# Musical Composer Identification

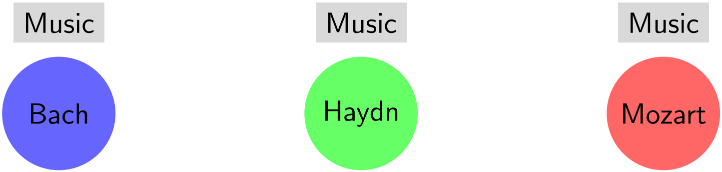
Niko Abeler

June 22, 2015

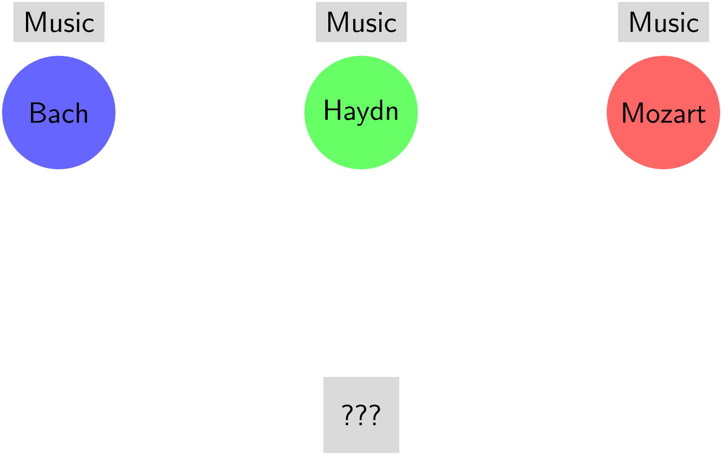
# The Problem

# The Problem

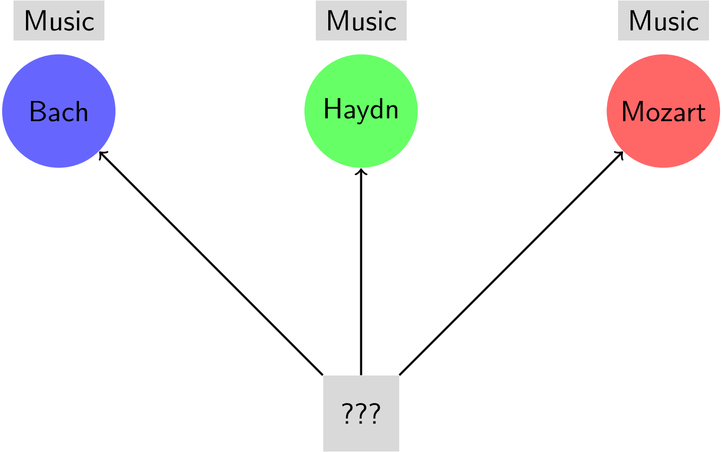
# The Problem



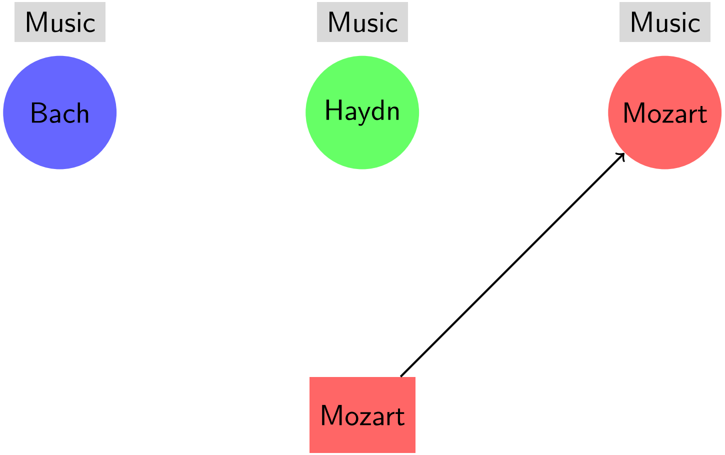
# The Problem



# The Problem



# The Problem



# History

# History

- First system "as good as experts"
- 2001, Pollastri and Simoncelli[5]
- Hidden Markov Models for every composer
- Short sequences of relative pitch changes
- 42% Success Rate (Experts: 48%)

# History

- Using features extracted from music sheets[2]
- Assumption: Every composer has a unique note distribution
- Classical Machine Learning techniques:
  - Support Vector Machine
  - Naive Bayes
- $\sim$ 50% Accuracy

# History

- More sophisticated features
- Mostly extracted from music sheets
- Training of models:
  - Neural Networks[1 3]
  - Markov Chains[4]
  - n-grams
- $\sim$ 60-80% Accuracy

# Identifiers

# Pitch Class Profile

- Summation over notes

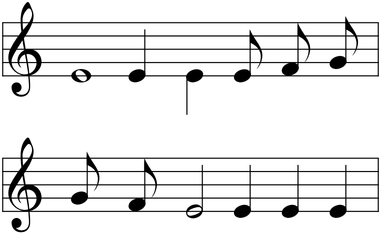
$$CP(n \mod (12) + 1) = \sum_{n \mod (12) \in M}^{1} 1$$

- Normalization as pieces have different lengths

$$N(i) = \frac{CP(i)}{\max_{1 \leq j \leq 12} CP(j)}$$

# Pitch Class Profile

- Ignores the key of the musical pieces
- Ignores tempo / note length
- Different use cases:
  - single global descriptor
  - fixed-length time windows
  - sliding windows
- several variations



Both have the same PCP: $\begin{pmatrix} \vdots \\ 4 \\ 1 \\ 1 \\ \vdots \end{pmatrix}$

# Mel Frequency Cepstral Coefficients (MFCC)

- Commonly used in speech recognition
- Robust against noise
- Takes human perception into account

# Mel Frequency Cepstral Coefficients (MFCC)

- Compute the Fourier Transform of the audio signal
- Map the powers of Fourier Transform onto the Mel Scale and take the log

$$m = log(2595 \log_{10}(1 + \frac{f}{700}))$$

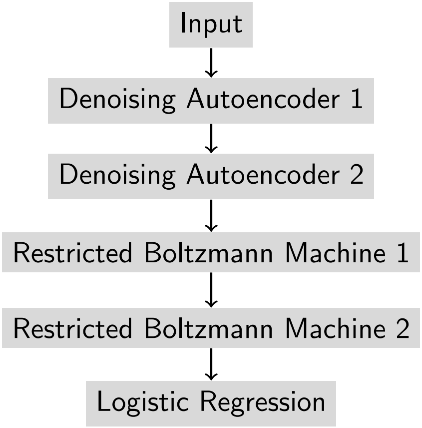
- Compute Discrete Cosine Transform (DCT) of the mel log powers
- The MFCCs are the amplitudes of the DCT
- Variant Mel-Phon Coefficient (MPC) does not apply DCT

# Deep Neural Network

# Audio Classical Composer Identification by Deep Neural Network

- by Zhen Hu, Kun Fu and Changshui Zhang (2013)[1]
- Construction of a 5-layer Deep Neural Network
- Works with audio clips
  - no prior cleaning/preparation process
- High success rates

# DNN Overview



# Input

- Recordings of different performances
- no prior denoising or selection
- separated into 30 second clips
- Per clip:
    - separation into 3 second fragments with 50% overlap
    - Mel-Phon Coefficient (MPC) for every fragment
- 592 MPC as input for the Deep Neural Network

# Denoising Autoencoder (DA)

- Encoder maps input $x \in [0,1]^n$ onto representation $y \in [0,1]^m$

$$y = sigm(W \cdot x)$$

- Decoder maps $y$ back onto $z \in [0,1]^n$

$$z = sigm(W^T \cdot y)$$

# DA Training

- The DA is trained to minimize the error

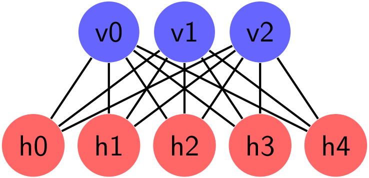
$$\min_{W} ||sigm(W^T \cdot sigm(W \cdot x')) - x||_2$$

- $x'$ is a corrupted version of x
- The DA learns something underneath the original data

# Stacked DA

- Multiple Denoising Autoencoders ( here 2 )
- Trained in two steps:
- Train every layer separately
  - Input: output of previous layer
  - Target: clean data
- Fine-Tuning of the whole stack

# Restricted Boltzmann Machine (RBM)




- energy-based model

$$P(v, h) = \frac{1}{Z} \exp(h^T W v + b^T v + c^T h)$$

- (W: weights ; b,c: offsets ; h: hidden nodes ; v: visible nodes)

# RBM Training

- maximize probability of training input V

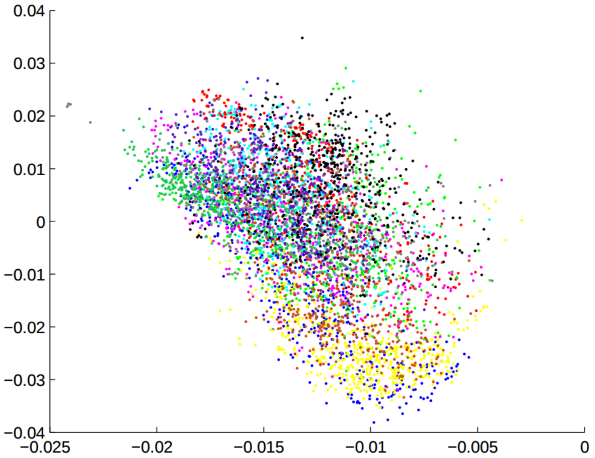
$$\max_{W,b,c} \log P(V) = \max_{W,b,c} \sum_{v \in V} \log P(v)$$

- iterative learning algorithm
  - Gibbs sampling
  - Contrastive Divergence
  - Back Propagation

# Logistic Regression

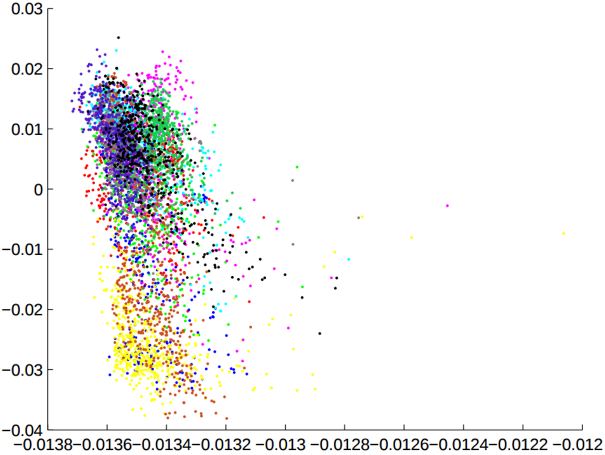
$$P(Y) = \frac{1}{1 + \exp(-(\beta_0 + x_i^T \beta))}$$

- Scalar $\beta$ and offset $\beta_0$
- Assigns a confidence value to every composer
- Composer with highest confidence is chosen



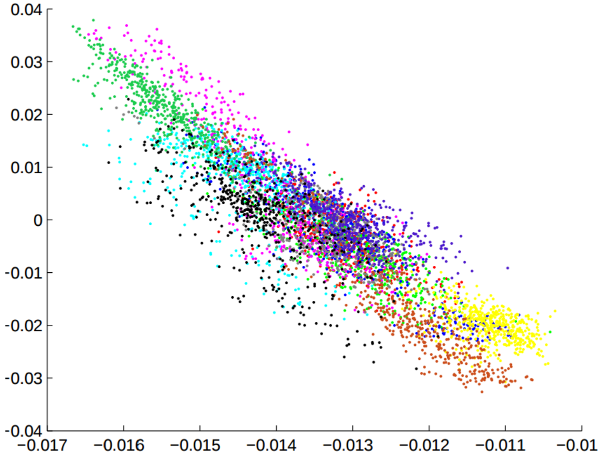

(a) 2-D reduction for the testing data.

1



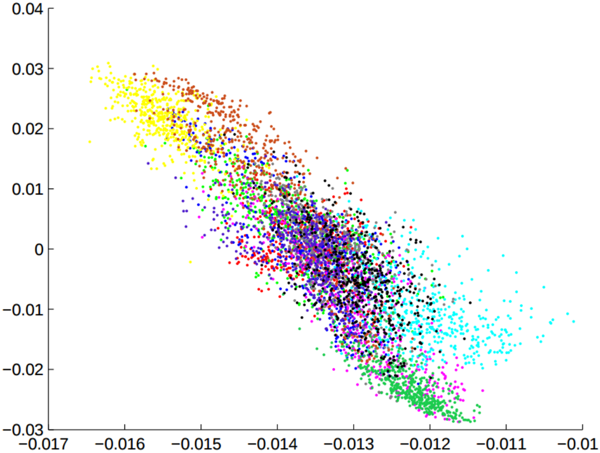

(b) 2-D reduction for the output of layer 1.

1

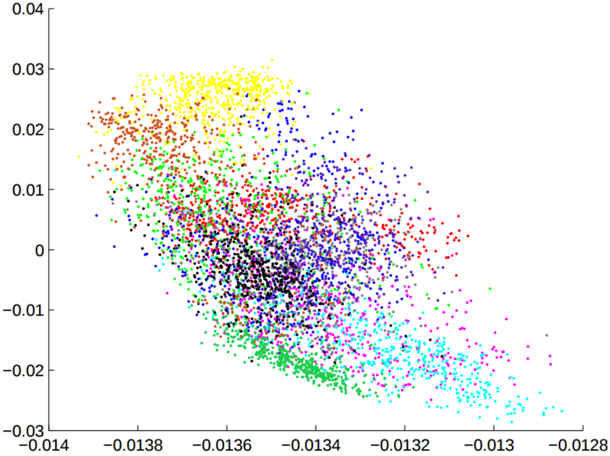



(c) 2-D reduction for the output of layer 2.

1




(d) 2-D reduction for the output of layer 3.




(e) 2-D reduction for the output of layer 4.

# Accuracy

# Test Data

- 11 composers of classical music
- 360 clips à 30 seconds
  - 250 Training
  - 50 Validation
  - 60 Testing
- Clip of musical piece in training set $\Leftrightarrow$ no clip of musical piece in test set

# Accuracy

- Total accuracy: 76.26%[1]

| Composer | Accuracy | Composer | Accuracy |
|---|---|---|---|
| Bach | 93.10% | Haydn | 40.00% |
| Beethoven | 63.33% | Mendelssohn | 100.00% |
| Brahms | 75.51% | Mozart | 74.58% |
| Chopin | 98.11% | Schubert | 20.59% |
| Dvorak | 97.01% | Vivaldi | 87.04% |
| Handel | 100.00% | | |

# Conclusion

# Conclusion

- Musical Composer Identification is solvable
- Error Rates still high
- Final evaluation of an expert still needed
- Some Pairings still hard to distinguish
- Only few methods working with audio files instead of music sheets

# References I

[1] Hu, Z., Fu, K., and Zhang, C. (2013). Audio artist identification by deep neural network. *CoRR*, abs/1301.3195.

[2] Justin Lebar, Gary Chang, D. Y. (2008). Classifying musical scores by composer: A machine learning approach.

[3] Kaliakatsos-papakostas, M. A., Epitropakis, M. G., and Vrahatis, M. N. (2010). Musical composer identification through probabilistic and feedforward neural networks.

[4] Kaliakatsos-Papakostas, M. A., Epitropakis, M. G., and Vrahatis, M. N. (2011). Weighted markov chain model for musical composer identification. In *Applications of Evolutionary Computation*, pages 334–343. Springer.

[5] Pollastri, E. and Simoncelli, G. (2001). Classification of melodies by composer with hidden markov models. In *Web Delivering of Music, 2001. Proceedings. First International Conference on*, pages 88–95. IEEE.