Machine Learning Applied to Musical Improvisation

Robert M. Keller
Harvey Mudd College

Constructive Machine Learning Workshop
NIPS 2013, 10 December 2013
Lake Tahoe, Nevada, USA

Abstract

Development of music education software inevitably leads to questions of how to acquire musical knowledge to be made available to the student user. I will describe machine learning of patterns for accompaniment styles and grammars for improvisation, based on melodic abstraction, clustering, and chaining. I will also discuss supervised and unsupervised approaches to improvising over chord progressions using neural network. Finally, I will mention a challenging unsolved application: learning to classify idiomatic patterns in chord progressions.

Collaborators for Machine Learning Aspects

Jon Gillick

Kevin Tang

Jim Herold

Brandy McMenamy

Sayuri Soejima

Hayden Blauzvern

Greg Bickerman

Sam Bosley

Peter Swire

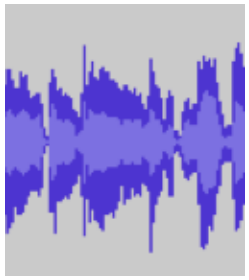
Kevin Choi

Context: Educational Music Software

- Intelligent Music Software Project at Harvey Mudd College
  - **Impro-Visor**
    - Style pattern learning
    - Melodic grammar learning
    - Melody generation using neural network critic
  - RBM-provisor: Using Deep-Belief Networks
    - Melody generation using unsupervised learning

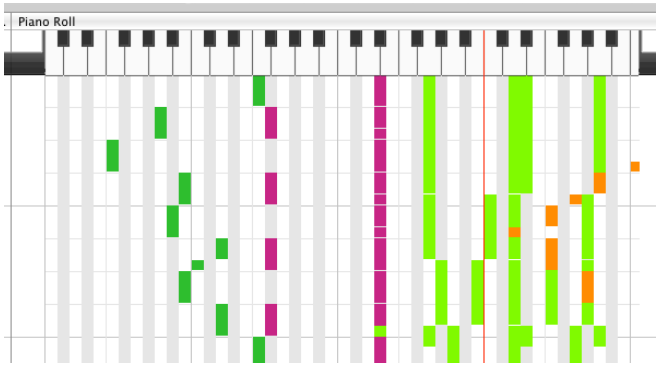
Two Major Music Universes

- **Audio** (mp3, wav, AIFF, au, …)



- **Symbolic**

  MIDI = “Musical Instrument Digital Interface”



Example: SmartMusic
MakeMusic, Inc.

- Provides feedback for student practice sessions ("used by over 75,000 students worldwide")
- http://www.youtube.com/watch?v=xhYXO6TPKw4
- Developed by Prof. Roger Dannenberg at CMU.
- Proprietary

Learning

- Intelligent software can also “learn”, so as to ***improve*** its ability to make decisions beneficial to the user.

- Also ideally, humans can learn from the software, whether or not the software learns.

Music Plus One (formerly Music++)

Prof. Chris Raphael, Indiana University

- Uses Hidden Markov Models
- Virtual orchestra anticipates player's tempo, follows retakes, etc.
- http://www.music.informatics.indiana.edu/~craphael/music_plus_one/index.html

Creativity

Ideally, intelligent music software can also “create”, i.e. use its ability to make decisions to produce new results that will inform or intrigue the user.

GenJam (Genetic Jammer)

Prof. Al Biles, Rochester Inst. of Technology

- Improvises jazz solos
- Trades interactively with human soloist.
  - http://www.youtube.com/watch?v=xWHU8uE043g
- Learning based on Genetic Algorithm
- Proprietary code

HMC Intelligent Music Software Project

- Oriented toward helping musicians learn to improvise
- Focus is on jazz education, but not limited to jazz

Impro-Visor

- Short for “Improvisation Advisor”.
- A software “workbook” that can learning to improvise by:
  - Helping users **write out** solos
  - **Creating** solos while **trading** with user
  - Playing backing accompaniments

Impro-Visor Parameters

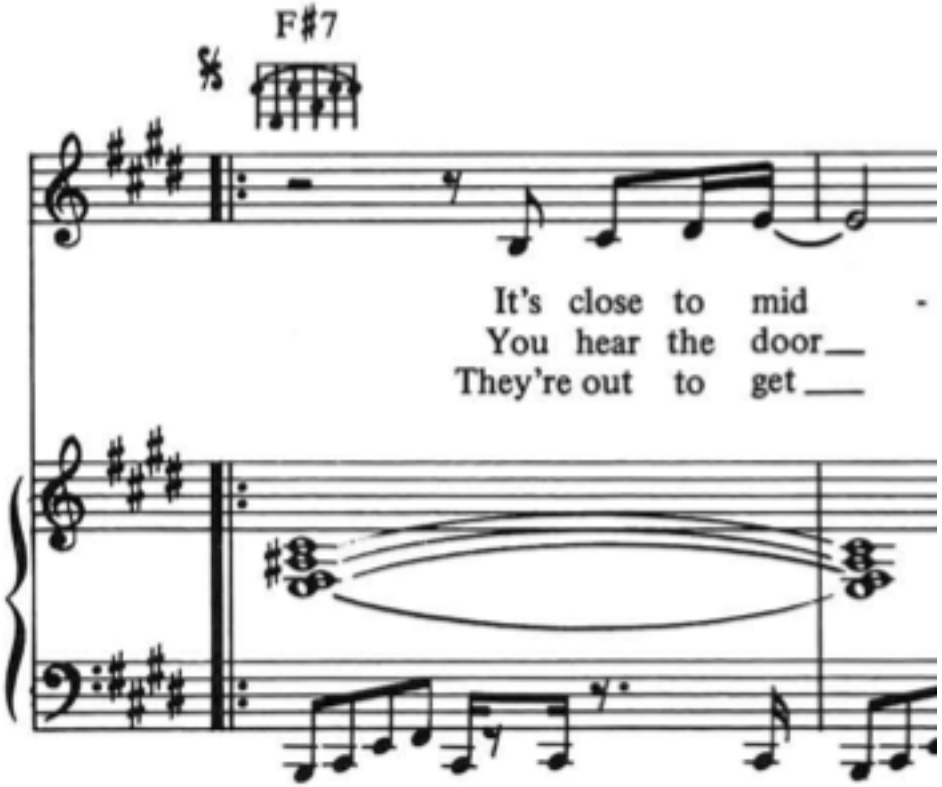
- Much musical information is in the form of user-editable text files:
  - **Vocabulary** defines

    Scales, Chords, Cells, Idioms, Licks, Quotes
  - **Styles** govern

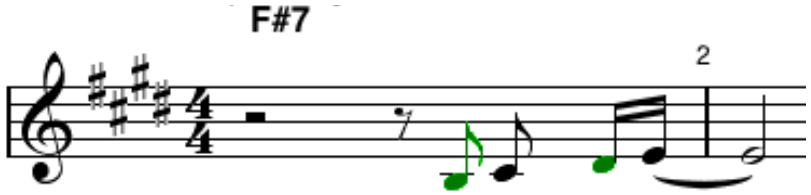
    How accompaniment is played and sound
  - **Grammar** creates melodies

    Somewhat in the style of specific players
  - **Leadsheet** specifies
    - Chord progression
    - Melody, solos

Leadsheet vs. Sheet Music



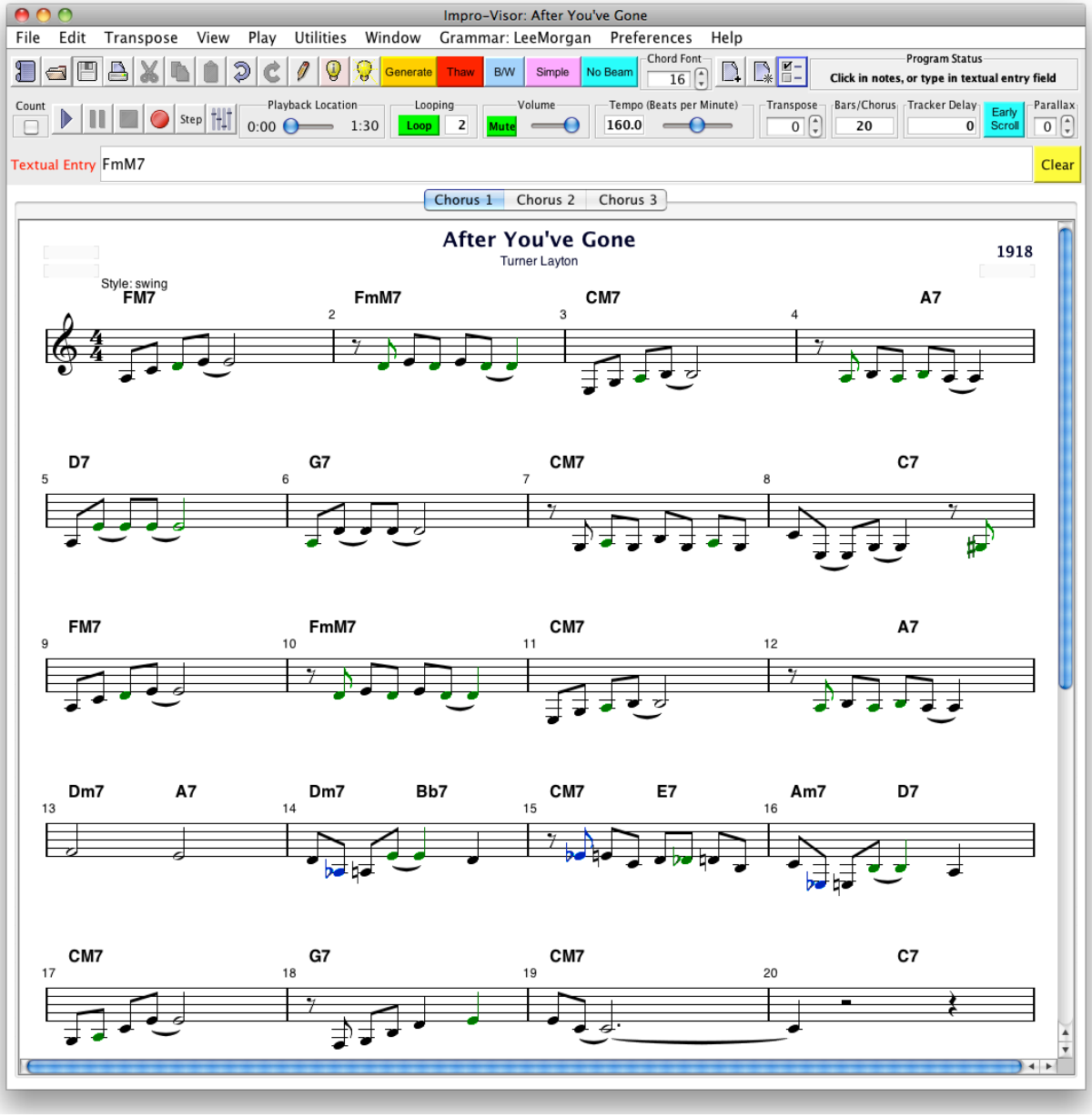
1 bar of **sheet music**



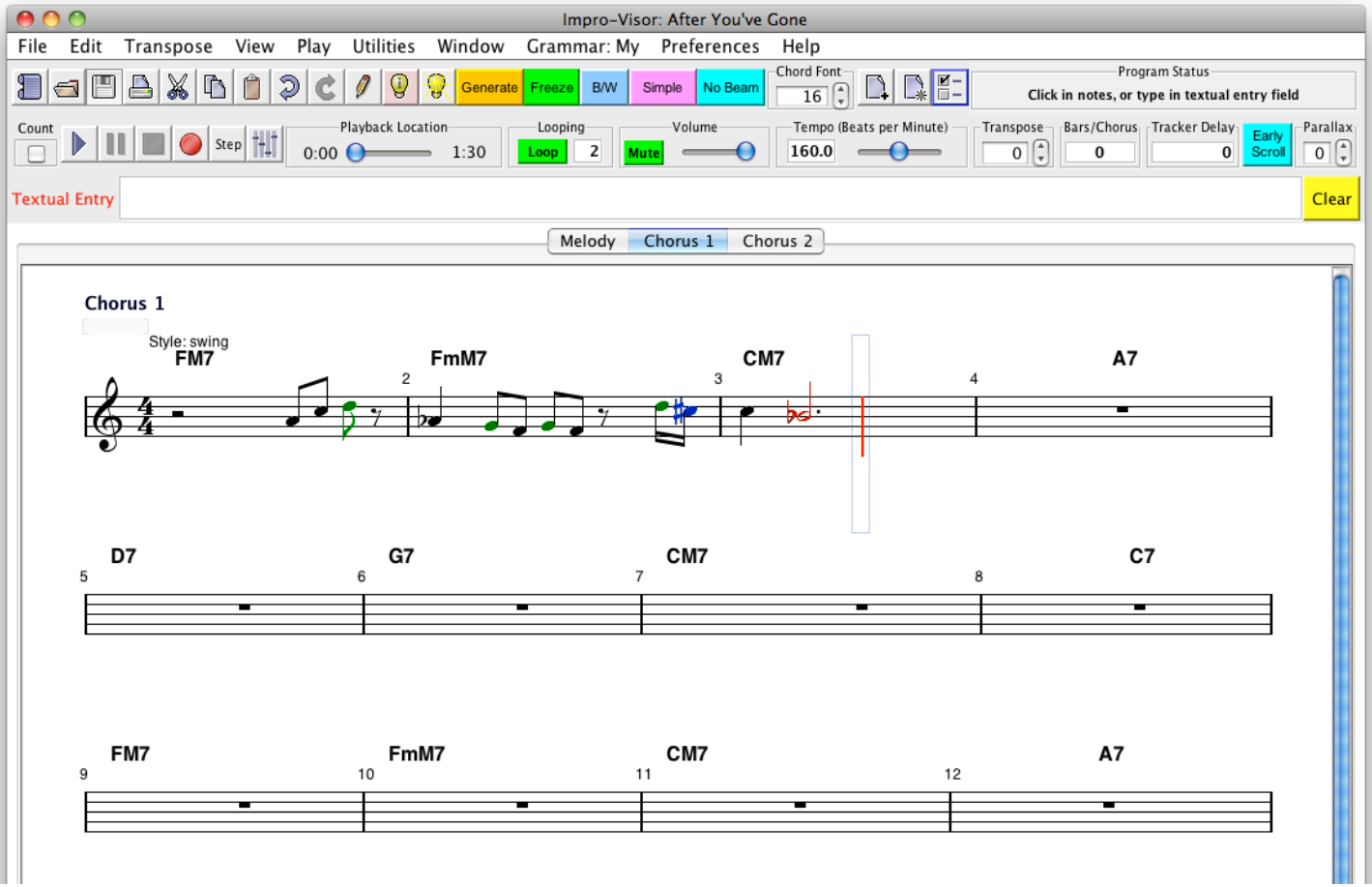
1 bar of a **leadsheet**

In a leadsheet, the accompaniment aspect is left to the performer.

Impro-Visor Leadsheet View



User Constructing a Solo

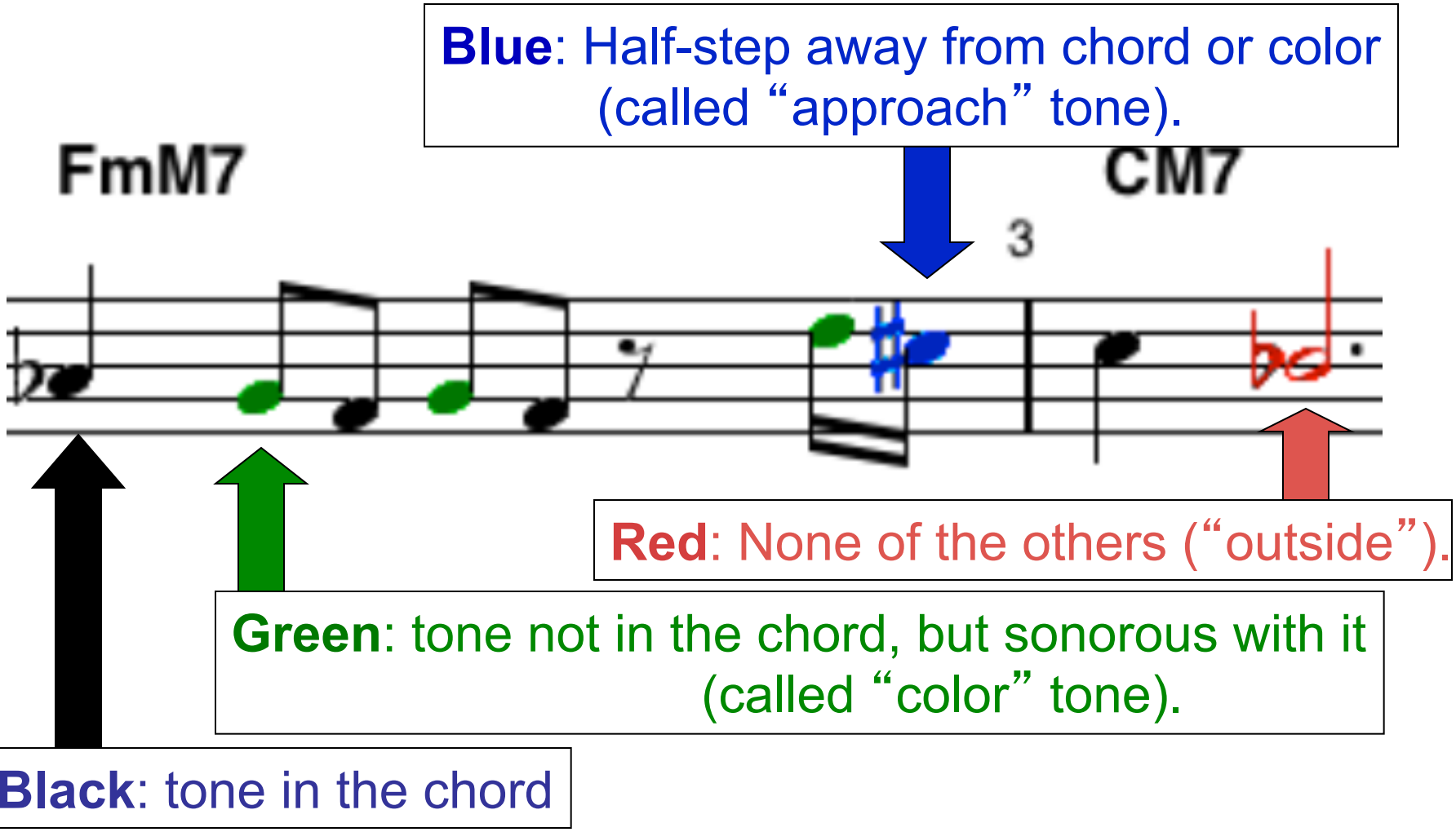


Entry Options:

- Point-and-click
- Cut-and-paste
- Textual
- Midi-keyboard

Note Coloration: Chord-Related

Part of the “Advisor” aspect of Impro-Visor



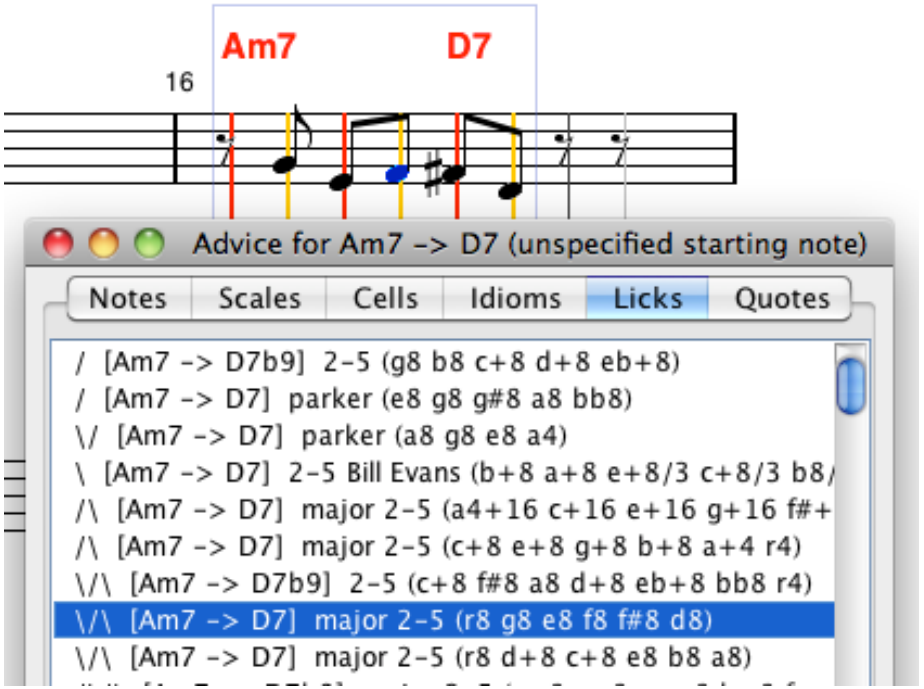
A Helpful Insight

- Note coloration **categories**, originally intended to educate users, can be useful for **machine learning** aspects.
- Use in Grammar Learning
- Use in Critic Development

Generating Licks for Examples

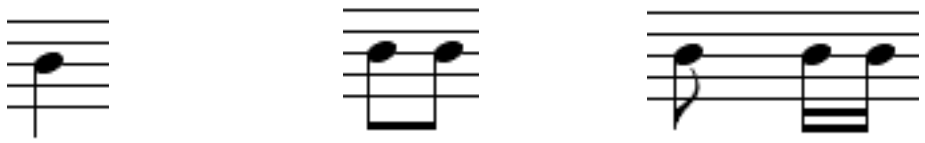
- **Lick** = a short melodic phrase
  - sometimes idiomatic
  - sometimes original

- Prior to introducing lick generation, Impro-Visor used a **database** to store lick suggestions.



Probabilistic Grammar Illustration

- We could fill a beat with a variety of rhythms:



. . .

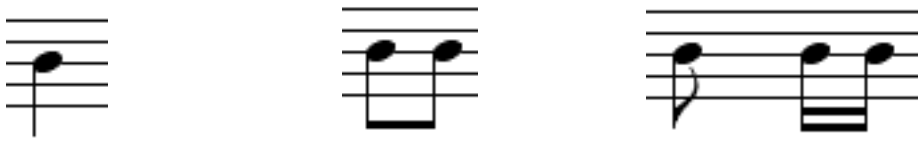
- Let B denote one beat of music
- A grammar represents all of these possibilities:

B → X4	4 means quarter note
B → X8 X8	8 means eighth note
B → X8 X16 X16	etc.

Here X4, X8, X16 are understood “terminal” symbols, while B is a non-terminal to be expanded.

Probabilistic Grammar Illustration

- Assign a probability to the various choices
- Probabilities will then dictate a prevalent style



- A grammar represents a distribution of these possibilities:

B → X4	p = 0.3	common
B → X8 X8	p = 0.6	frequent
B → X8 X16 X16	p = 0.1	rare

Grammars Can Exhibit Hierarchy and Recurrence

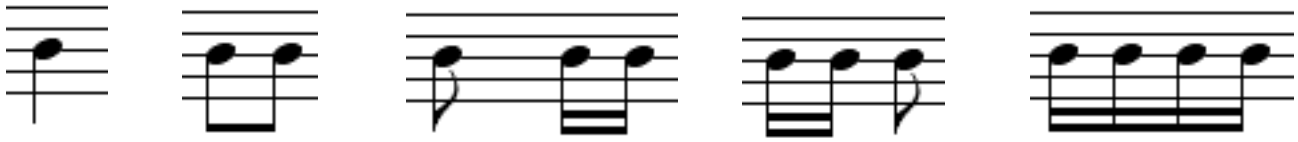
- Instead of

B → X4	p = 0.3	common
B → X8 X8	p = 0.6	frequent
B → X8 X16 X16	p = 0.1	rare

- Use

B → X4	p = 0.3	common
B → C C	p = 0.7	frequent
C → X8	p = 0.8	very frequent
C → X16 X16	p = 0.2	rare

- Generates



p = 0.3 p = 0.448 p = 0.112 p = 0.112 p = 0.028

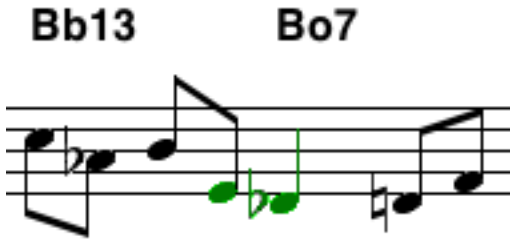
Recurrence Allows a Grammar to Fill an Arbitrary Number of Beats

- R → B R One beat, then more
- R → empty No more

- So R can produce B, BB, BBB, BBBB, etc.

Abstract vs. Real Melodies

A real melody on the staff and as text



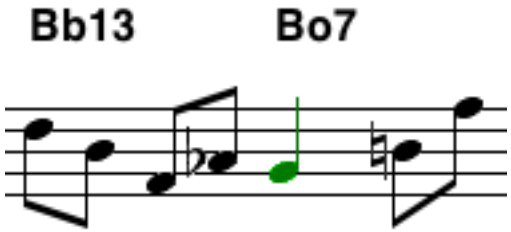
c+8 ab8 bb8 e8 db4 d8 f8

Abstract melody as text: C8 C8 C8 C8 L4 C8 C8

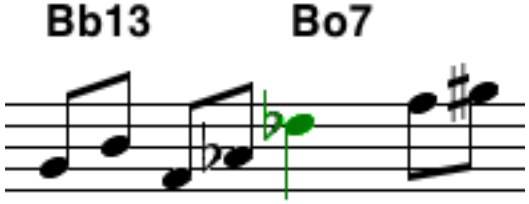
C = **C**hord tone
L = co**L**or tone
8 = 8th note
4 = quarter noter

Two other melodies with the same abstract melody

d+8 bb8 f8 ab8 g4 b8 f+8

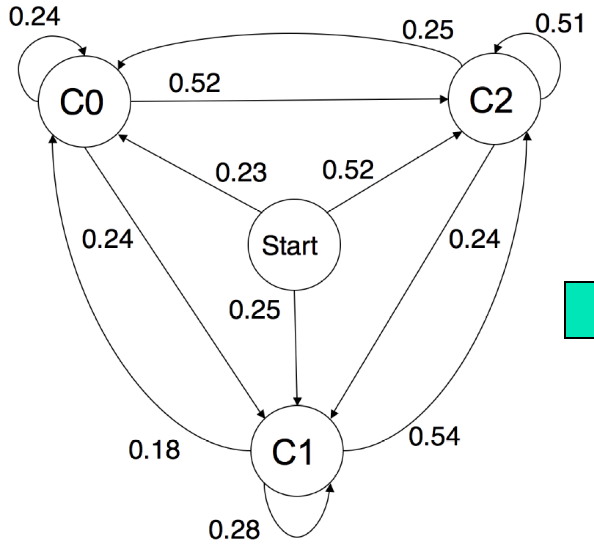


g8 bb8 f8 ab8 db+4 f+8 g#+8



Markov Chains in Grammars

- Recurrent productions allow us to embed an arbitrary Markov chain in the grammar.
- The reason for wanting this will be explained shortly.



Markov chain

Production Rule	Probability
Start(Z)→ C0(Z)	0.23
Start(Z) → C1(Z)	0.25
Start(Z) → C2(Z)	0.52
C0(0) → ()	1
C1(0) → ()	1
C2(0) → ()	1
C0(Z) → Q0 C0(Z-1)	0.24
C0(Z) → Q0 C1(Z-1)	0.24
C0(Z) → Q0 C2(Z-1)	0.52
C1(Z) → Q1 C0(Z-1)	0.18
C1(Z) → Q1 C1(Z-1)	0.28
C1(Z) → Q1 C2(Z-1)	0.54
C2(Z) → Q2 C0(Z-1)	0.25
C2(Z) → Q2 C1(Z-1)	0.24
C2(Z) → Q2 C2(Z-1)	0.51
Q0→((Δ 0 0 R2 R4 R8 C16/3) (Δ1 1 A16/3 L16/3)	1
Q1→((Δ 0 0 C8) (Δ -9 -9 C8) (Δ 2 3 C8 G4+8 R4))	1
Q2→((Δ 0 0 C4/3) (Δ 1 2 L4/3 A4/3) (Δ-7 -1 C4/3 G4 C8/3))	1

Grammar

A Complete Grammar (Terminals in Bold)

(startsymbol P)
(base (P 0) () 1.0)
(rule (M4) (**A4**) 0.01)
(rule (M4) (**C4**) 0.2)
(rule (M4) (**C4**) 0.1)
(rule (M8) (**A8**) 0.01)
(rule (M8) (**C8**) 0.4)
(rule (M8) (**C8**) 0.2)
(rule (M8) (**C8**) 0.1)
(rule (N2) (**C2**) 1.0)
(rule (N4) (**M4**) 0.75)
(rule (N4) (**R4**) 0.25)
(rule (N8) (**M8**) 0.9)
(rule (N8) (**R8**) 0.1)

(rule (P Y) (Seg1 (P (- Y 120))) 0.0010)
(rule (P Y) (Seg2 (P (- Y 240))) 0.25)
(rule (P Y) (Seg4 (P (- Y 480))) 0.75)
(rule (Seg1) (C4) 1.0)
(rule (Seg2) (N2) 0.06)
(rule (Seg2) (N8 C4.) 0.3)
(rule (Seg2) (V2) 0.3)
(rule (Seg2) (V4 V4) 0.6)
(rule (Seg2) (V8 N4 V8) 0.12)
(rule (Seg2) (V8 V8 V8 V8) 0.6)
(rule (Seg4) (C4. N8 Seg2) 0.1)
(rule (Seg4) (C4/3 C4/3 C4/3 Seg2) 0.02)
(rule (Seg4) (Seg2 C4/3 C4/3 C4/3) 0.02)
(rule (Seg4) (Seg2 V4 V4) 0.52)
(rule (Seg4) (V8 N4 N4 N4 V8) 0.01)

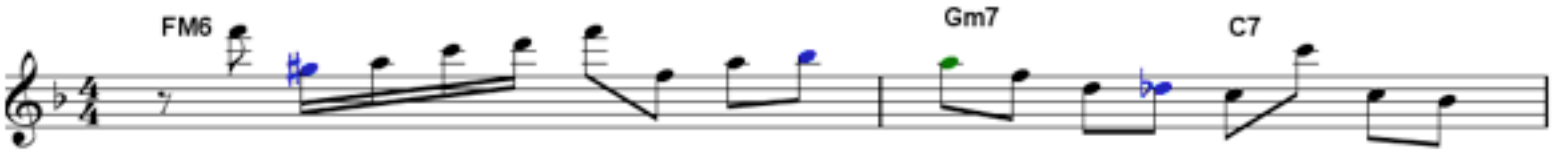
(rule (V2) (**C16 C16 C16 C16** M4) 0.05)
(rule (V2) (**C16/5 C16/5 C16/5 C16/5 C16/5** M4) 0.0050)
(rule (V2) (**C8 C8 C8 C8**) 0.3)
(rule (V2) (**C8/5 C8/5 C8/5 C8/5 C8/5**) 5.0E-4)
(rule (V4) (**C8/3 C8/3 A8/3**) 0.01)
(rule (V4) (**C8/3 C8/3 C8/3**) 0.05)
(rule (V4) (**C8/3 C8/3 C8/3**) 0.02)
(rule (V4) (N4) 0.22)
(rule (V4) (V8 V8) 0.72)
(rule (V8) (**C16 A16**) 0.01)
(rule (V8) (N8) 0.99)

Grammar Learning Feature

- Impro-Visor can **learn a grammar** by examining one or more transcribed solos.
- For greater coherence special construct called a ***slope*** is introduced, from which **melodic contours** can be constructed.
- Slopes can appear in the rules in the place of terminals.

Slopes Encode Contours

(a) Original melody:



(b) Melody contour:

(c) Abstract melody using slopes (Δ's):

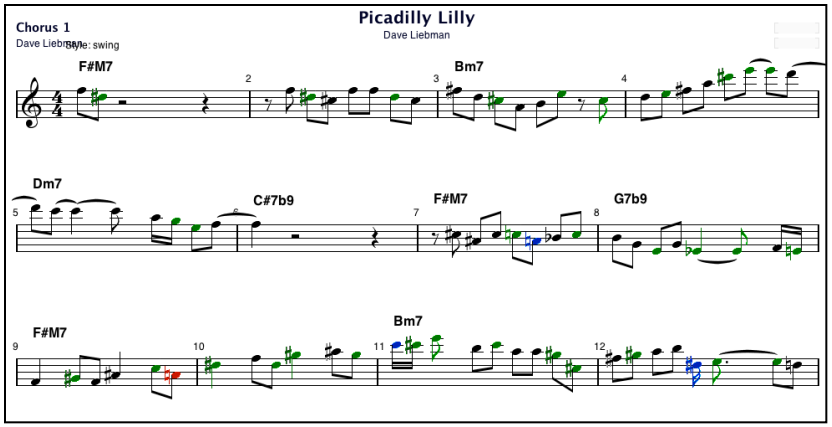
(R8 C8 (Δ -9 -9 A16) (Δ1 3 C16 C16 C16 C8) (Δ-12 -12 C8) (Δ1 4 C8 A8)

(Δ -4 -1 L8 C8 C8 A8 C8) (Δ12 12 C8) (Δ-12 -2 C8 C8))

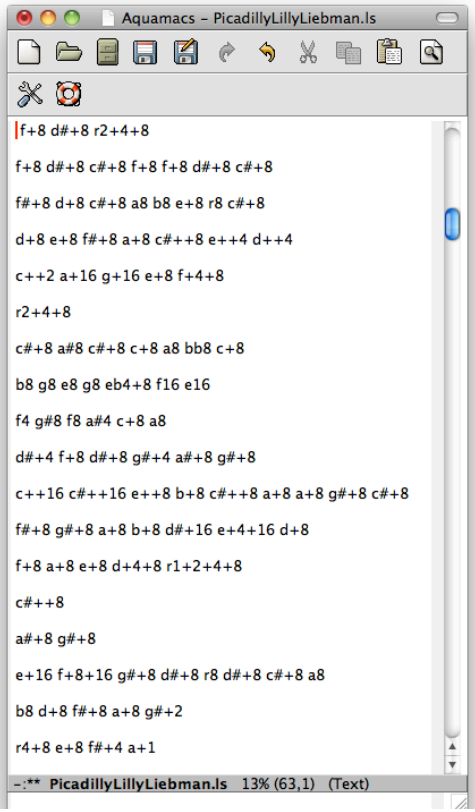
Grammar Learning Algorithm

enables grammar to be learned from transcriptions

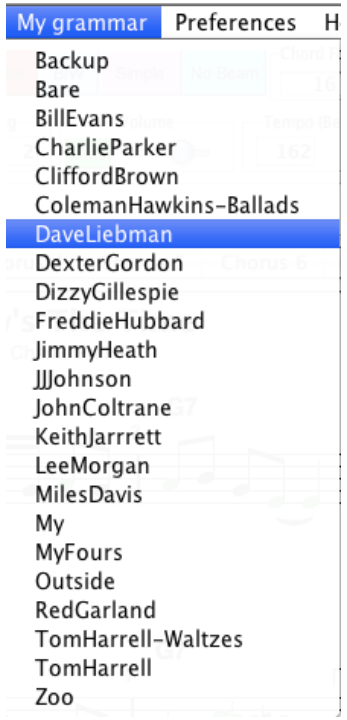
Transcription of Dave Liebman's Solo on Picadilly Lilly:



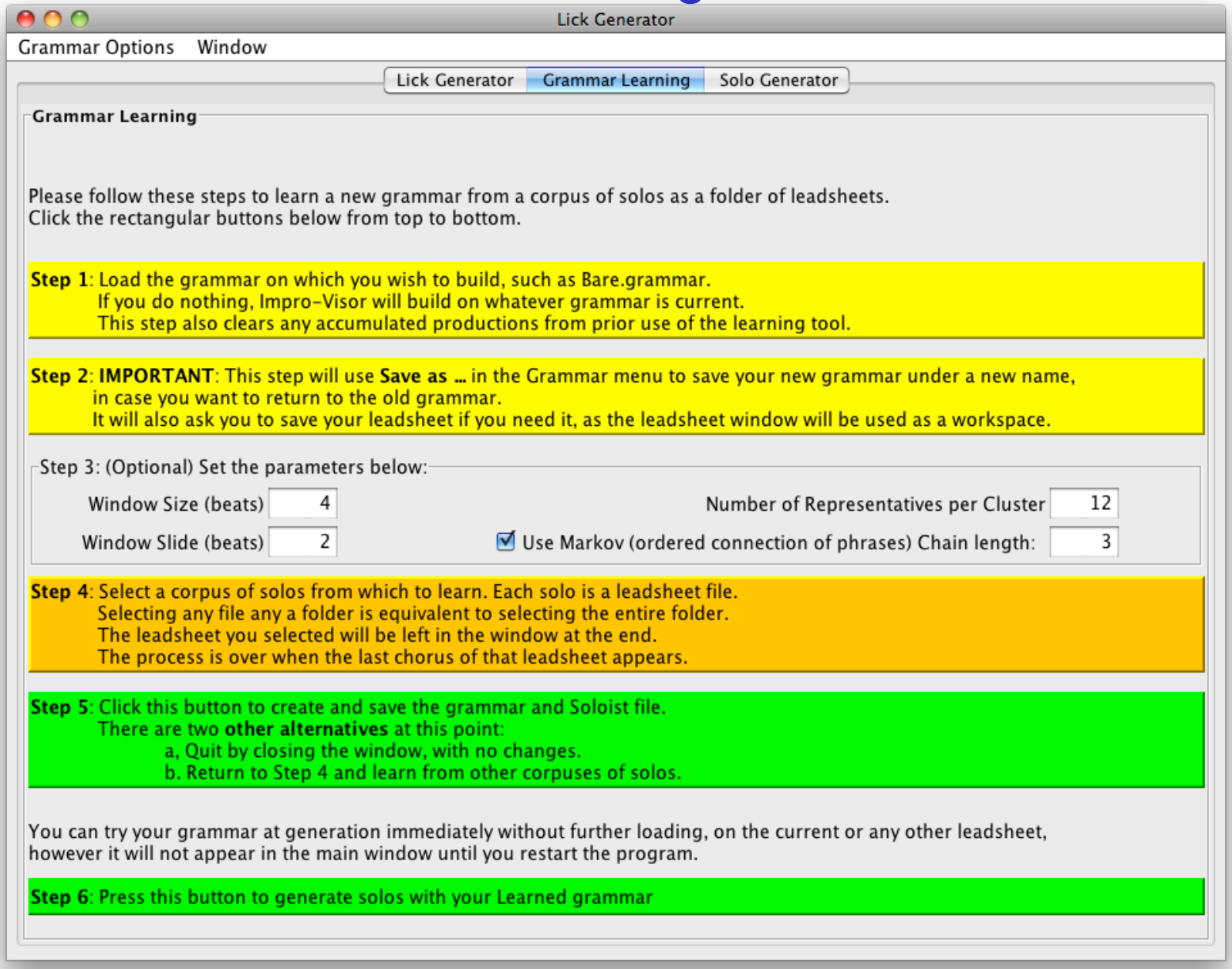
Transcription in Text



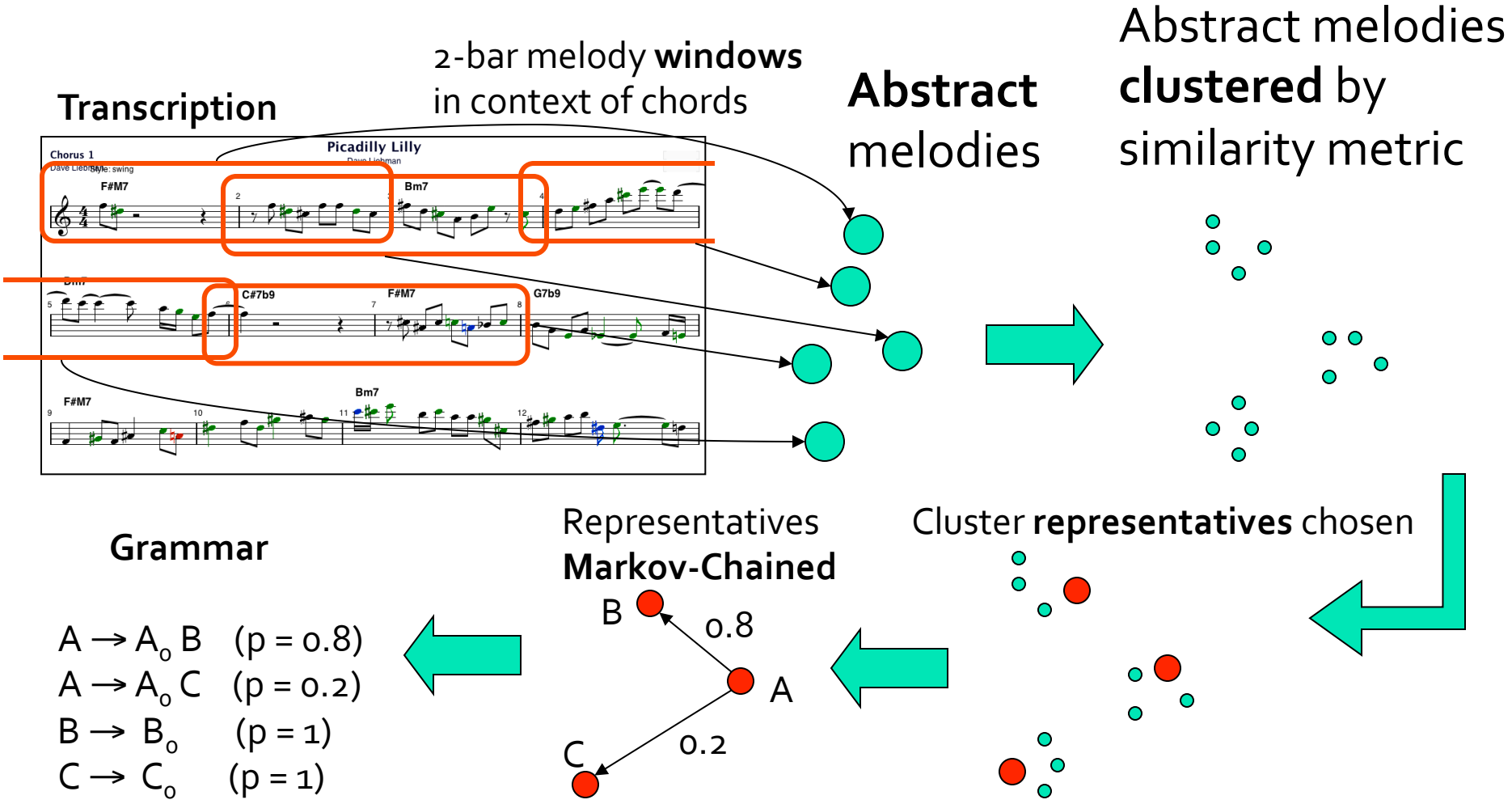
Learned Grammars



Grammar Learning Interface



From Transcription to Grammar



Example: Dave Liebman Grammar Excerpt (The full grammar is over 1000 lines)

```
(startsymbol P)
(base (P 0) () 1.0)
(rule (P Y) (Seg1 (P (- Y 120))) 1.0)
(rule (Seg1) (C4) 1.0)
(rule (P Y) ((START 1) (P (- Y 480))) 1.0)
(rule (P Y) ((START 2) (P (- Y 960))) 10.0)
(rule (P Y) ((START 4) (P (- Y 1920))) 100.0)
(rule (P Y) ((START 8) (P (- Y 3840))) 1000.0)

(rule (START Z) ((Cluster0 Z)) 0.03)
(rule (START Z) ((Cluster1 Z)) 0.02)
(rule (START Z) ((Cluster2 Z)) 0.07)
(rule (START Z) ((Cluster3 Z)) 0.08)
 . . .
(rule (START Z) ((Cluster28 Z)) 0.08)

(base (Cluster0 0) () 1)
(base (Cluster1 0) () 1)
 . . .
(base (Cluster28 0) () 1)
```

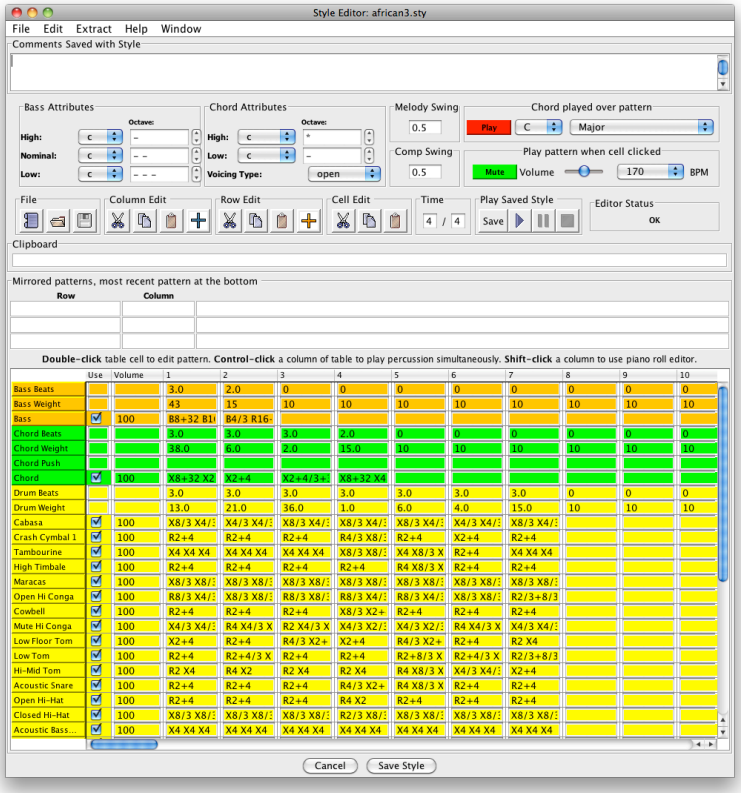
```
(base (Cluster0to3 0) () 1)
(base (Cluster0to4 0) () 1)
(base (Cluster0to11 0) () 1)
 . . .
(base (Cluster28to28 0) () 1)

(rule (Cluster0 Z) (Q0 (Cluster0to3 (- Z 1))) 0.33)
(rule (Cluster0 Z) (Q0 (Cluster0to4 (- Z 1))) 0.11)
 . . .
(rule (Cluster28to28 Z) (Q28 (Cluster28to13 (- Z 1))) 0.33)

(rule (Q0)((slope 0 0 C2)(slope -4 -4 R4+8 L8)) 0.20)
(rule (Q0)((slope 0 0 C4)(slope -2 -2 R4+8 L8)(slope 1 5 X8 A8)) 0.20)
(rule (Q0)((slope -2 -1 L2)(slope -4 -4 R4+8 L8)) 0.20)
. . .
(rule (Q28)((slope 2 4 L4+8)(slope -2 -1 A8 C8 L8 C8)(slope 2 2 L8)) 0.20)
```

Style Learning in Impro-Visor

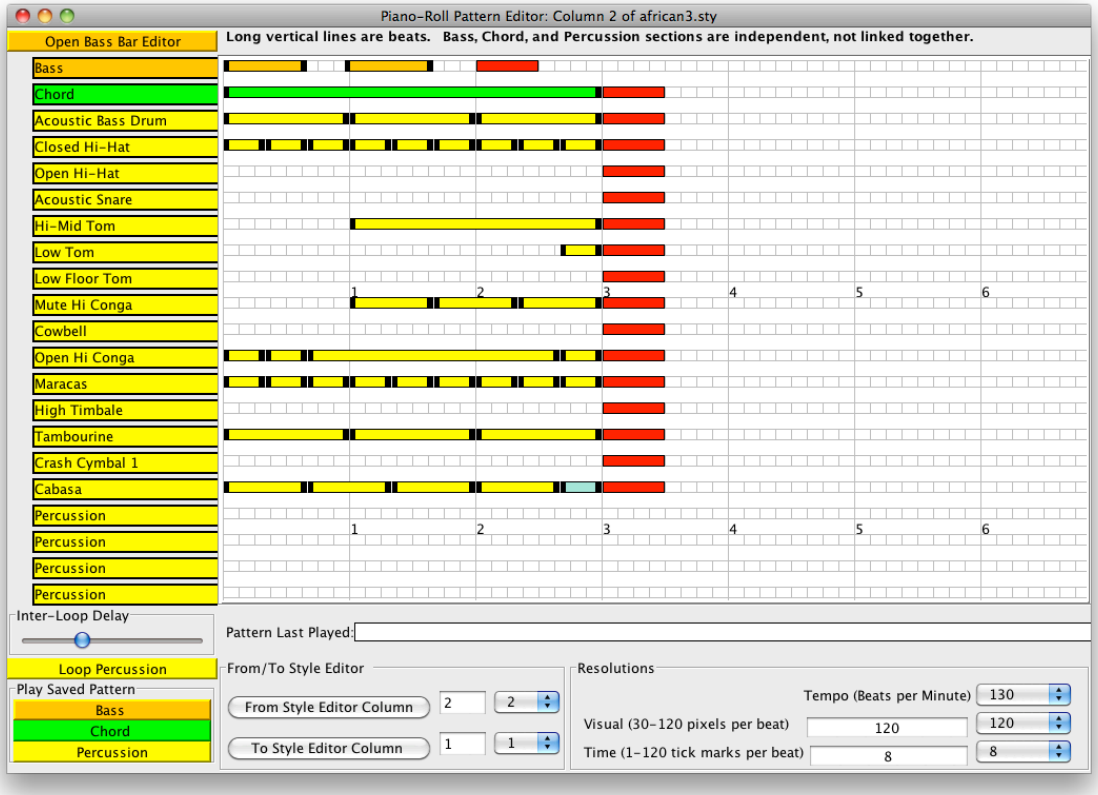
- Style Patterns are used (along with chord sequence) in creating **accompaniment**.
- Patterns are like a non-recursive grammar.
- Impro-Visor can **learn** a style specification (in its own language), given a MIDI file of a **performance** in that style.
- As with grammar learning, **clustering** is used.

Style Patterns Represented Graphically

Style Spreadsheet



"Piano roll" for one column of spreadsheet



A Different Approach to Learning
RBM-provisor

- Applies **Restricted Boltzmann Machines** (RBMs) stacked as ***Deep Belief Networks*** (Geoffrey Hinton).
- RBMs are neural networks based on **probabilities** of switching, determined by unsupervised learning of synaptic weights.
- An RBM tries to learn a set of **concepts** based on a set of input samples: melodies over chords.
- RBM's stabilize to **probability distributions** reflecting those concepts, and can generate music probabilistically, as can grammars.

Deep Belief Network Rationale

- Try to learn with **as little wired-in** musical knowledge as possible.

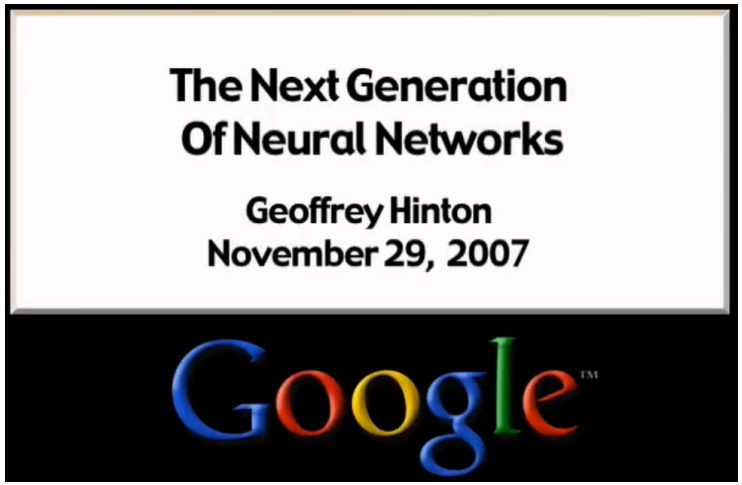
- Use probabilistic behavior of network to generate novelty.

Deep Belief Networks

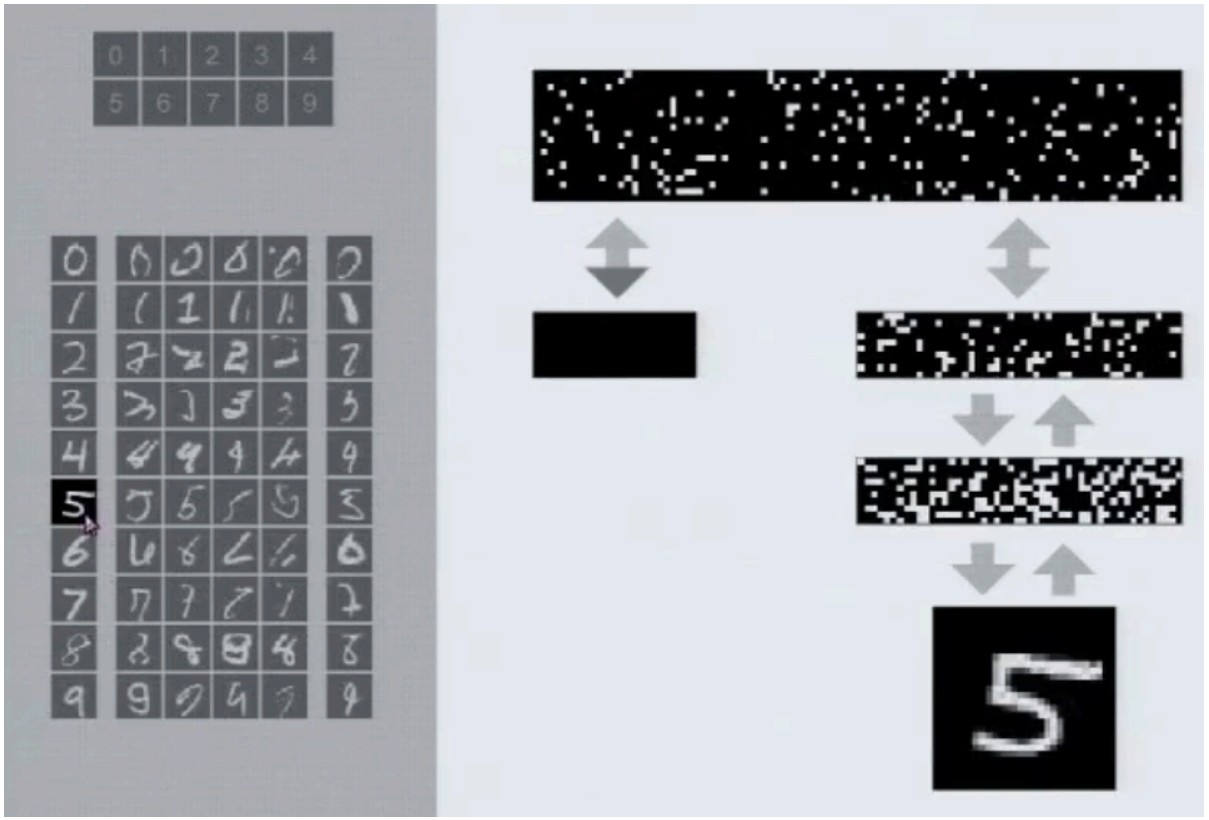
Geoffrey Hinton, U. of Toronto

- Hinton demonstrated how a stack of RBM's can learn higher order concepts sufficient to perform tasks such as digit recognition.

- We applied a similar idea to learning concepts that **produce melodies** over chord progressions.

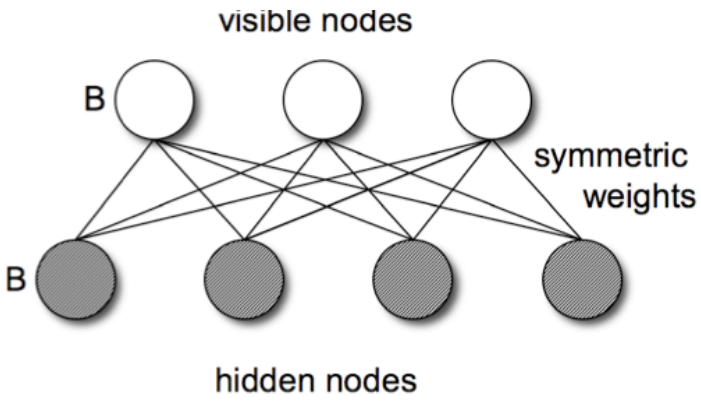


Pattern learning and generation inspired by G. Hinton

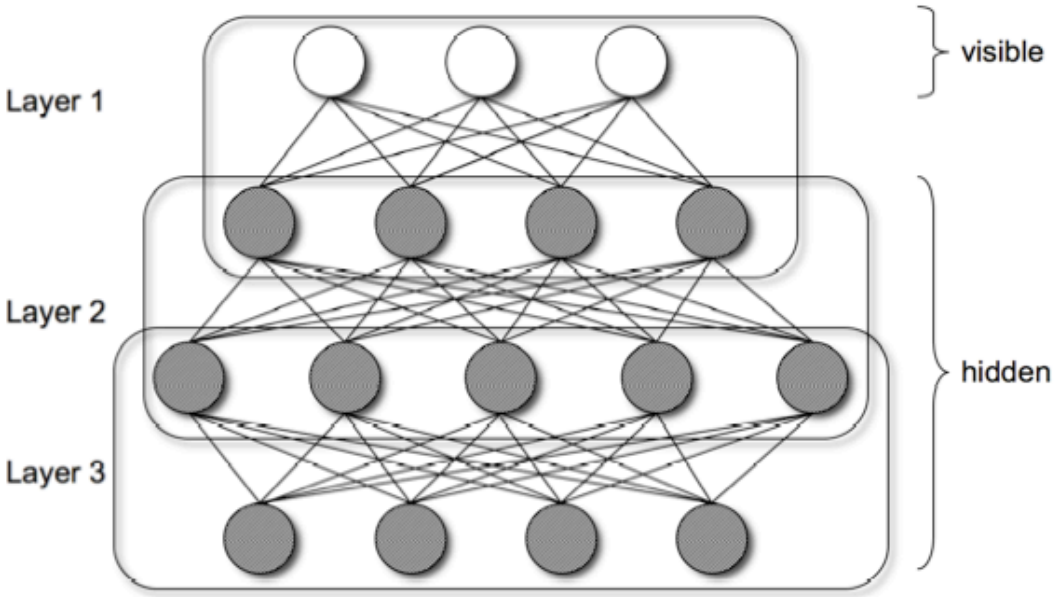


Restricted Boltzmann Machines vs. Deep Belief Networks

RBM

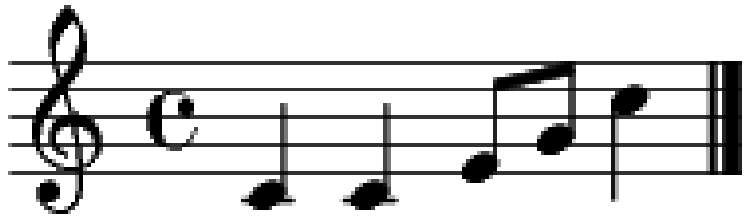


DBN (3-layer)



Deeper Concepts

Data Representation

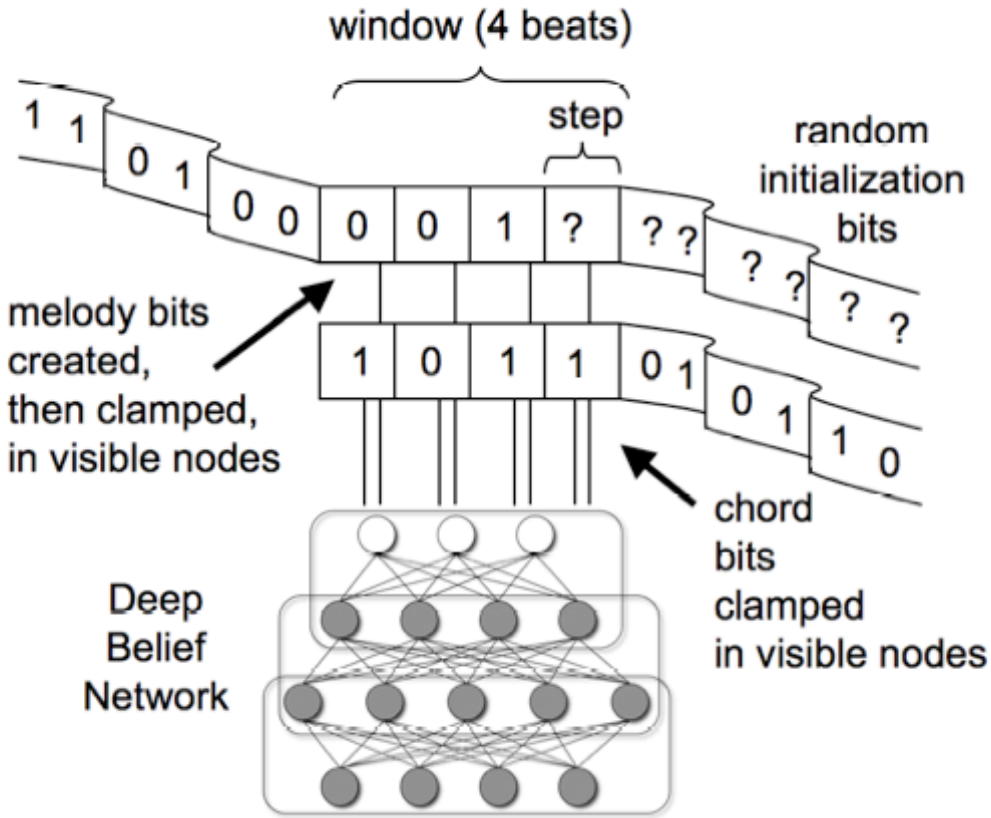


strike/rest | pitches | octave

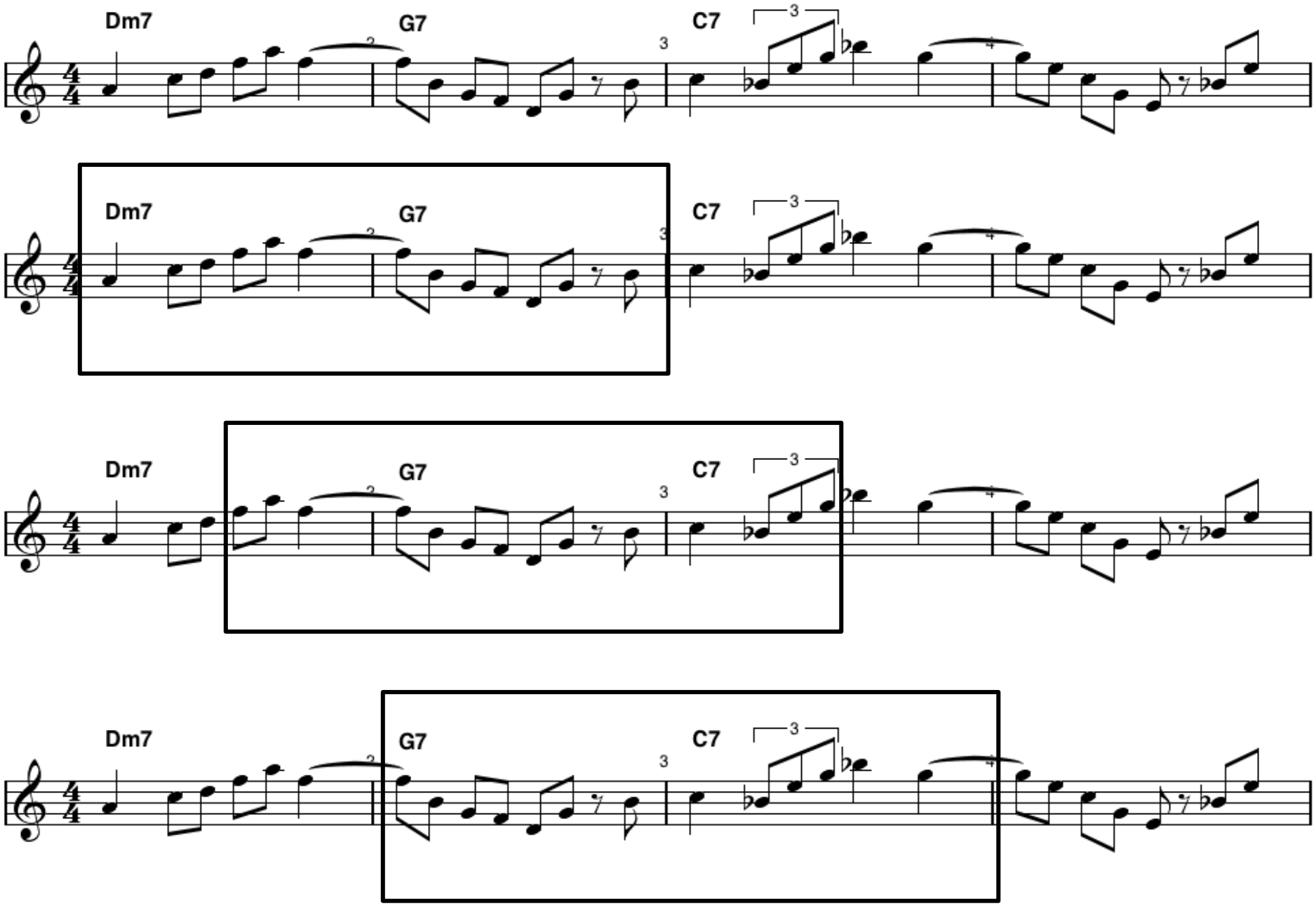
Beat	S	R	C	D	E	F	G	A	B	1	2	3	4
1	0	0	1	0	0	0	0	0	0	0	1	0	0
	1	0	0	0	0	0	0	0	0	0	1	0	0
2	0	0	1	0	0	0	0	0	0	0	1	0	0
	1	0	0	0	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	1	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	1	0	0	1	0	0
4	0	0	1	0	0	0	0	0	0	0	0	1	0
	1	0	0	0	0	0	0	0	0	0	0	1	0

Improvising Jazz with a Deep Belief Network

Melody Out ⟵ Chord Progression In

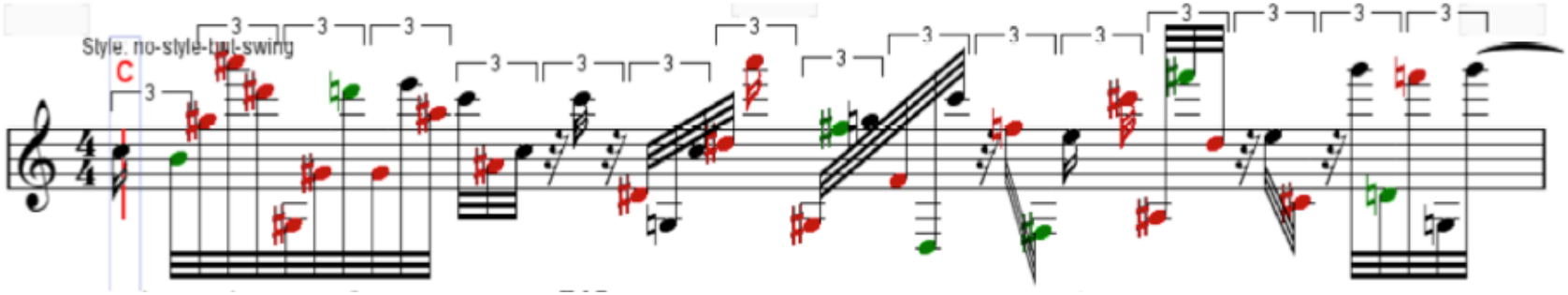


Moving Window Training Approach

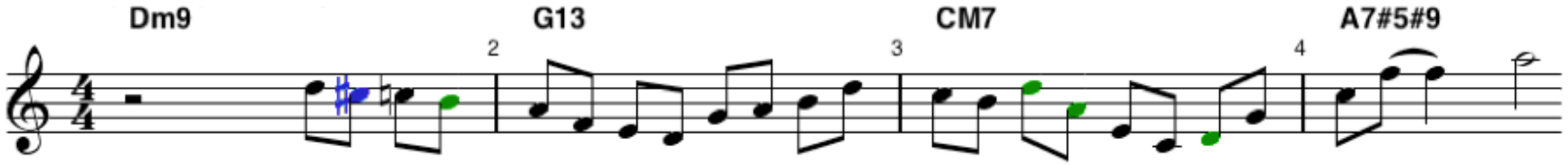


RBM-provisor Examples

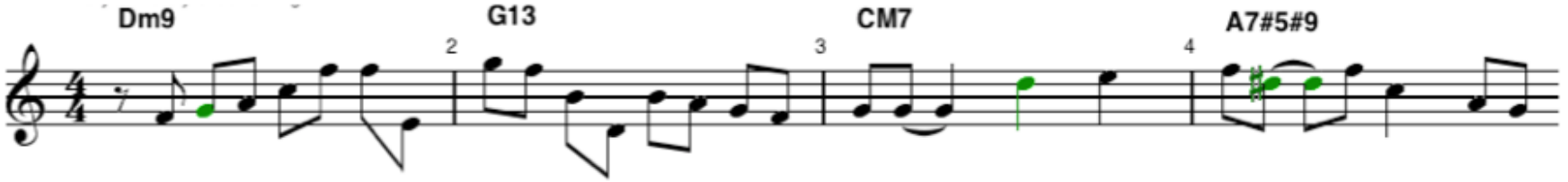
Random output from Untrained Network



Example from Training Set



Output from Trained Network



Issues with Deep Belief Approach

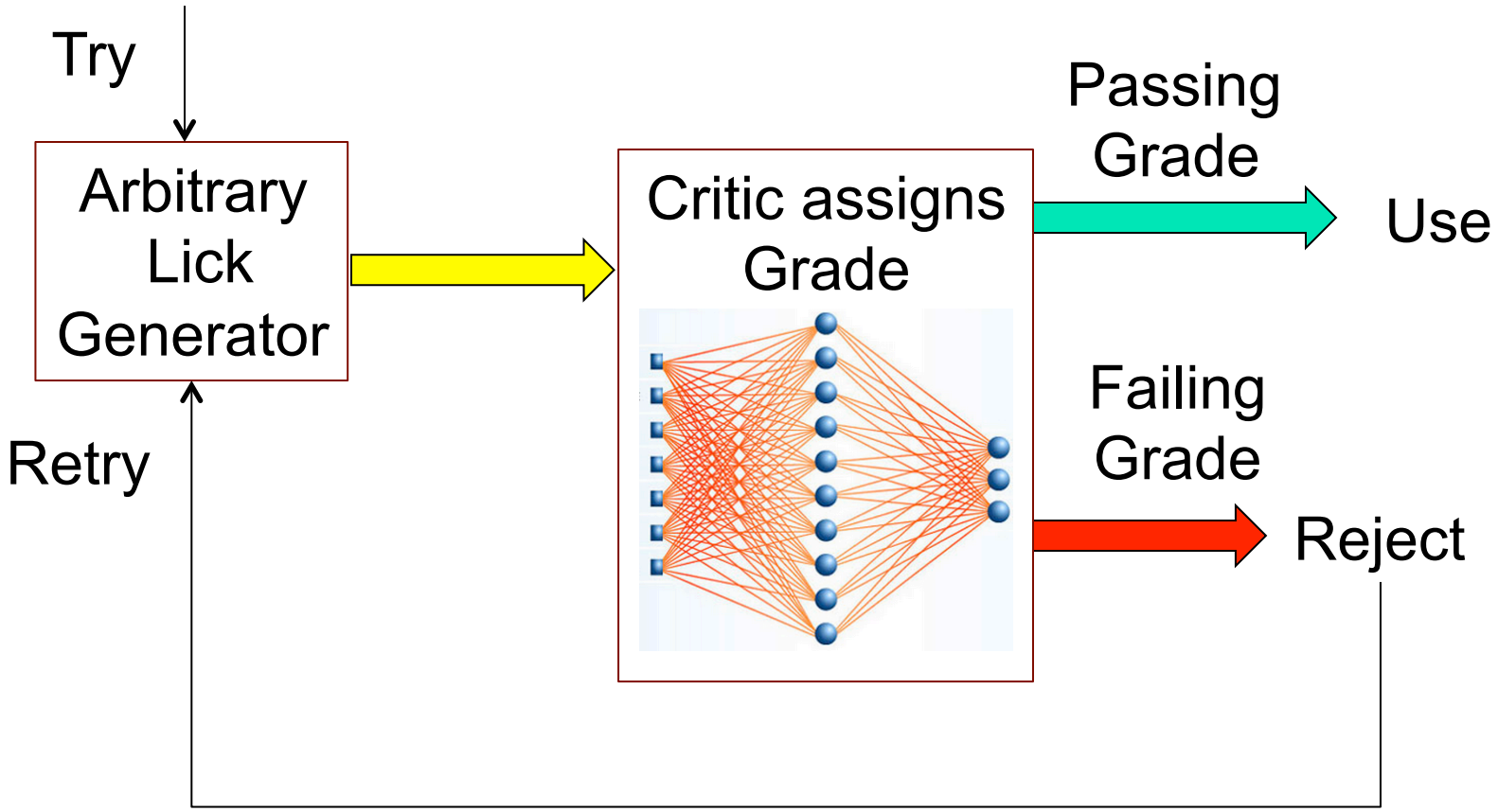
- Learning is very slow
- So far, not enough variety compared to what grammars can do.
- Thus too soon to integrate with Impro-Visor

Neural Network Critic for Improved Improvisation

Recent work by Hayden Blauzvern and the speaker

Integrated into the development version of Impro-Visor (v 6.0, not yet released)

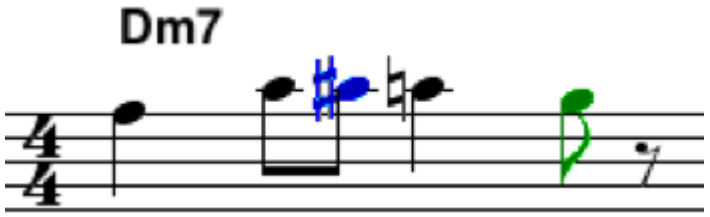
Using a Critic



New Critic Data Representation
(compared to RBM-provisor)

- Learn **abstract note categories** over chords rather than absolute pitches.
- Learn **pitch intervals** between successive notes rather than absolute pitches.
- Use **non-uniform sub-division** representation for note durations rather than uniform spacing and tie bits.
  (Partly motivated by accenting concerns.

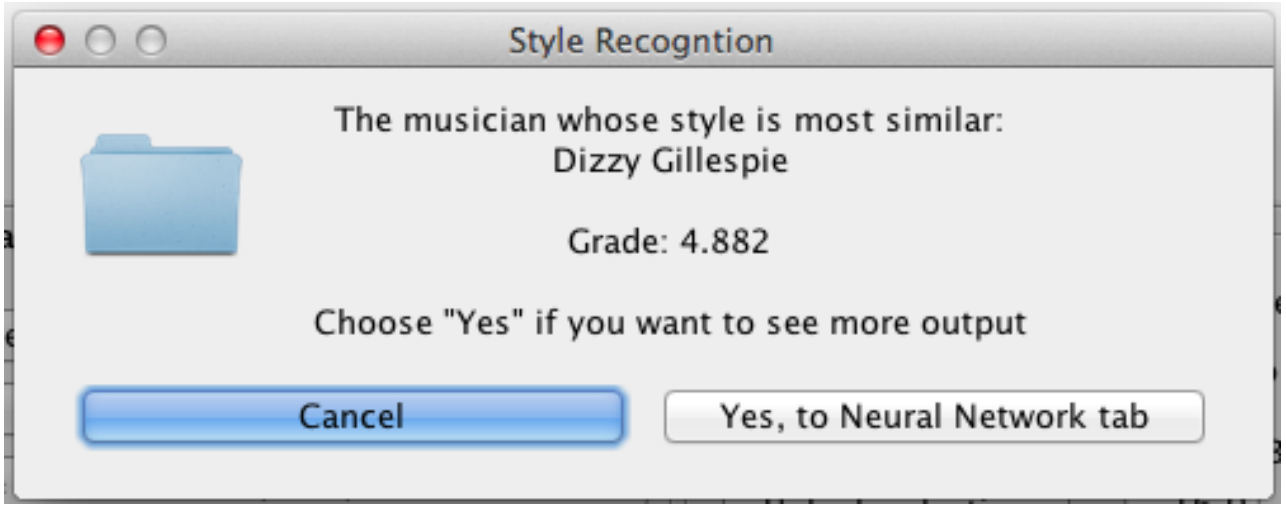
Critic Melody Note Representation



Note	Rest	Tonality	Distance	Placement	Duration
	0	1 1	0 0 0 0	1 1 1 1 1 1	0 0 1 0 0 0
	0	1 1	0 1 0 0	1 1 1 1 0 0	0 0 0 1 0 0
	0	0 1	0 0 0 1	1 1 1 0 0 0	0 0 0 1 0 0
	0	1 1	0 0 0 1	1 1 1 1 1 0	0 0 1 0 0 0
	0	1 0	0 0 1 0	1 1 1 1 0 0	0 0 0 1 0 0
	1	0 0	0 0 0 0	1 1 1 0 0 0	0 0 0 1 0 0

Style Recognition, One NN per Style

- Grammar generated solos
  - 32 measures, 16-bar blues
- 50 generations for 22 musicians
- Grade = Confidence score



Other Recent Work

- Automate analysis of **idiomatic harmonic sequences** (“chord bricks”) and key centers.
- Helps musicians understand tune construction.
- Helps players recognize the importance of key centers in improvisation.

Analyzing a Tune using Bricks

Input Chord Progression

Cm69 | / | Fm7 | / |

Dm7b5 | G7alt | Cm69 | / |

Ebm7 | Ab7 | DbM7 | / |

Dm7b5 | G7alt | Cm69 | Dm7b5 G7alt |

Impro-Visor

Output Roadmap

Join Names (yellow tags)

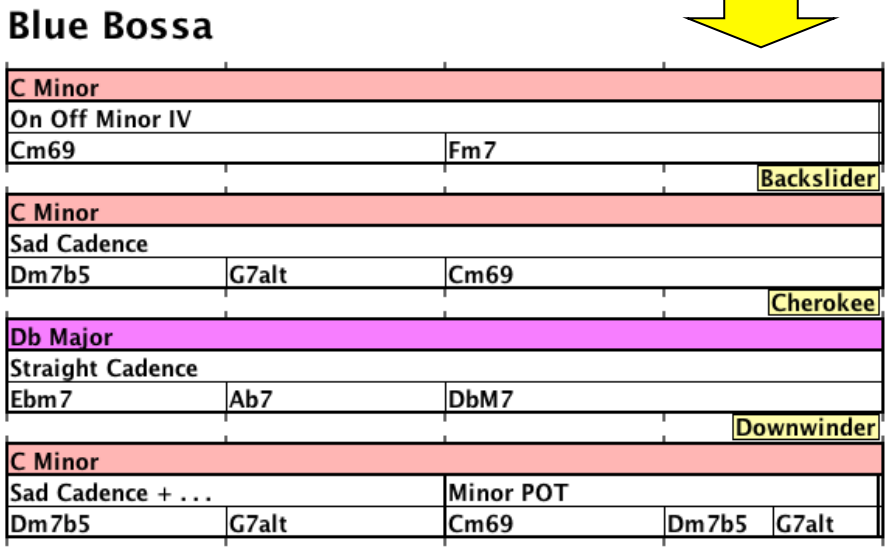
Open ML Problem:

How to learn the Brick Dictionary?

Input Chords

Inferred Key

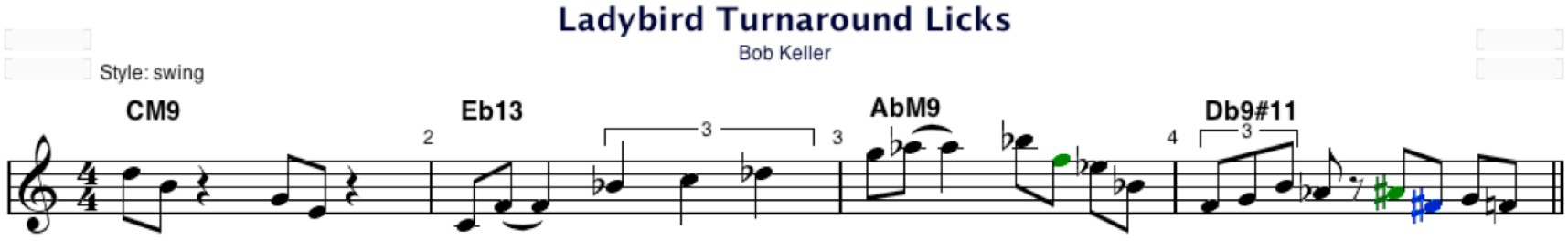
Brick Name



Other Future Work:
Bricks as a Basis for Grammar Learning

See: http://www.cs.hmc.edu/~keller/jazz/improvisor/licks/

Name	Chord Progression			
Cadence + Dropback	IIm^7	V^7	I	VI^7
Sad Cadence + Dropback	IIm^{7}b5	V^7	Im	VIm^{7}b5
POT (*Plain Old Turnaround*)	I	VI^7	IIm^7	V^7
Minor POT (*Minor Plain Old Turnaround*)	Im	VI^7	IIm^{7}b5	V^7
Pullback	IIm^7	V^7	IIIm^7	VI^7
Ladybird Turnaround	I	bIII^7	bVI	bII^7



Concluding

- Unsupervised learning:
  - Clustering to produce Grammars
  - Deep-Belief Networks (stacked RBMs)
- Supervised learning:
  - Training an MLP Melody Critic

Some References

- Gillick, Tang, and Keller, **Machine Learning of Jazz Grammars**, *Computer Music Journal*, 34:3, pp. 56–66, Fall 2010, MIT Press.

- Bickerman, Bosley, Swire, and Keller, **Learning to Create Jazz Melodies Using Deep Belief Nets**, *Proc. First International Conference on Computational Creativity*, 228-237, January, 2010.

- Keller, Schofield, Toman-Yih, Merritt, **A Creative Improvisation Companion Base on Idiomatic Harmonic Bricks**, Proc. Third ICCC, June, 2012.

- Keller, Schofield, Toman-Yih, Merritt, and Elliott, **Automating the Explanation of Jazz Chord Progressions Using Idiomatic Analysis**, *Computer Music Journal*, 37:4, pp. 54–69, Winter 2013, MIT Press.