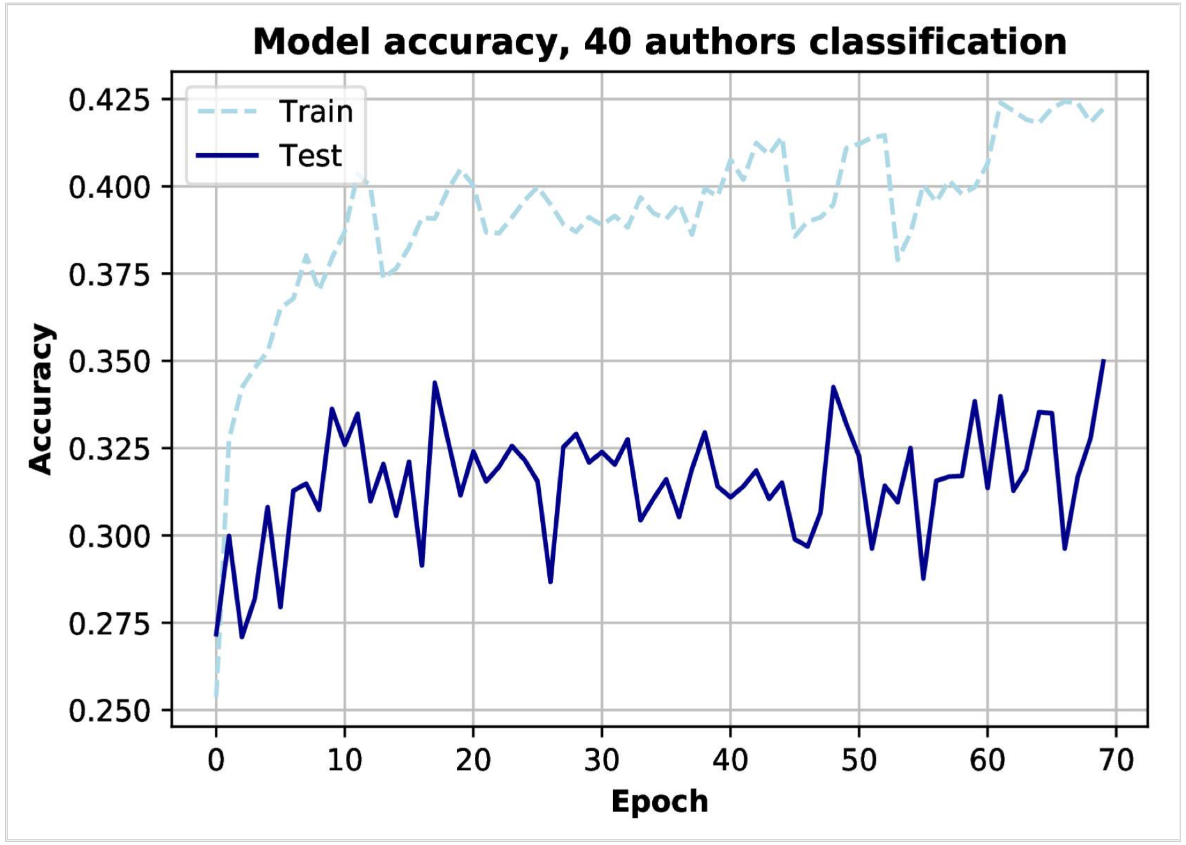
Classification

Aim: assign a given slice to one of the 40 authors.

- 80% training set
- 20% test set

Common approaches include RNN and CNN.

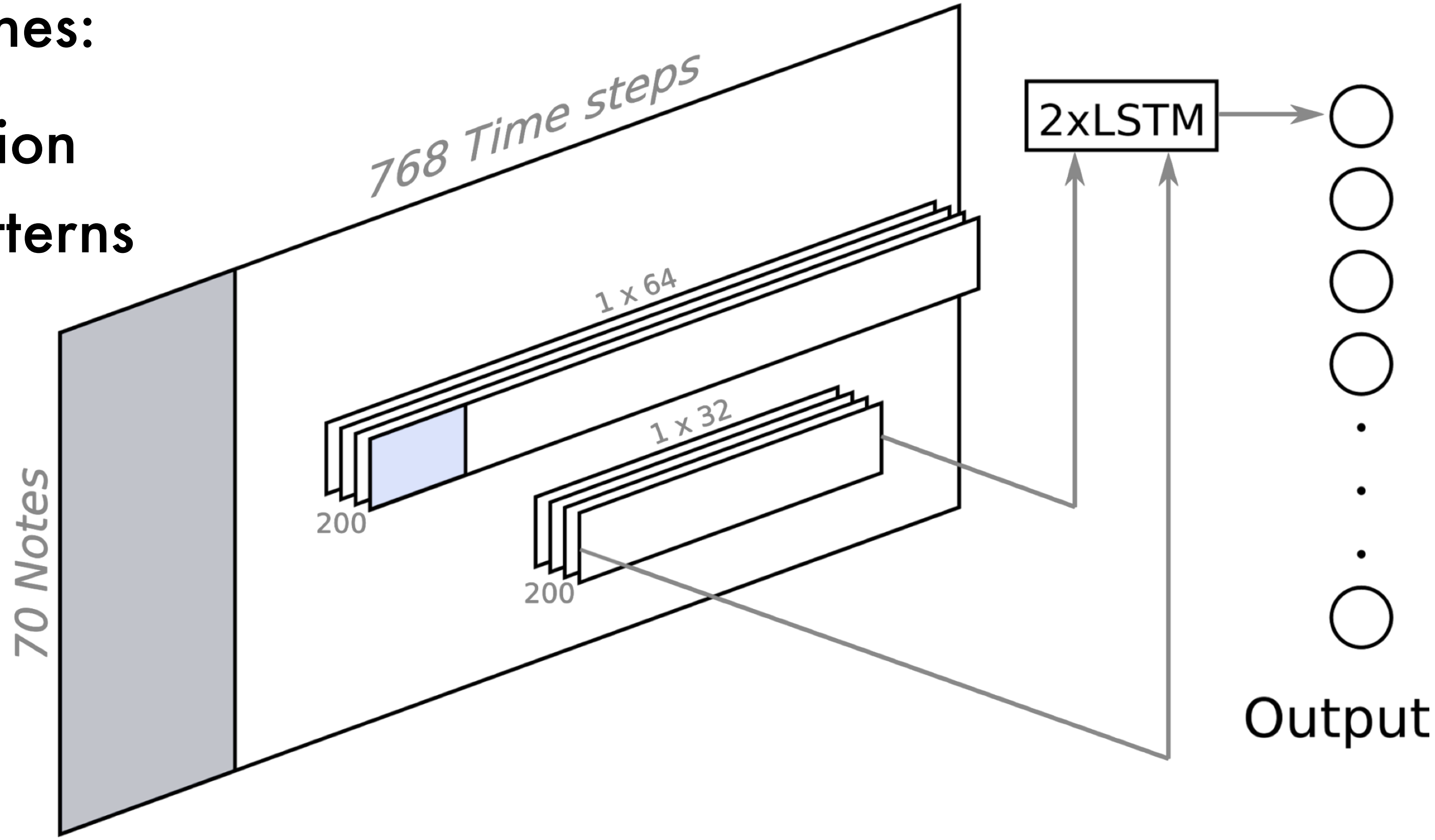
CNN classification:



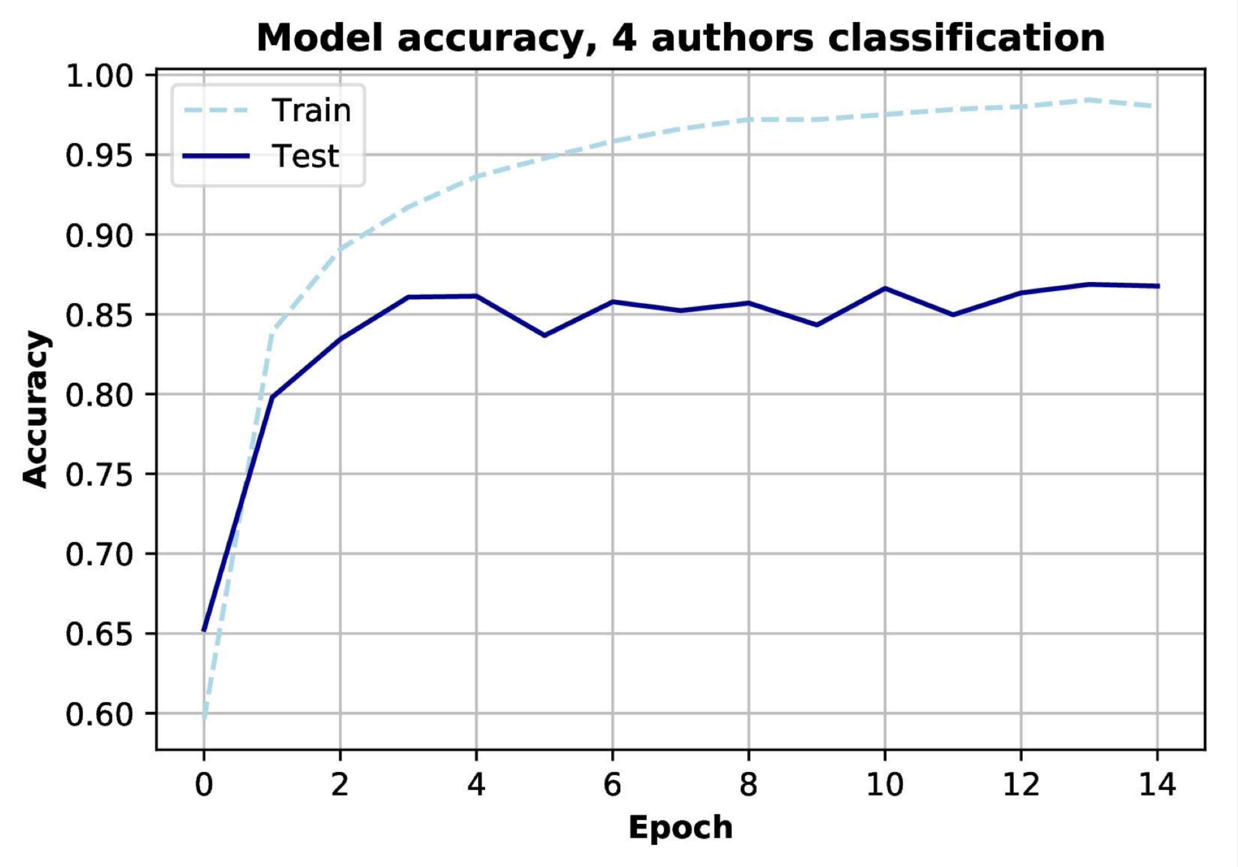
CRNN

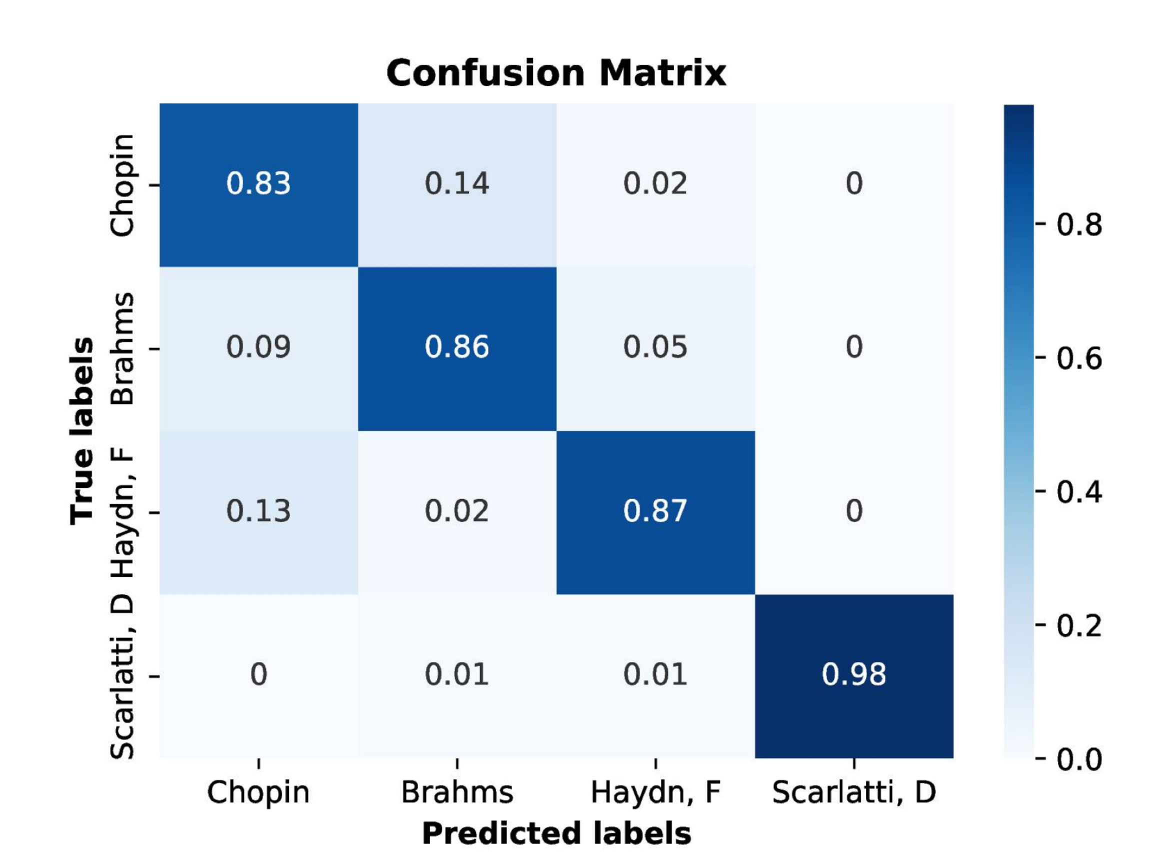
Combine the previous approaches:

- CNN for local feature extraction
- RNN (LSTM) for temporal patterns

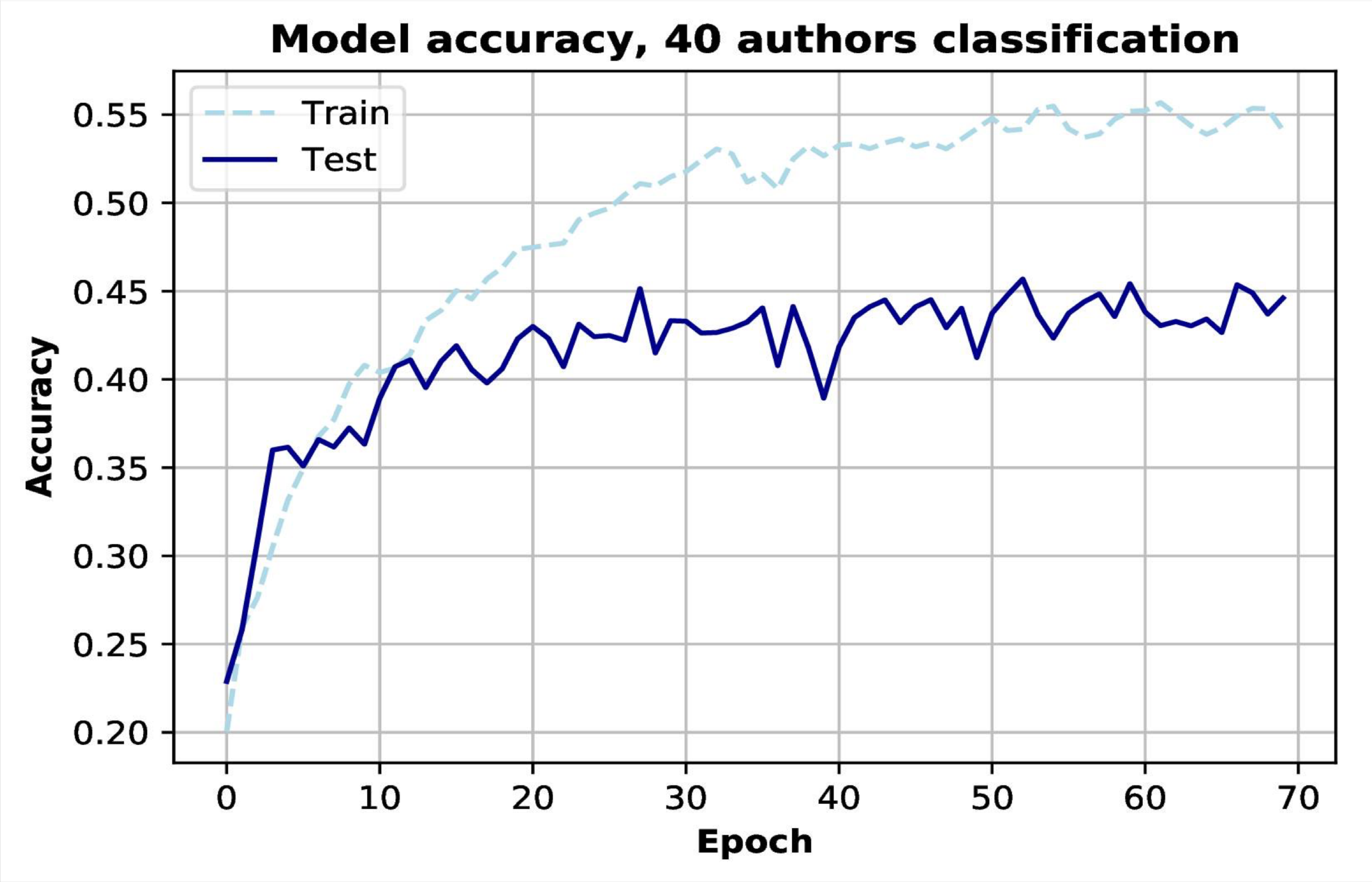


CRNN performance on balanced 4 authors dataset





CRNN performance on whole dataset

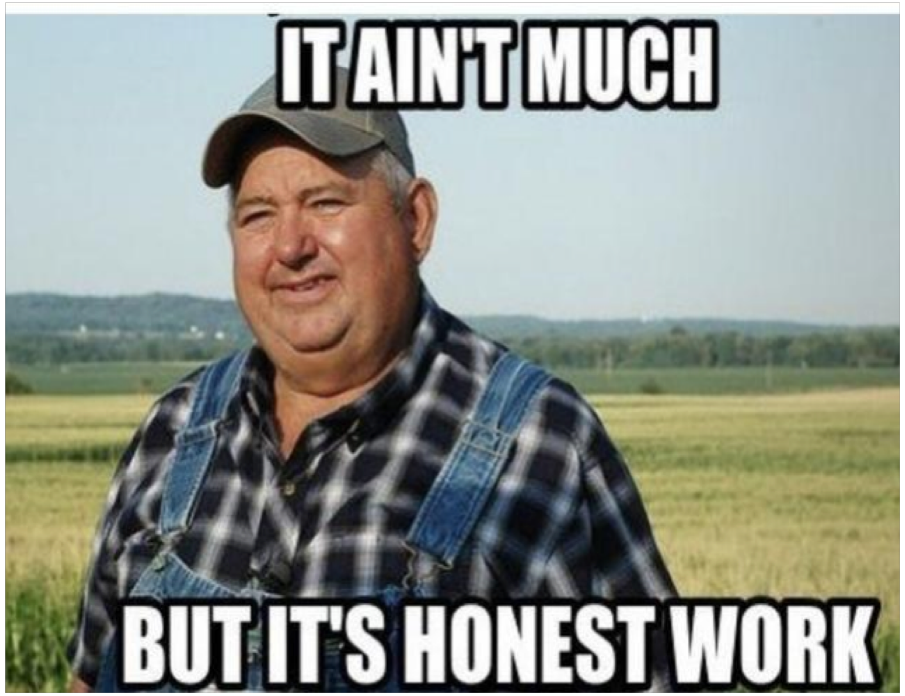


Comments of performance

Dataset	Test accuracy
2 authors, balanced	96%
4 authors, balanced	88%
40 authors	47%

Comments of performance

Dataset	Test accuracy
2 authors, balanced	96%
4 authors, balanced	88%
40 authors	47%



Generation

Generation

- Produce new musical pieces that are similar (but not the same) to the ones from the dataset.
- This means find an approximation of the distribution of these pieces.

It is necessary to reduce the dimensionality of our data (768x70 is too much) such that it will be easier to sample points that are near to the dataset ones.

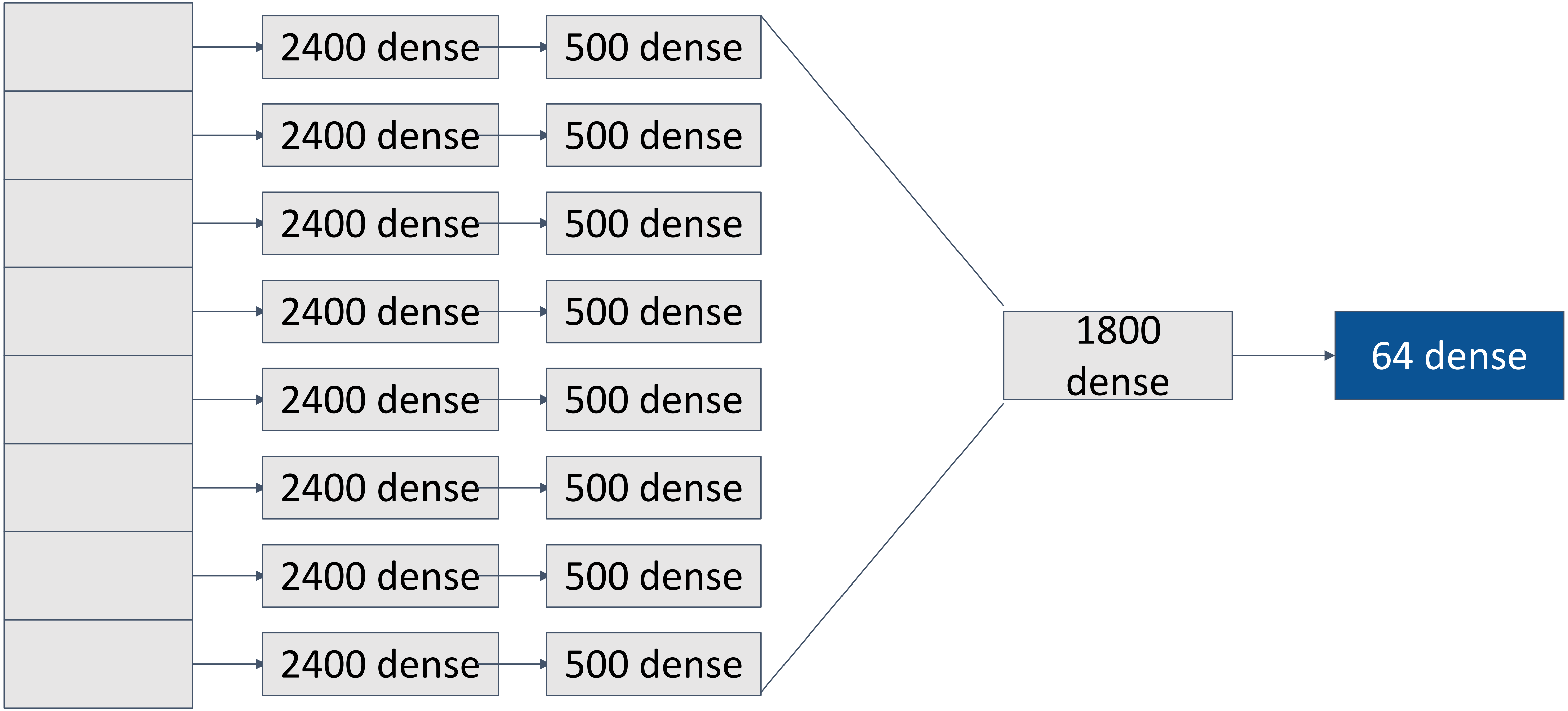
Train an **autoencoder**:

- Encoder: used to know (roughly) how latent features are distributed
- Decoder: used to generate pieces from chosen latent features

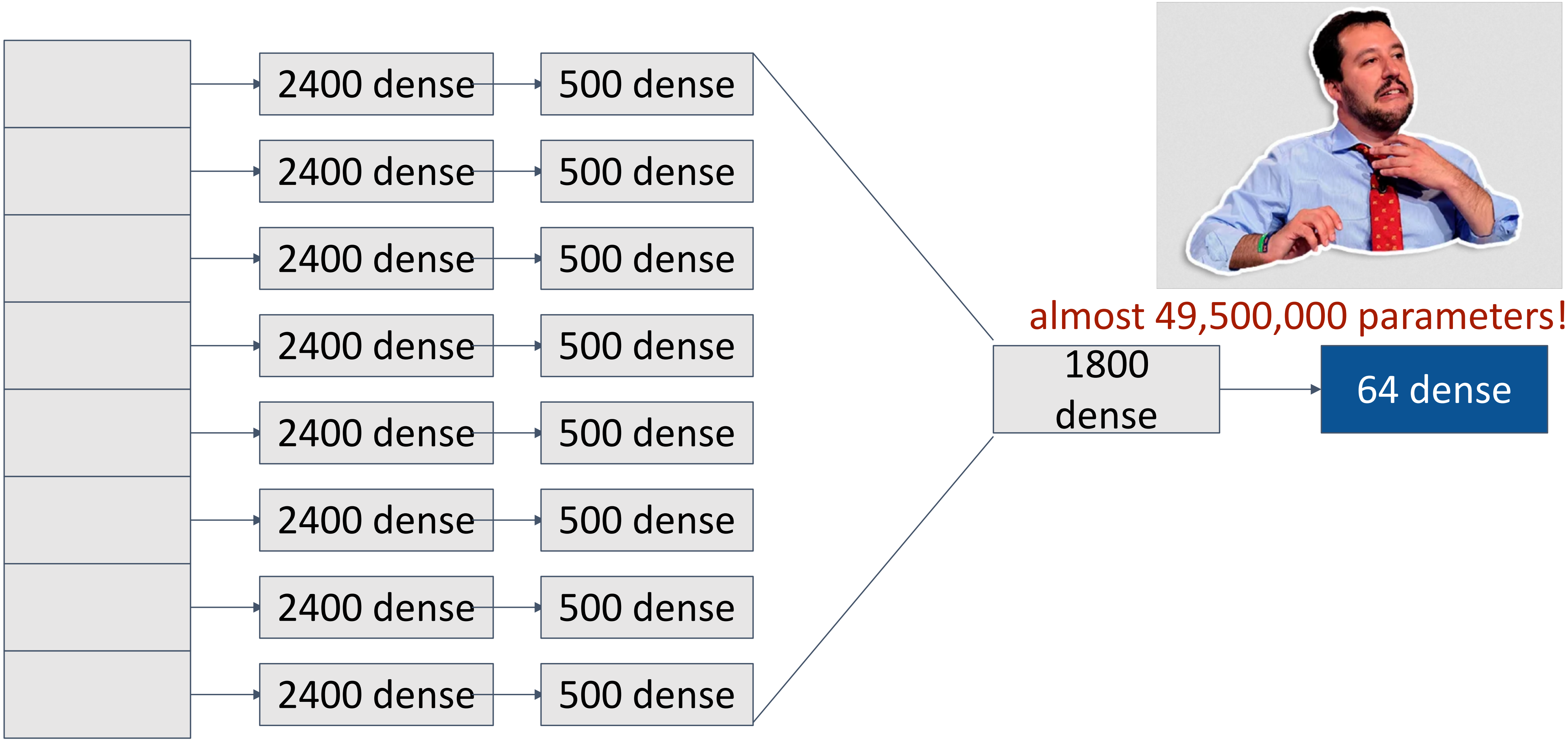
Details

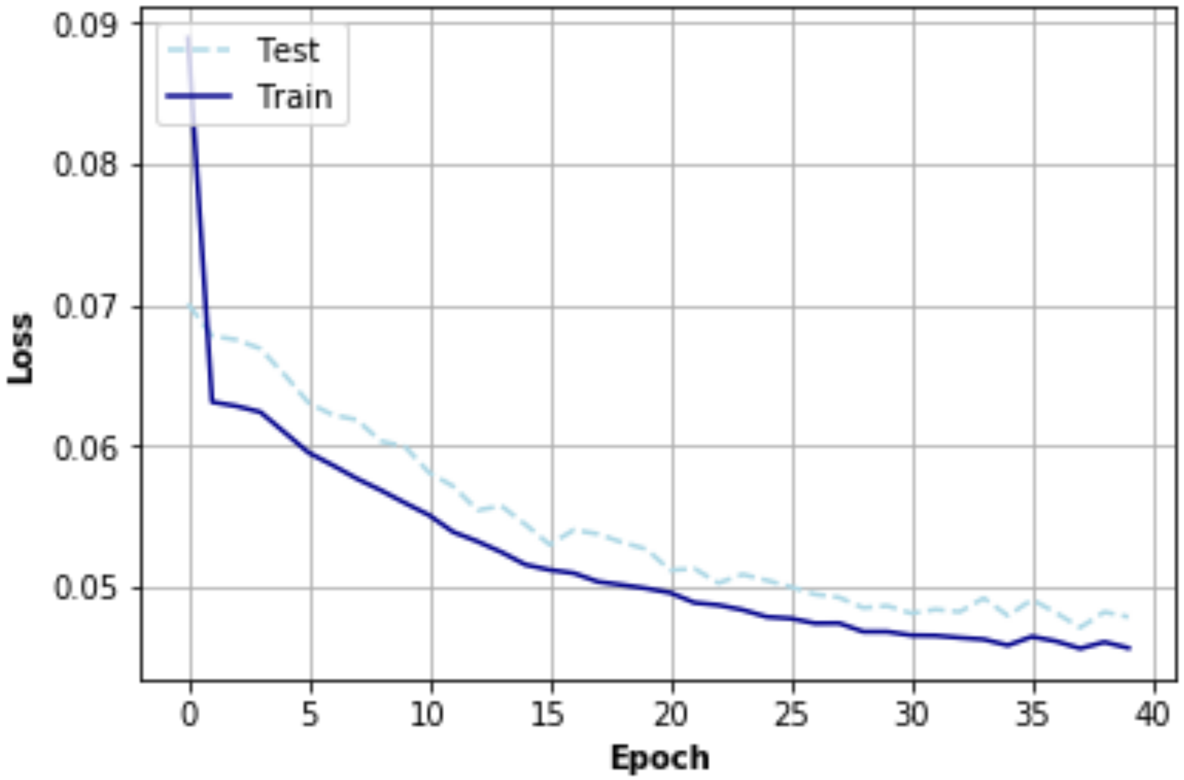
- We choose to generate 8 bars musical fragments (768x70).
- How many latent features?
  Trade-off between quality of reconstruction and dimensionality reduction.
  We experimented with 64 and 128 latent features.
- I know, you're thinking about VAEs, but we experimented an alternative solution.

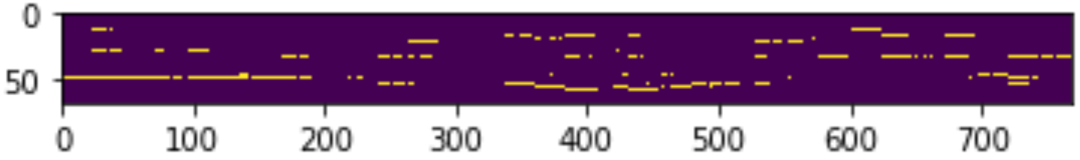
Autoencoder Architecture



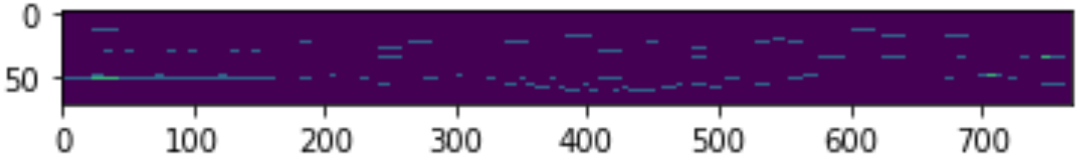
Autoencoder Architecture







Reconstructed



Original

Manupulating Hidden Variables

We would like to have the pieces encoding in this reduced features space distributed in the following way:

- centered around 0 (mean 0)
- features uncorrelated between each other
- same variance for each feature

even better would be: $Z \sim \mathcal{N}(\mu = 0,\ \Sigma = I)$

The distribution was not like this.

Partial Solution: PCA + Standardization

Generation Process

Preprocessing Phase

Original Pieces

Encoded Pieces

Computing Principal Components

Computing PCs scores standard deviations

Generation

Input Gaussian Noise

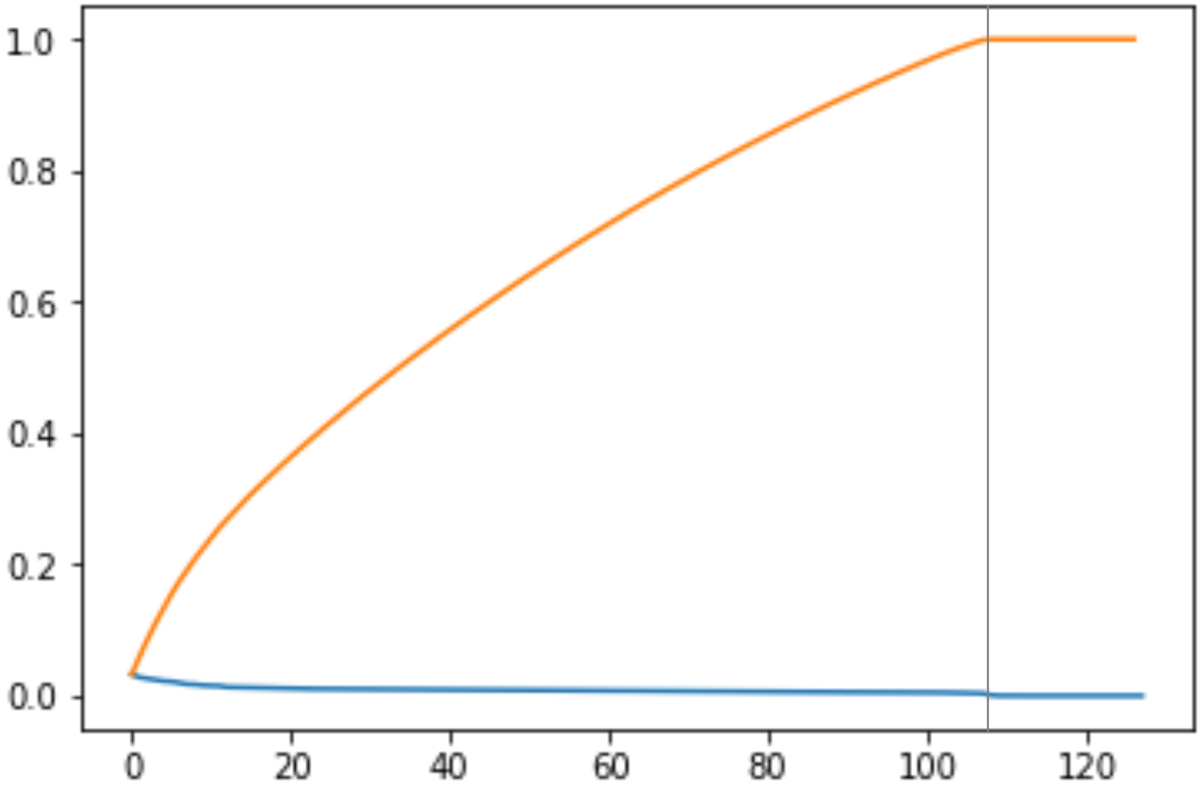
Multiply Components by PC std

Compute inverse PC

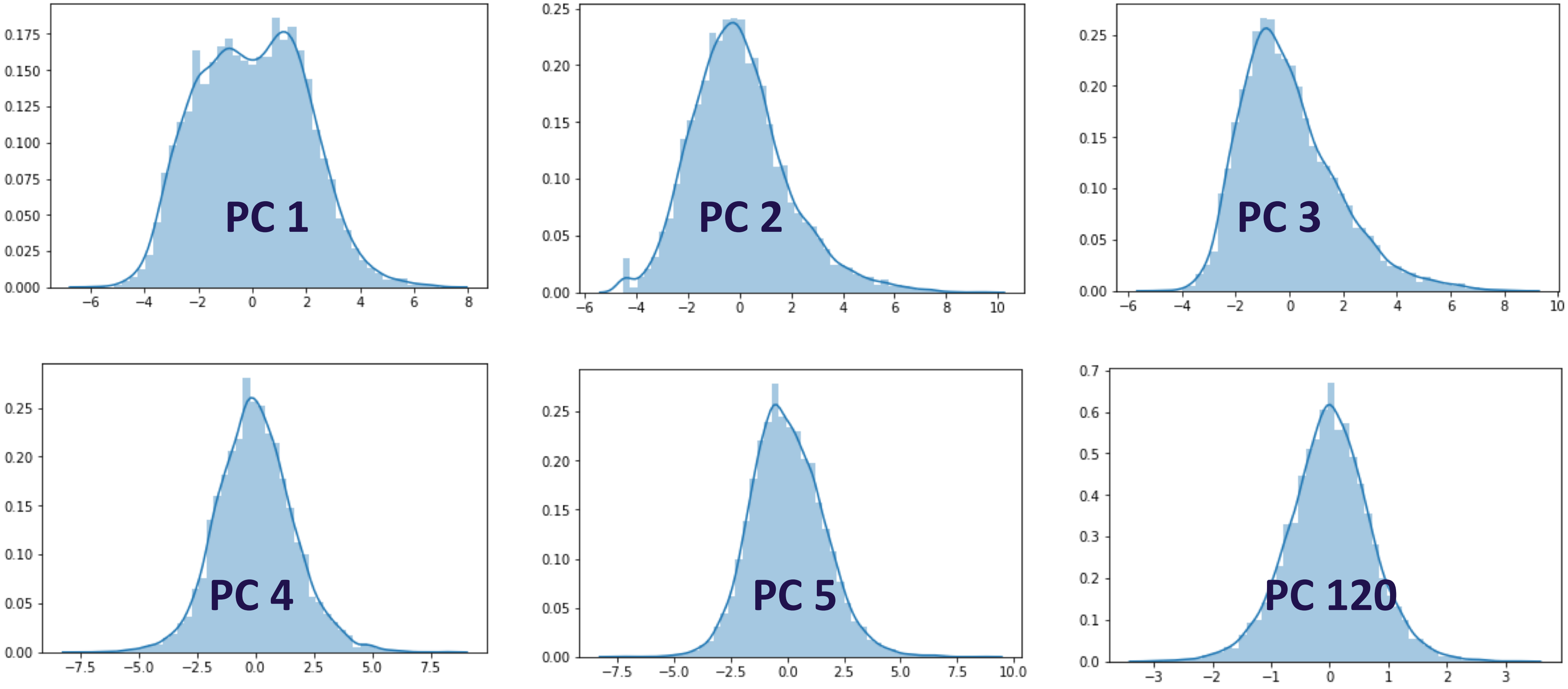
Decode

New Random Generated Piece

Cumulative explained variance by the PCs (Bach), using 128 latent features.

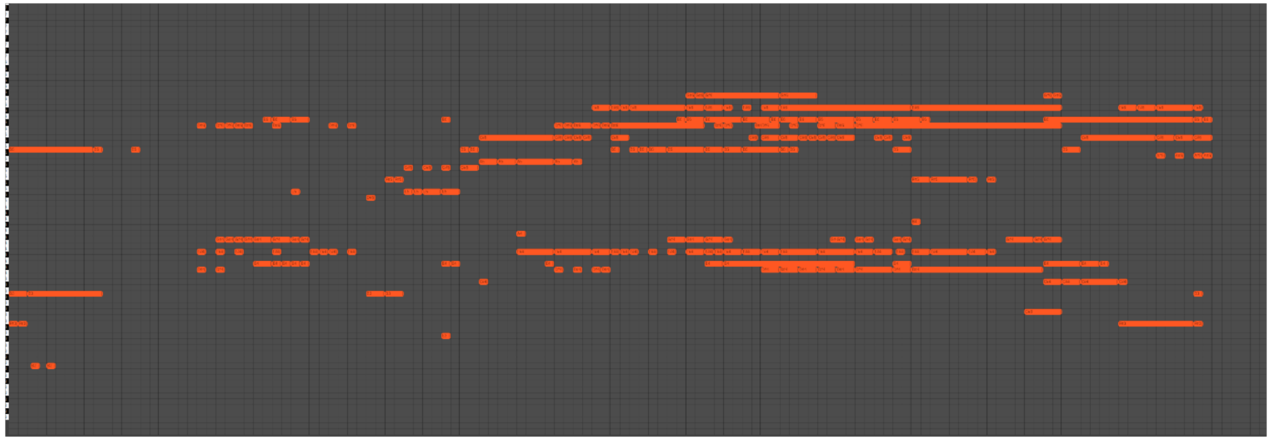


Each PC distribution is almost normal
(this doesn't imply $Z \sim \mathcal{N}(\mu = 0,\ \Sigma = I)$)



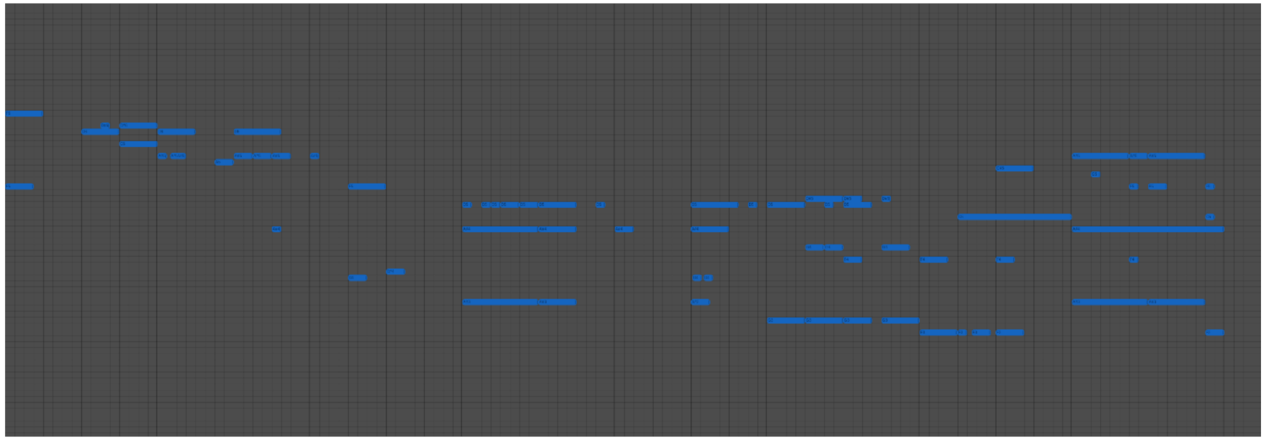
Results: Something that resembles musical fragments.

These pieces were generated with random gaussian noise features.



TRAINING ON BACH PIECES.

TRAINING ON BEETHOVEN PIECES.

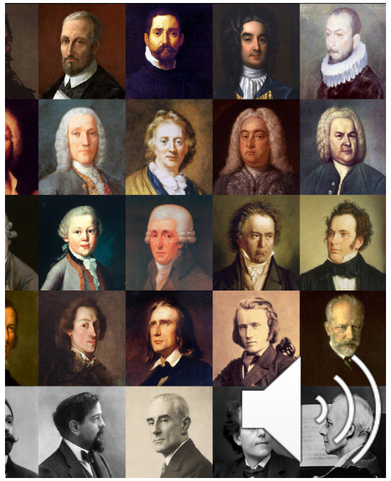


Results

Bach

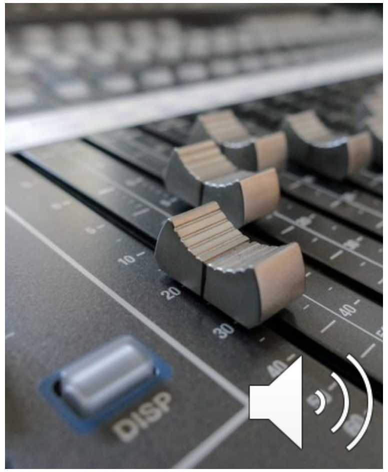
Bach & Scarlatti

Beethoven



Total

Mozart (& Scassola)



Mix

Conclusions

Conclusions

Best classification result on Kaggle...

Conclusions

Best classification result on Kaggle...

...Since nobody has ever tried to solve this task.

References

- Choi, Keunwoo, et al. "Convolutional recurrent neural networks for music classification." 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2017.
- Jahya Burke, et al. "Music Composition using an Autoencoder Network". Department of Electrical and Computer Engineering, University of California, San Diego, 2019.
- Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang and Yi-Hsuan Yang, "MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment.", AAAI Conference on Artificial Intelligence 2018.

Thank you!