# AI Applications in Music

**CSE 352**
Anita Wasilewska
Fall 2017

**Team 2**
Sai Ande
Timothy Hart
Bryan Koelbel
William Schweigert
Rubin Thomas

# Works Cited

- Dieleman, Sander. "Keynote." *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems - DLRS 2016* (2016): n. pag. Web.
- "Memetic Music Composition." *IEEE Transactions on Evolutionary Computation, Evolutionary Computation, IEEE Transactions On, IEEE Trans. Evol. Computat*, no. 1, 2016, p. 1. EBSCO*host*, doi:10.1109/TEVC.2014.2366871. <http://proxy.library.stonybrook.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.6945353&site=eds-live&scope=site>

- "Recommending Music on Spotify with Deep Learning." *Sander Dieleman*. N.p., 5 Aug. 2014. Web.
- Van Der Maaten, Laurens. "T-SNE." *Laurens Van Der Maaten*. N.p., 2017. Web.
- Liebman, Elad, et al. “Peter Stone's Selected Publications.” *Peter Stone: DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation*, Peter Stone's Selected Publications, May 2015, www.cs.utexas.edu/users/pstone/Papers/bib2html/b2hd-AAMAS2015-eladlieb.html.
- Balzano, Gerald J. “The American Journal of Psychology.” *The American Journal of Psychology*, vol. 100, no. 1, 1987, pp. 135–139. *JSTOR*, JSTOR, www.jstor.org/stable/1422649.
- Franklin, Judy A. "Recurrent Neural Networks and Pitch Representations for Music Tasks." *FLAIRS Conference*. 2004

## Outline

- Overview of AI in Music
- Music Generation Overview
- Music Generation - Memetic Music Composition
- Music Recommendation -  How Spotify Makes Music Recommendations
- Music Recommendation - DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation

## Overview

## What is Music?

- Music is a combination of sounds which when arranged correctly can be pleasing to listen to.
- Music is subjective. Not everyone has the same tastes in music.
- Music is expressive. Many artists compose music to express a feeling or to make a commentary on something.

## How can AI be applied to music?

- Music Curation (playlists, suggested songs)
- Music Generation (AI composes music)
- Music media generation (Music videos,etc.)

## Who's Using it?

- Sony - Flow Machines
  - Uses AI for music composition and generation
  - https://youtu.be/LSHZ_b05W7o
- Spotify
  - Using AI to suggest new songs and create playlists for users
- Intel
  - Using machine learning to assist in the creation of a music video for chinese pop star Chris Lee

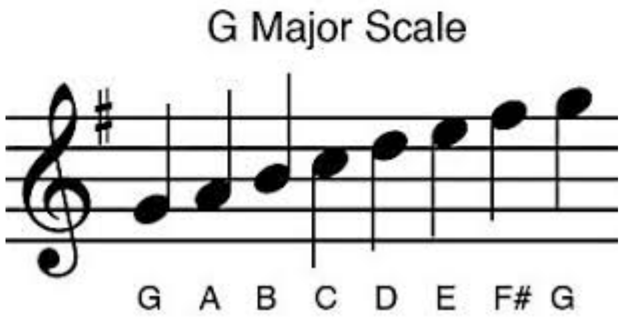
## Where is it going?

- Sony is planning on releasing a full album created by AI later this year.
- JukeDeck a startup monetizing custom song creation for $0.99 a user inputs the mood and instrument family and a song is generated in a few seconds.

## Music Generation Overview

## Music Theory

- Music theory states that music generally follows a certain set of rules.
    - E.g. Notes in the key of G Major

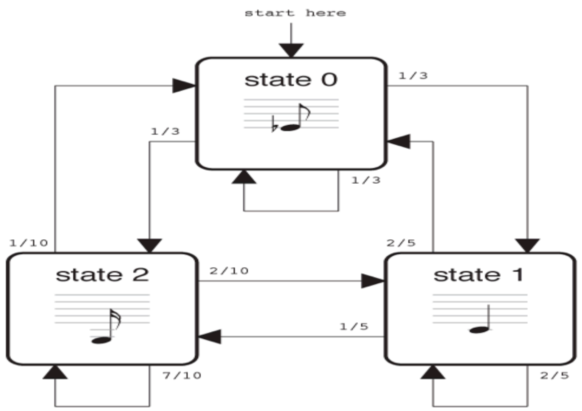


## Theory Driven Approach

- In 1955, Hiller and Isaacson programmed the Illiac Computer to write a musical composition.

- Random generator tested against a markov chain built from music theory conventions.
    - For example if a selected note is a "C" then next Available notes may be "G" (65%), "E" (15%), "F"(10%), or A (5%)

- Melodic but not Creative



## Questions:

- How do we make the computer as expressive and creative as a Human?

- How do humans acquire expression and creativity through music?

## Questions:

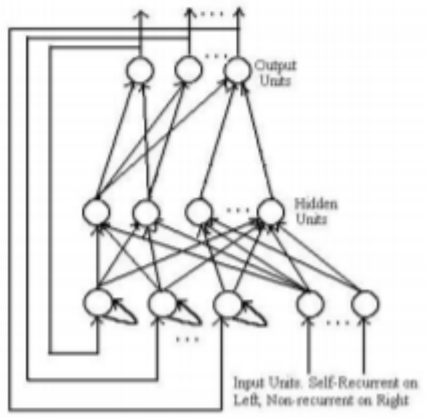
- How do musicians create music?

  While there is more than one school of thought, the general consensus is humans increase musical cognition by listening to music. (Dowling and Harwood 1986)

  Music is created by a process of observation, combination, and imitation.

# Data Driven Approach

- J. Franklin (2001) used a *recurrent* neural network generate sax solo’s
- Inputs are binary representations of chromatic notes e.g.
    - C is represented as 100000000000. C# is 010000000000, D is 001000000000.
- Input: Transcripts from Sonny Rollins.
- Neural network rewards based on jazz harmony Criteria and style, recursion allows for continued creativity.



Source: Judy A. Franklin, "Recurrent Neural Networks and Pitch Representations for Music Tasks"

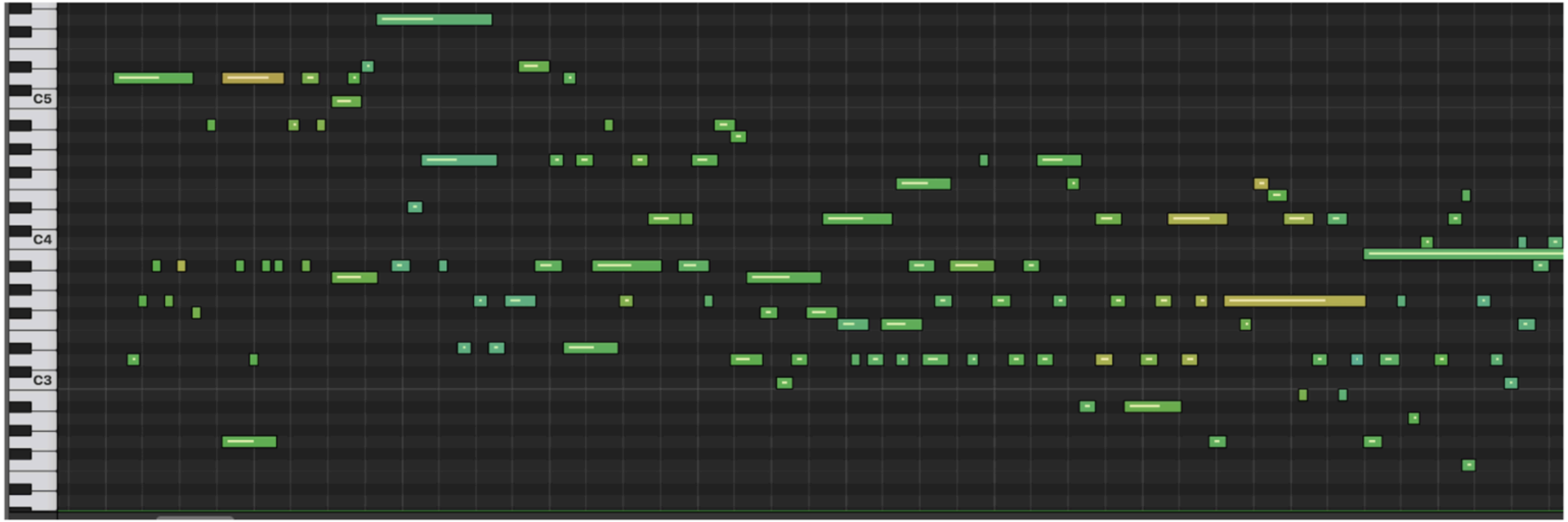
## Problem

- The problem with this approach - Missing many musical attributes such as
    - dynamics (loudness),
    - rubato (variations in note duration and attack time),
    - vibrato (repeated small variation in pitch frequency and amplitude),
    - articulation (variations in the transition time between notes)

Example: [https://magenta.tensorflow.org/performance-rnn](https://magenta.tensorflow.org/performance-rnn)

## Google’s Project Magenta

- Google has one solution for the expressive timing and dynamics problem.
- Dataset - “Yamaha e-Piano Competition”
  - MIDI (Musical Instrument Digital Interface)
- Done by creating a much more advanced neural network that can read events such as:
  - Note-ON
  - Note-OFF
  - Velocity of KeyPress



MIDI stream: RNN training set (Yahama data)

## Music Generation

Memetic Music Composition

"Memetic Music Composition." *IEEE Transactions on Evolutionary Computation, Evolutionary Computation, IEEE Transactions On, IEEE Trans. Evol. Computat*, no. 1, 2016, p. 1. EBSCO*host*, doi: 10.1109/TEVC.2014.2366871. <http://proxy.library.stonybrook.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.6945353&site=eds-live&scope=site>

## Memetic Music Composition

- Enrique Muñoz, Jose Manuel Cadenas, Yew Soon Ong, Giovanni Acampora, 2016
- **Challenges:**
    - Replicating skill of human composer
    - Defining a measure for the quality of generated music
- **Goals:**
    - Use unfigured bass technique
        - Human provides bass melody, machine improvises chords and harmony

## The Algorithm

- Take in bass line
- Figuration step: decide which chords produce the best melody from a given bass line
- Pairs of chords given weight based on "pleasantness"
- Reducing solution search space from all chords to a subset of chords
- NP-hard problem

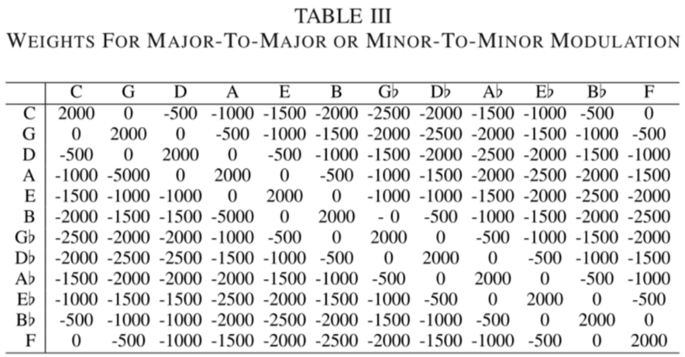


TABLE III
WEIGHTS FOR MAJOR-TO-MAJOR OR MINOR-TO-MINOR MODULATION

| | C | G | D | A | E | B | G♭ | D♭ | A♭ | E♭ | B♭ | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | 2000 | 0 | -500 | -1000 | -1500 | -2000 | -2500 | -2000 | -1500 | -1000 | -500 | 0 |
| G | 0 | 2000 | 0 | -500 | -1000 | -1500 | -2000 | -2500 | -2000 | -1500 | -1000 | -500 |
| D | -500 | 0 | 2000 | 0 | -500 | -1000 | -1500 | -2000 | -2500 | -2000 | -1500 | -1000 |
| A | -1000 | -5000 | 0 | 2000 | 0 | -500 | -1000 | -1500 | -2000 | -2500 | -2000 | -1500 |
| E | -1500 | -1000 | -1000 | 0 | 2000 | 0 | -1000 | -1000 | -1500 | -2000 | -2500 | -2000 |
| B | -2000 | -1500 | -1500 | -5000 | 0 | 2000 | - 0 | -500 | -1000 | -1500 | -2000 | -2500 |
| G♭ | -2500 | -2000 | -2000 | -1000 | -500 | 0 | 2000 | 0 | -500 | -1000 | -1500 | -2000 |
| D♭ | -2000 | -2500 | -2500 | -1500 | -1000 | -500 | 0 | 2000 | 0 | -500 | -1000 | -1500 |
| A♭ | -1500 | -2000 | -2000 | -2000 | -1500 | -1000 | -500 | 0 | 2000 | 0 | -500 | -1000 |
| E♭ | -1000 | -1500 | -1500 | -2500 | -2000 | -1500 | -1000 | -500 | 0 | 2000 | 0 | -500 |
| B♭ | -500 | -1000 | -1000 | -2000 | -2500 | -2000 | -1500 | -1000 | -500 | 0 | 2000 | 0 |
| F | 0 | -500 | -1000 | -1500 | -2000 | -2500 | -2000 | -1500 | -1000 | -500 | 0 | 2000 |

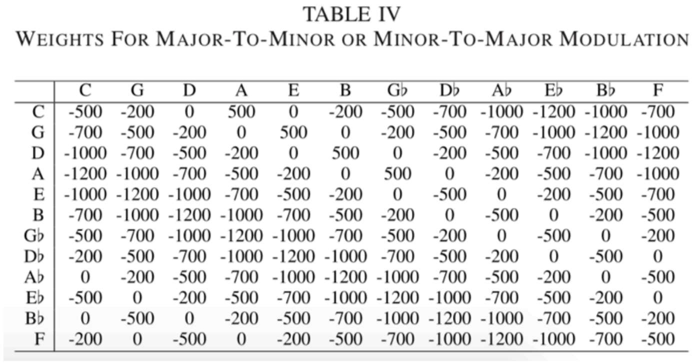
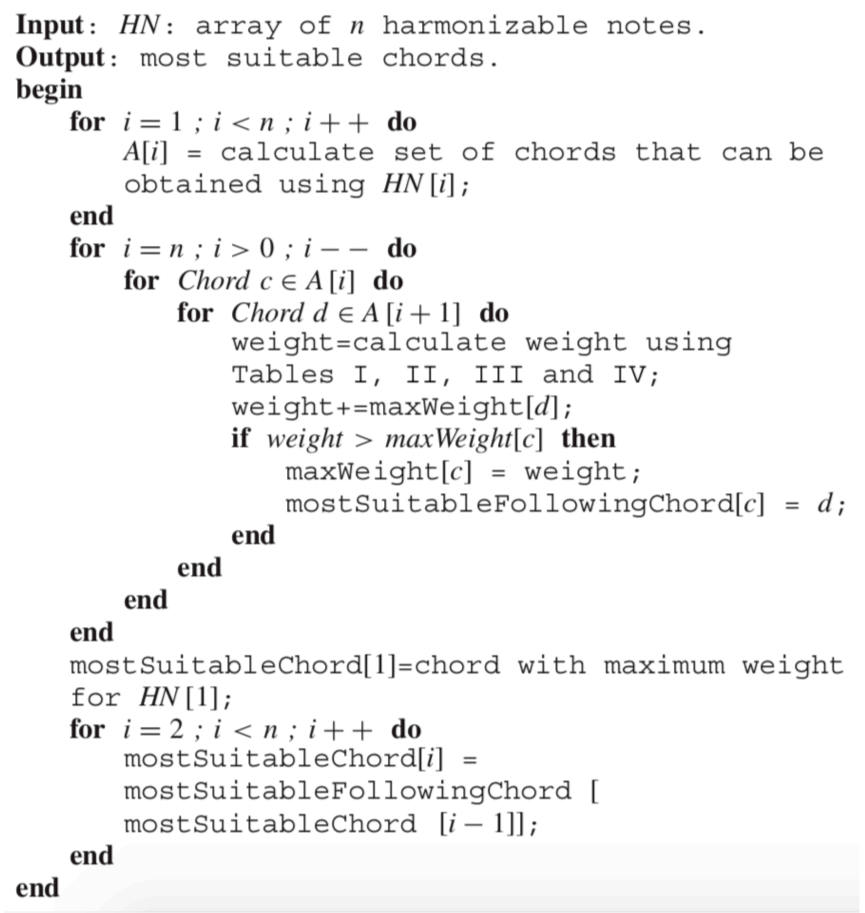


TABLE IV
WEIGHTS FOR MAJOR-TO-MINOR OR MINOR-TO-MAJOR MODULATION

| | C | G | D | A | E | B | G♭ | D♭ | A♭ | E♭ | B♭ | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | -500 | -200 | 0 | 500 | 0 | -200 | -500 | -700 | -1000 | -1200 | -1000 | -700 |
| G | -700 | -500 | -200 | 0 | 500 | 0 | -200 | -500 | -700 | -1000 | -1200 | -1000 |
| D | -1000 | -700 | -500 | -200 | 0 | 500 | 0 | -200 | -500 | -700 | -1000 | -1200 |
| A | -1200 | -1000 | -700 | -500 | -200 | 0 | 500 | 0 | -200 | -500 | -700 | -1000 |
| E | -1000 | -1200 | -1000 | -700 | -500 | -200 | 0 | -500 | 0 | -200 | -500 | -700 |
| B | -700 | -1000 | -1200 | -1000 | -700 | -500 | -200 | 0 | -500 | 0 | -200 | -500 |
| G♭ | -500 | -700 | -1000 | -1200 | -1000 | -700 | -500 | -200 | 0 | -500 | 0 | -200 |
| D♭ | -200 | -500 | -700 | -1000 | -1200 | -1000 | -700 | -500 | -200 | 0 | -500 | 0 |
| A♭ | 0 | -200 | -500 | -700 | -1000 | -1200 | -1000 | -700 | -500 | -200 | 0 | -500 |
| E♭ | -500 | 0 | -200 | -500 | -700 | -1000 | -1200 | -1000 | -700 | -500 | -200 | 0 |
| B♭ | 0 | -500 | 0 | -200 | -500 | -700 | -1000 | -1200 | -1000 | -700 | -500 | -200 |
| F | -200 | 0 | -500 | 0 | -200 | -500 | -700 | -1000 | -1200 | -1000 | -700 | -500 |

Source: "Memetic Music Composition."



```
Input: $HN$: array of $n$ harmonizable notes.
Output: most suitable chords.
begin
    for $i = 1 ; i < n ; i++$ do
        $A[i]$ = calculate set of chords that can be
        obtained using $HN[i]$;
    end
    for $i = n ; i > 0 ; i--$ do
        for $Chord\ c \in A[i]$ do
            for $Chord\ d \in A[i+1]$ do
                weight=calculate weight using
                Tables I, II, III and IV;
                weight+=maxWeight[$d$];
                if $weight > maxWeight[c]$ then
                    maxWeight[$c$] = weight;
                    mostSuitableFollowingChord[$c$] = $d$;
                end
            end
        end
    end
    mostSuitableChord[1]=chord with maximum weight
    for $HN[1]$;
    for $i = 2 ; i < n ; i++$ do
        mostSuitableChord[$i$] =
        mostSuitableFollowingChord [
        mostSuitableChord [$i-1$]];
    end
end
```

Source: "Memetic Music Composition."

## Composer Agent Cooperation

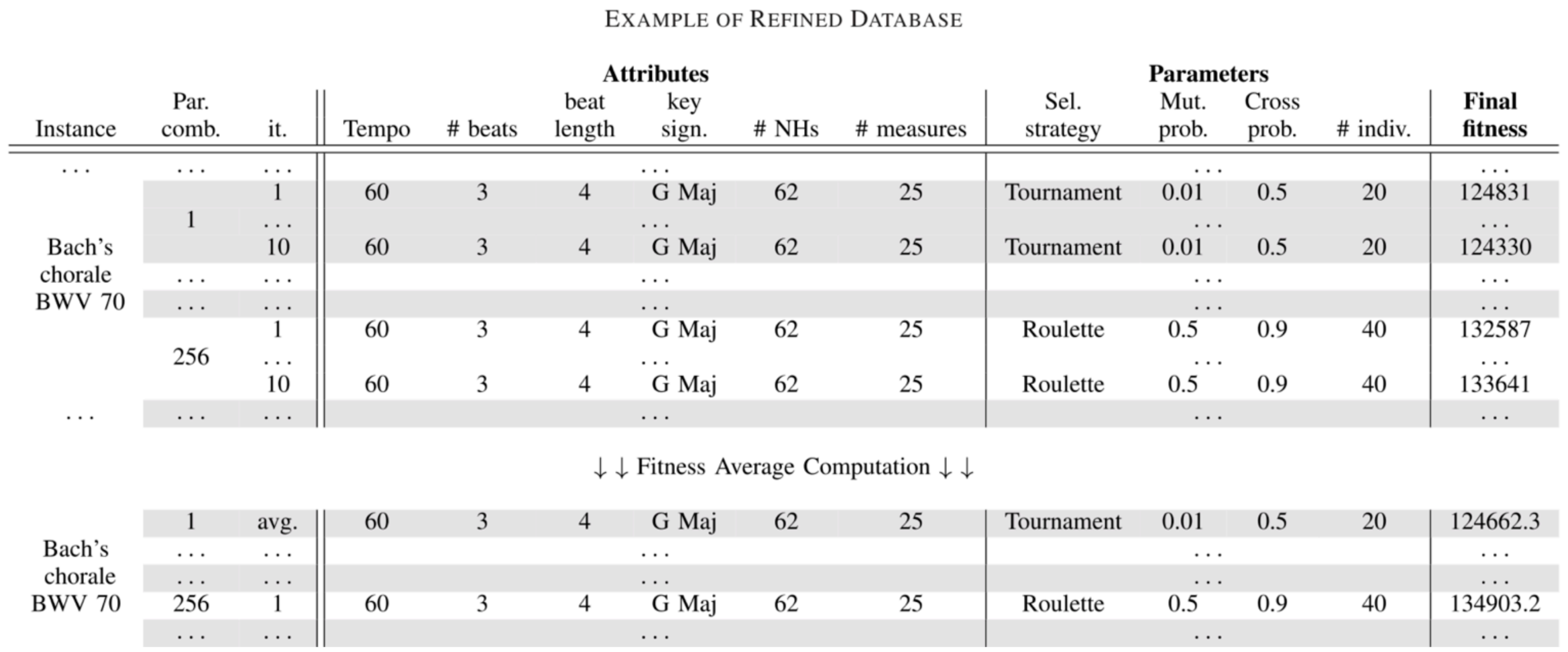
- Once the subspace is defined, six "composer agents" search for musical pieces within this subspace
- Master coordinator agent controls exchange of information between the composer agents

## Composer Agent Cooperation

- Composer agents with different search techniques:
    - Genetic algorithm
    - Particle swarm optimization
    - Ant colony optimization
    - Tabu search
    - Simulated annealing
    - Variable neighborhood search

## Composer Agent Cooperation

- The coordinator agent is trained to pick out strong melodies from each agent
- If a strong melody is found, it "intelligently moves" this information between agents to aid their searches
- Uses fuzzy decision trees to decide melody strength
- Training set: Bach



EXAMPLE OF REFINED DATABASE

| | | | Attributes | | | | | | Parameters | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance | Par. comb. | it. | Tempo | # beats | beat length | key sign. | # NHs | # measures | Sel. strategy | Mut. prob. | Cross prob. | # indiv. | Final fitness |
| ... | ... | ... | | | | ... | | | | ... | | | ... |
| | | 1 | 60 | 3 | 4 | G Maj | 62 | 25 | Tournament | 0.01 | 0.5 | 20 | 124831 |
| | 1 | ... | | | | ... | | | | ... | | | ... |
| | | 10 | 60 | 3 | 4 | G Maj | 62 | 25 | Tournament | 0.01 | 0.5 | 20 | 124330 |
| Bach's chorale BWV 70 | ... | ... | | | | ... | | | | ... | | | ... |
| | ... | ... | | | | ... | | | | ... | | | ... |
| | | 1 | 60 | 3 | 4 | G Maj | 62 | 25 | Roulette | 0.5 | 0.9 | 40 | 132587 |
| | 256 | ... | | | | ... | | | | ... | | | ... |
| | | 10 | 60 | 3 | 4 | G Maj | 62 | 25 | Roulette | 0.5 | 0.9 | 40 | 133641 |
| ... | ... | ... | | | | ... | | | | ... | | | ... |

↓ ↓ Fitness Average Computation ↓ ↓

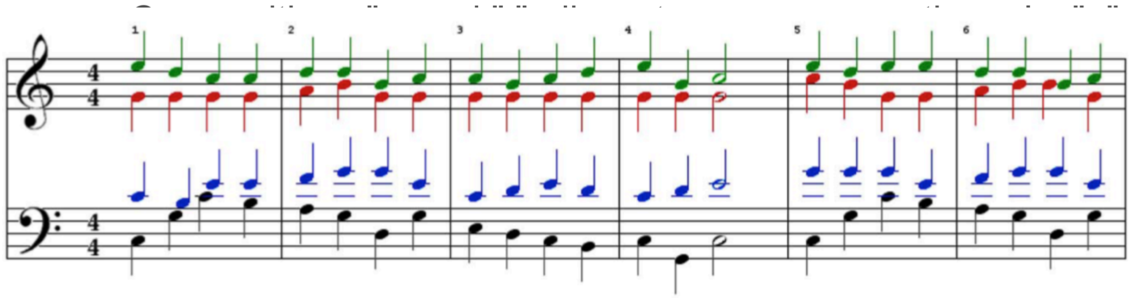
| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | avg. | 60 | 3 | 4 | G Maj | 62 | 25 | Tournament | 0.01 | 0.5 | 20 | 124662.3 |
| Bach's chorale BWV 70 | ... | ... | | | | ... | | | | ... | | | ... |
| | ... | ... | | | | ... | | | | ... | | | ... |
| | 256 | 1 | 60 | 3 | 4 | G Maj | 62 | 25 | Roulette | 0.5 | 0.9 | 40 | 134903.2 |
| | ... | ... | | | | ... | | | | ... | | | ... |

Source: "Memetic Music Composition."

## Conclusions

- Cooperation of composer agents improved results
- Some repeated errors: consecutive fifths, unisons, parallel octaves
- Once results were cleaned programmatically, listening tests were given to musical professionals

urprised that a computer



Solution obtained by $\mathrm{MMC}^{G\&P\&A}_{S\&T\&V}$.

Source: "Memetic Music Composition."

## Music Recommendation

How Spotify Makes Music Recommendations

## Spotify’s Discover Weekly Playlist

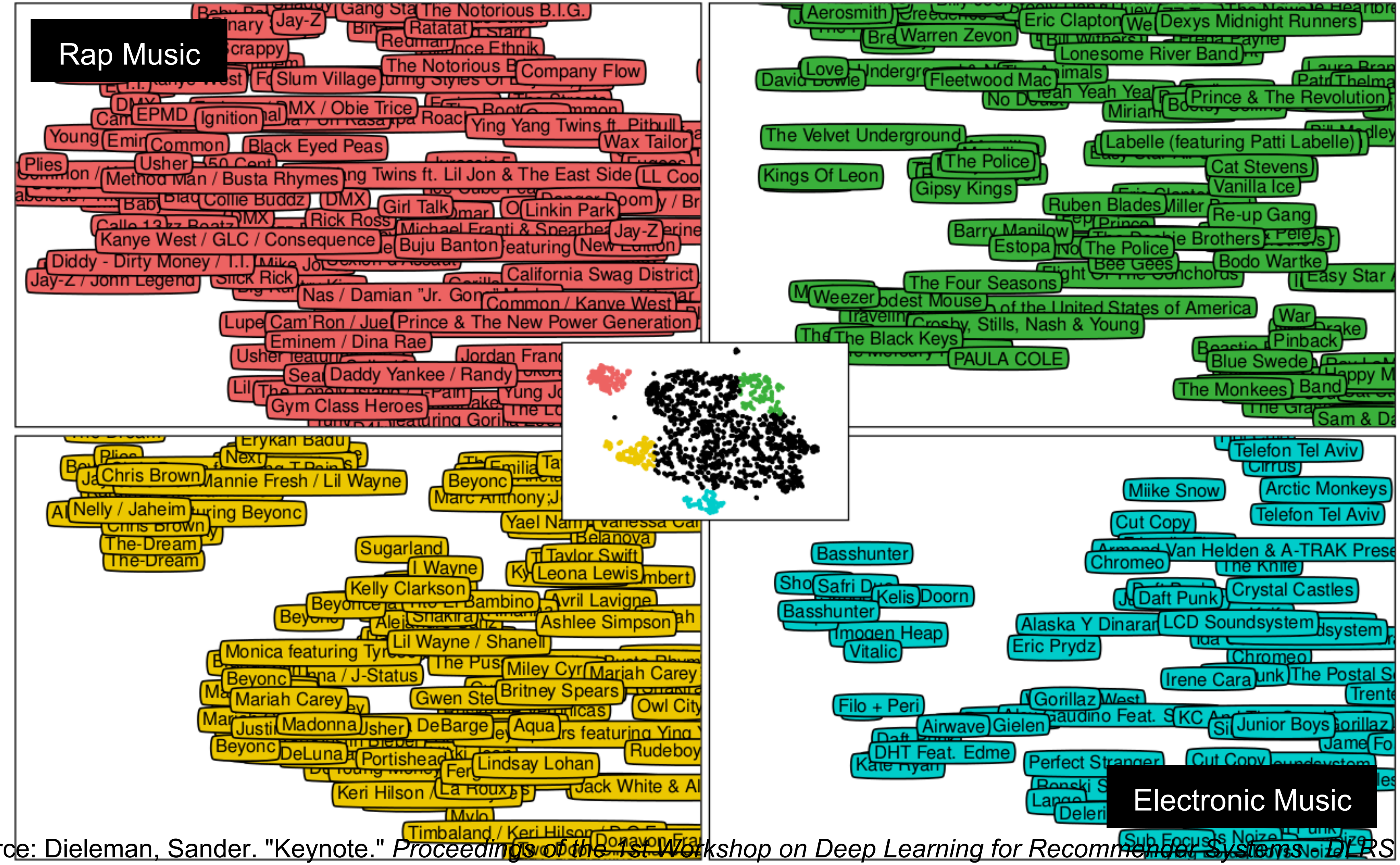
- Content-agnostic recommendations (collaborative filtering)
  - Determines a user’s preferences from historical usage data
  - If two users listen to largely the same set of songs, their tastes are probably similar
  - **Biggest flaw:** new and unpopular songs cannot be recommended (**cold-start problem**)
- Content-based recommendations
  - Lots of information: tags, artist and album information, lyrics, text mind from the web (reviews, interviews, …), and the **audio signal** itself.

Source: "Recommending Music on Spotify with Deep Learning." *Sander Dieleman*. N.p., 5 Aug. 2014. Web.

## t-Distributed Stochastic Neighbor Embedding

- t-SNE is a non-linear dimensionality reduction algorithm used for exploring high-dimensional data
- Algorithm
    - 1. Calculate similarity between audio signals.
    - 2. *...Complicated Mathematics/Statistics...*
    - 3. Data points are placed on a 2-D or 3-D plot. The closer two artists/songs are, the more similar they are.

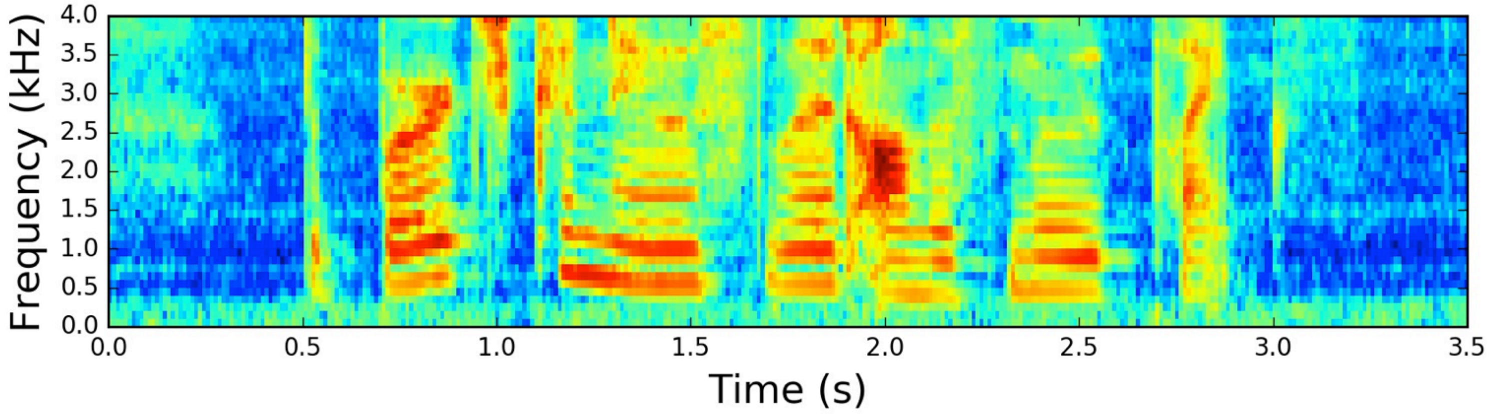
Source: Van Der Maaten, Laurens. "T-SNE." *Laurens Van Der Maaten*. N.p., 2017. Web.



Source: Dieleman, Sander. "Keynote." *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems - DLRS*

# Applying a Neural Network

- Use a convolutional neural network (CNN) to qualify and categorize songs
  - Alternative: Bag-of-words (BoW) representation
- The input to the network consists of mel-spectrograms



Source: Dieleman, Sander. "Keynote." *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems - DLRS*

## Network Layers

Primary Layers

- Finding songs that maximally activate a given filter in 30 seconds
  - Filters picked up: vibrato singing, ringing ambience, vocal thirds, bass drum sounds
- Finding songs that activate a given filter for the longest time, on average
  - Filters picked up: distortion, specific pitches, drones, chords

Terminal Layers

- At the topmost layer of the network, filters are very selective for certain subgenres.

Source: Dieleman, Sander. "Keynote." *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems - DLRS 2016*.

## Natural Language Processing

- Similar to Word2Vec
  - Takes words and encodes them into a mathematical representation -- a vector
- Spotify takes playlists and treats them as a paragraph or big block of text, and treats each song in the playlist as an individual word.
- This results in vector representations of songs that can be used to determine two pieces of music that are similar.
  - tackles cold-start problem

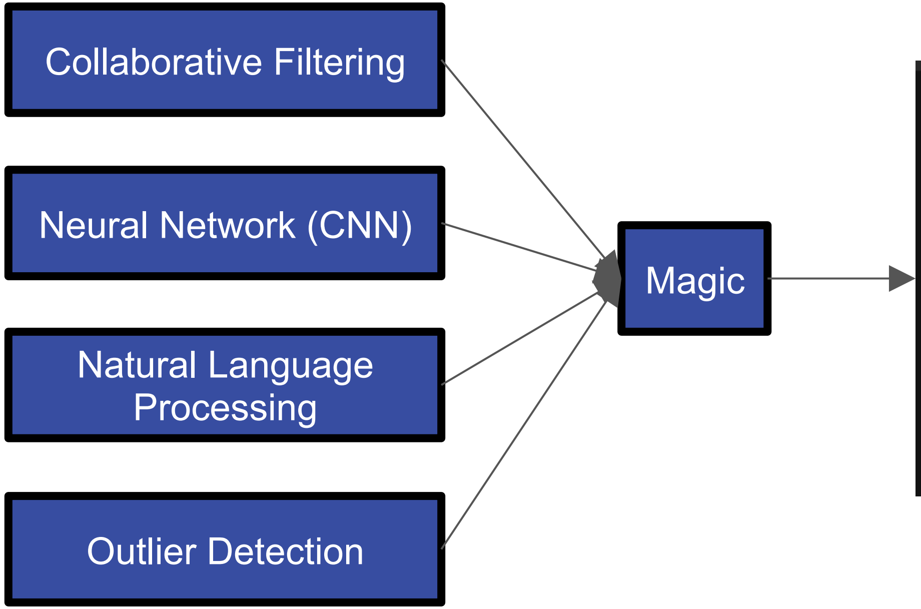
Source: "Recommending Music on Spotify with Deep Learning." *Sander Dieleman*. N.p., 5 Aug. 2014. Web.

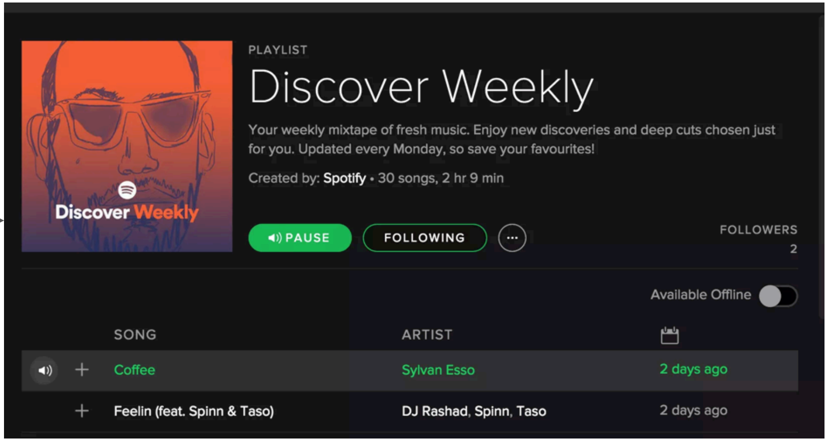
## Outlier Detection

- Able to determine if a particular usage is normal
- For example, if you usually listen to classic rock and ‘90s alternative, your Discover Weekly playlist won’t get filled up with pop hits when you accidently clicked on Justin Bieber one time

Source: "Recommending Music on Spotify with Deep Learning." *Sander Dieleman*. N.p., 5 Aug. 2014. Web.

# Music Discovery: Made Easy!





Your weekly mixtape of fresh music. Enjoy new discoveries and deep cuts chosen just for you. Updated every Monday, so save your favourites!

Created by: Spotify • 30 songs, 2 hr 9 min

| SONG | ARTIST | |
|---|---|---|
| Coffee | Sylvan Esso | 2 days ago |
| Feelin (feat. Spinn & Taso) | DJ Rashad, Spinn, Taso | 2 days ago |

Source: "Recommending Music on Spotify with Deep Learning." *Sander Dieleman*. N.p., 5 Aug. 2014. Web.

## Music Recommendation

DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation

Liebman, Elad, et al. “Peter Stone's Selected Publications.” *Peter Stone: DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation*, Peter Stone's Selected Publications, May 2015, www.cs.utexas.edu/users/pstone/Papers/bib2html/b2hd-AAMAS2015-eladlieb.html.

## Concepts

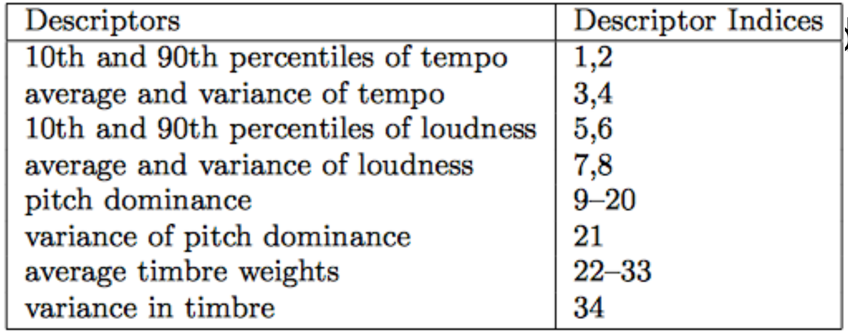
- This article focuses on DJ-MC, which is a framework for music recommendation that recommends songs based on song sequences and transitions in the playlist
- Many music applications till now have suggested individual songs based on other individual songs the user listened to (Pandora, Jango, etc.)

# The Process

1. Formulate the problem of choosing which sequence of songs to use - Markov Decision Process (MDP: S, A, P, R, T)
   a. S: set of states
   b. A: set of actions
   c. P: the probability  of transitioning from state s to s' when performing action a
   d. R: state - action reward (performing action a will be given a reward r)
   e. T: set of terminal states
2. Test whether or not the sequence affects the listener experience positively or negatively
3. Make sure the playlist created using song sequence and transitions affect

# Modeling

- Song descriptors: used as factors of the representation of a song
  - Spectral fingerprint of the song, rhythmic characteristics, overall loudness, change in volume overtime
- ( different songs



| Descriptors | Descriptor Indices |
|---|---|
| 10th and 90th percentiles of tempo | 1,2 |
| average and variance of tempo | 3,4 |
| 10th and 90th percentiles of loudness | 5,6 |
| average and variance of loudness | 7,8 |
| pitch dominance | 9–20 |
| variance of pitch dominance | 21 |
| average timbre weights | 22–33 |
| variance in timbre | 34 |

Source: "DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation"

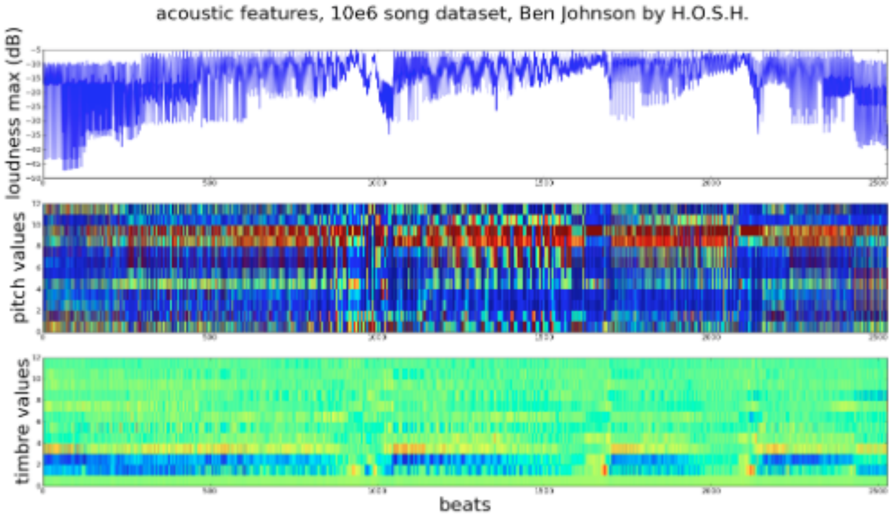
- Also uses amount of reward as an aspect to learn about preferences
  - From songs in isolation

## DJ-MC Architecture

- Two major roles:
  - Learning listener parameters
  - Planning the sequence of songs
- Φ - listener parameter
- R - reward
- a - action
- Ө - feature vector (#descriptors x #bins)
  - Bins - used to restrict a prescribed set of values associated with the amplitude of a

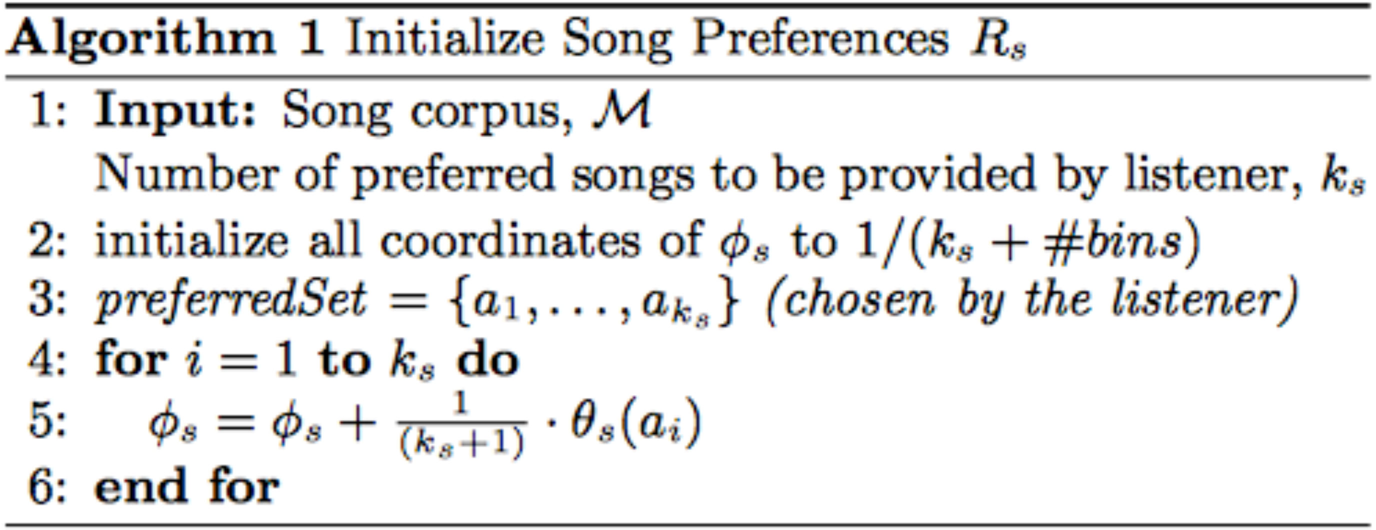
# Data

- Used Million Song Dataset to retrieve large amounts of popular music
- Sources for this article: Yes.com, Last.fm



Source: "DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation"

## Algorithm 1

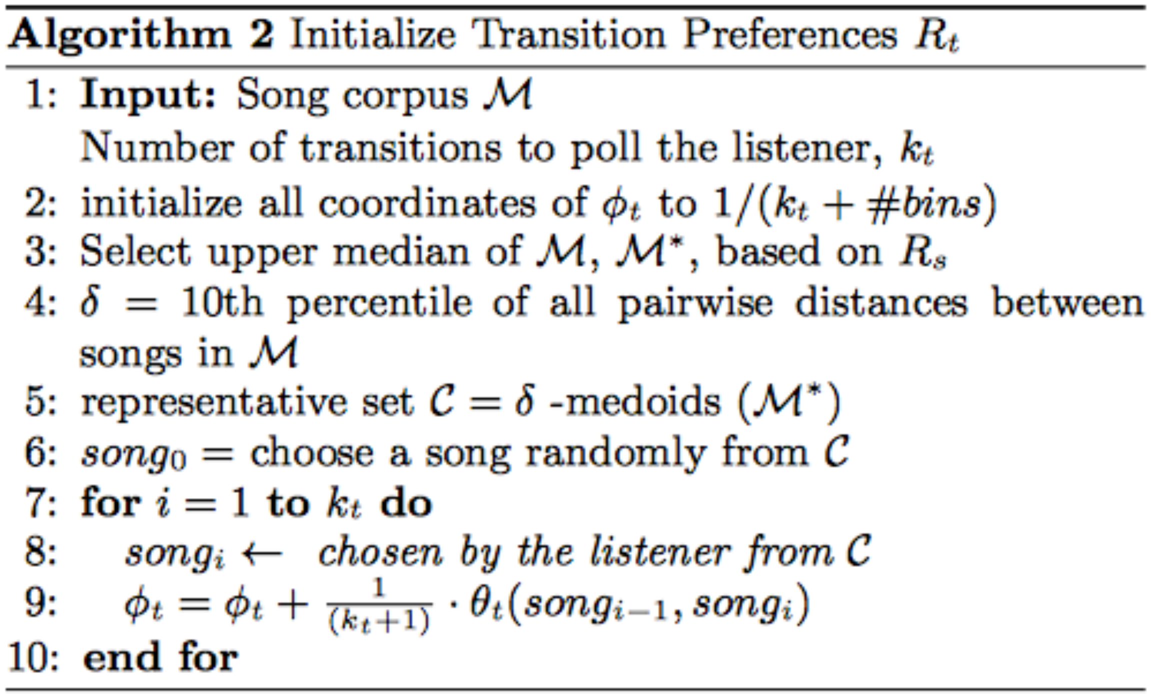


**Algorithm 1** Initialize Song Preferences $R_s$

```
1: Input: Song corpus, $\mathcal{M}$
   Number of preferred songs to be provided by listener, $k_s$
2: initialize all coordinates of $\phi_s$ to $1/(k_s + \#bins)$
3: $preferredSet = \{a_1, \ldots, a_{k_s}\}$ (chosen by the listener)
4: for $i = 1$ to $k_s$ do
5:     $\phi_s = \phi_s + \frac{1}{(k_s+1)} \cdot \theta_s(a_i)$
6: end for
```

Source: "DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation"

# Algorithm 2

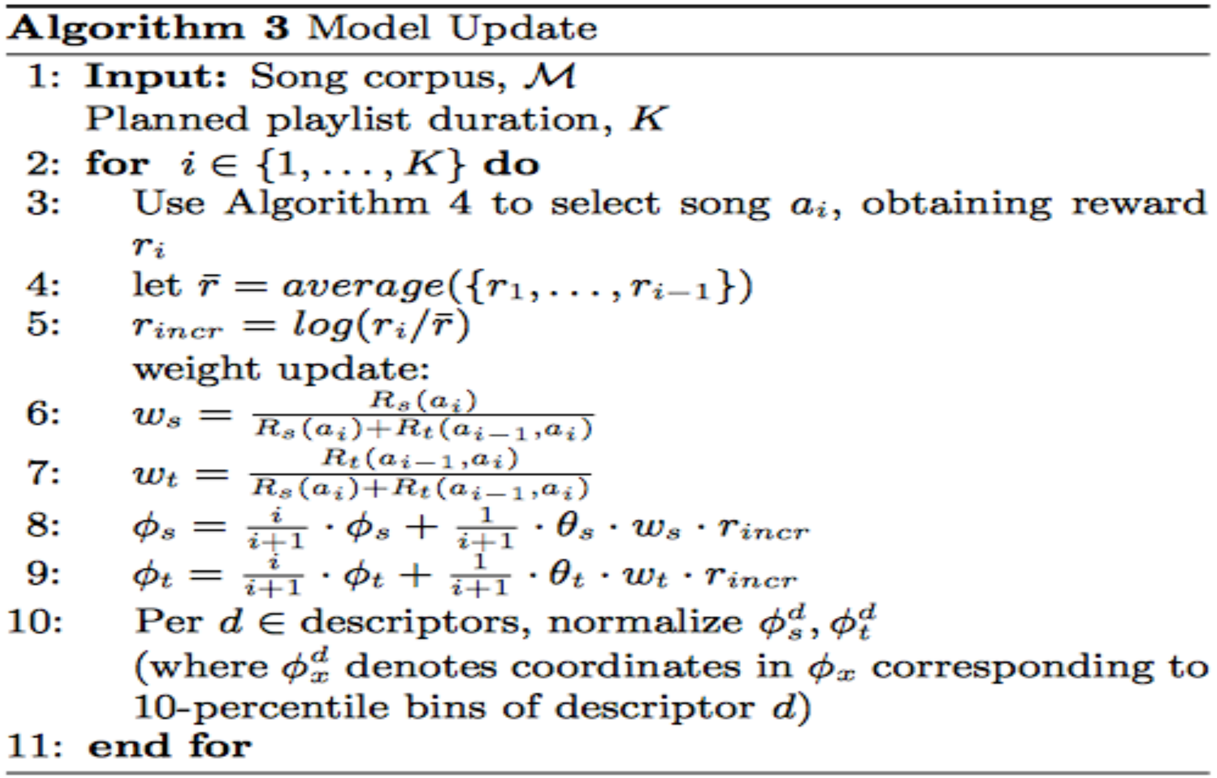


**Algorithm 2** Initialize Transition Preferences $R_t$

```
1: Input: Song corpus $\mathcal{M}$
   Number of transitions to poll the listener, $k_t$
2: initialize all coordinates of $\phi_t$ to $1/(k_t + \#bins)$
3: Select upper median of $\mathcal{M}$, $\mathcal{M}^*$, based on $R_s$
4: $\delta$ = 10th percentile of all pairwise distances between songs in $\mathcal{M}$
5: representative set $\mathcal{C} = \delta$ -medoids $(\mathcal{M}^*)$
6: $song_0$ = choose a song randomly from $\mathcal{C}$
7: for $i = 1$ to $k_t$ do
8:     $song_i \leftarrow$ chosen by the listener from $\mathcal{C}$
9:     $\phi_t = \phi_t + \frac{1}{(k_t+1)} \cdot \theta_t(song_{i-1}, song_i)$
10: end for
```

Source: "DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation"

# Algorithm 3

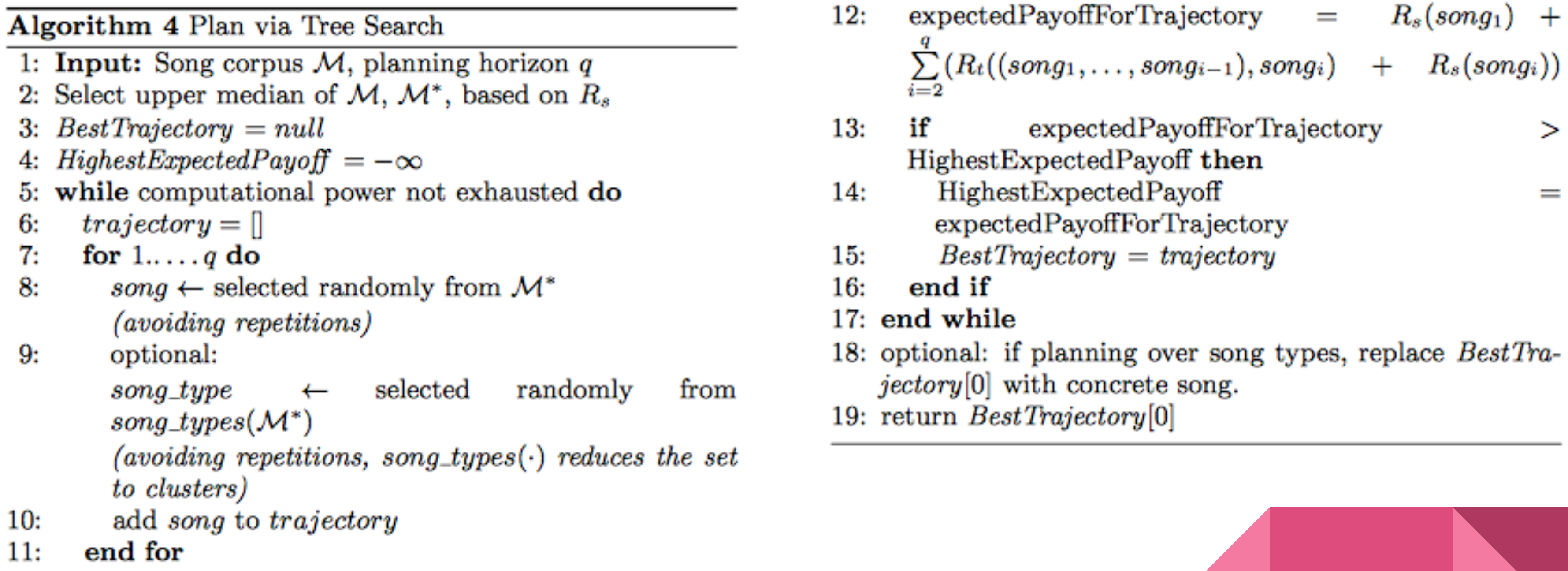


**Algorithm 3** Model Update

```
1: Input: Song corpus, $\mathcal{M}$
   Planned playlist duration, $K$
2: for $i \in \{1, \ldots, K\}$ do
3:     Use Algorithm 4 to select song $a_i$, obtaining reward
       $r_i$
4:     let $\bar{r} = average(\{r_1, \ldots, r_{i-1}\})$
5:     $r_{incr} = log(r_i / \bar{r})$
       weight update:
6:     $w_s = \frac{R_s(a_i)}{R_s(a_i) + R_t(a_{i-1}, a_i)}$
7:     $w_t = \frac{R_t(a_{i-1}, a_i)}{R_s(a_i) + R_t(a_{i-1}, a_i)}$
8:     $\phi_s = \frac{i}{i+1} \cdot \phi_s + \frac{1}{i+1} \cdot \theta_s \cdot w_s \cdot r_{incr}$
9:     $\phi_t = \frac{i}{i+1} \cdot \phi_t + \frac{1}{i+1} \cdot \theta_t \cdot w_t \cdot r_{incr}$
10:    Per $d \in$ descriptors, normalize $\phi_s^d, \phi_t^d$
       (where $\phi_x^d$ denotes coordinates in $\phi_x$ corresponding to
       10-percentile bins of descriptor $d$)
11: end for
```

Source: "DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation"

# Algorithm 4

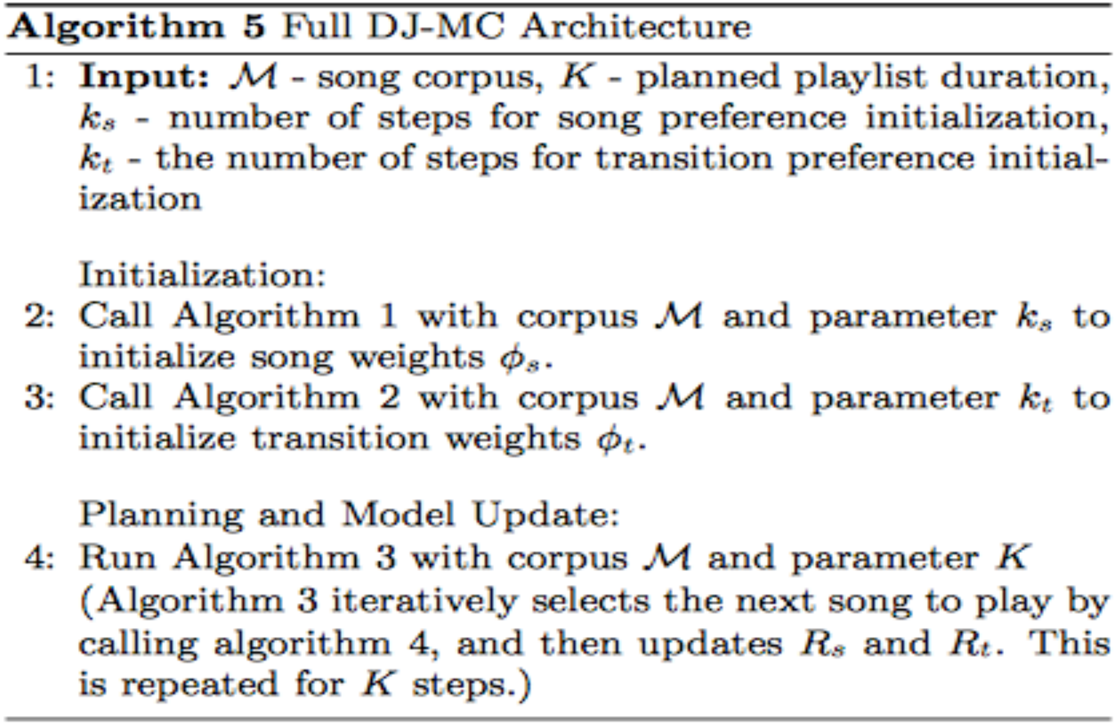


**Algorithm 4** Plan via Tree Search

```
1: Input: Song corpus $\mathcal{M}$, planning horizon $q$
2: Select upper median of $\mathcal{M}$, $\mathcal{M}^*$, based on $R_s$
3: $BestTrajectory = null$
4: $HighestExpectedPayoff = -\infty$
5: while computational power not exhausted do
6:     $trajectory = []$
7:     for $1 \ldots \ldots q$ do
8:         $song \leftarrow$ selected randomly from $\mathcal{M}^*$
           $(avoiding\ repetitions)$
9:         optional:
           $song\_type \leftarrow$ selected randomly from $song\_types(\mathcal{M}^*)$
           $(avoiding\ repetitions,\ song\_types(\cdot)\ reduces\ the\ set\ to\ clusters)$
10:        add $song$ to $trajectory$
11:    end for
12:    expectedPayoffForTrajectory $= R_s(song_1) + \sum_{i=2}^{q}(R_t((song_1, \ldots, song_{i-1}), song_i) + R_s(song_i))$
13:    if expectedPayoffForTrajectory $>$ HighestExpectedPayoff then
14:        HighestExpectedPayoff $=$ expectedPayoffForTrajectory
15:        $BestTrajectory = trajectory$
16:    end if
17: end while
18: optional: if planning over song types, replace $BestTrajectory[0]$ with concrete song.
19: return $BestTrajectory[0]$
```

Source: "DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation"

# Algorithm 5



**Algorithm 5** Full DJ-MC Architecture

```
1: Input: $\mathcal{M}$ - song corpus, $K$ - planned playlist duration,
   $k_s$ - number of steps for song preference initialization,
   $k_t$ - the number of steps for transition preference initial-
   ization

   Initialization:
2: Call Algorithm 1 with corpus $\mathcal{M}$ and parameter $k_s$ to
   initialize song weights $\phi_s$.
3: Call Algorithm 2 with corpus $\mathcal{M}$ and parameter $k_t$ to
   initialize transition weights $\phi_t$.

   Planning and Model Update:
4: Run Algorithm 3 with corpus $\mathcal{M}$ and parameter $K$
   (Algorithm 3 iteratively selects the next song to play by
   calling algorithm 4, and then updates $R_s$ and $R_t$. This
   is repeated for $K$ steps.)
```

Source: "DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation"

## Evaluation

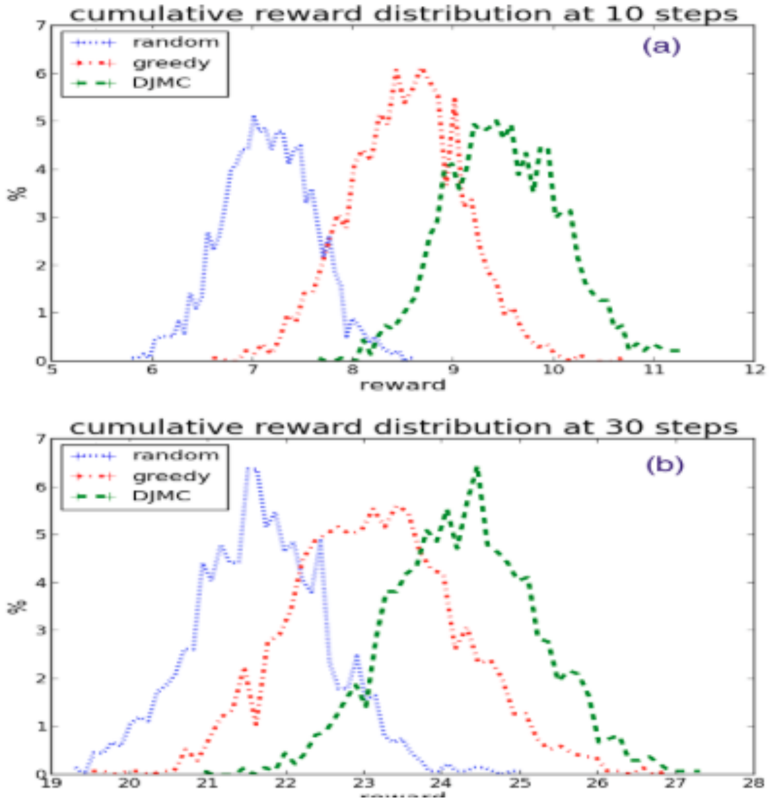
- Cannot use human testing - time constraints
- DJ-MC was tested on listener models that were built using actual playlists made by read individuals

Challenges:

- Cannot compare results to any other system of its kind
- We have to compare to a system that chooses songs randomly and one that chooses songs based on solely highest reward (Algorithm 1)

## Evaluation

Source: "DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation"



## Human-based Experiment

- Chose playlist from Rolling Stone's Magazine 500 greatest hits
- Each subject had to select like or dislike every time a song was played
- This experiment was compared to the greedy baseline from before
    - Greedy baseline concluded in better results in the early stages, but once DJ-MC started learning it increased in positive results faster
- Can conclude that DJ-MC offers significant improvement in song recommendation compared to traditional systems that use only reward mechanisms

Questions?