The Effect of Symbolic Representation Design on Notions of Difference Between Musical Scores

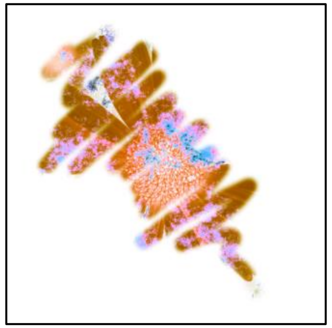
Tim de Reuse
Distributed Digital Music Archives and Libraries Lab
McGill University, Montreal

Sequences in London, May 12-13
Goldsmiths, University of London

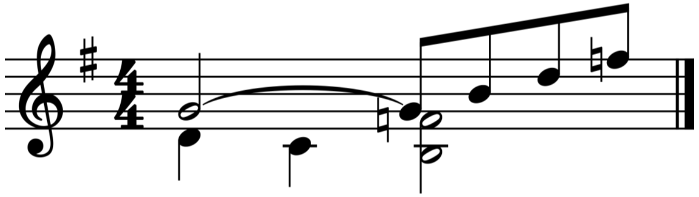
What is a musical difference (for polyphonic music)?

Original image

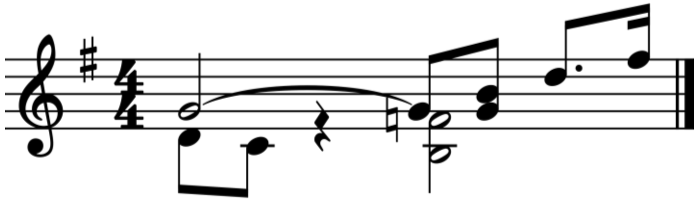
Image with noise added



Pixel-wise difference



Original score



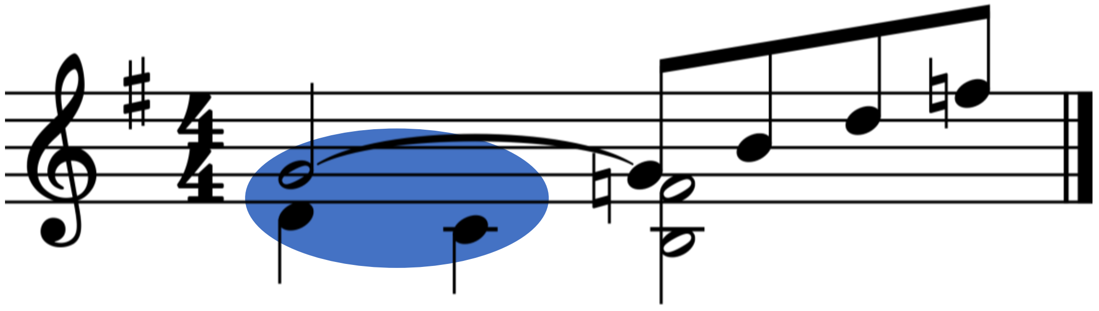
Score with errors

????

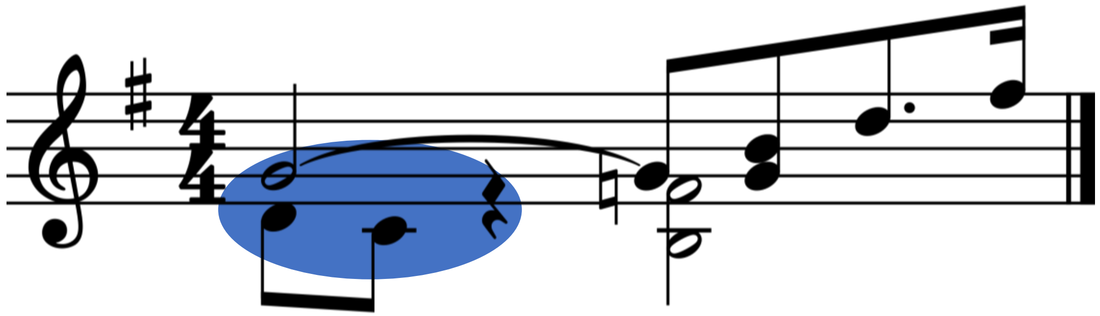
What is a musical difference?

How many differences are there between these two scores, and **what kinds?**

Depends on **underlying representation**



J.S. Bach, Air on the G String, Orchestral Suite No.3 in D Major, arr. for solo Piano. Measure 32.



As above, but manually edited to include errors.

Representation matters

- Pattern matching – what kinds of patterns are matched?
- Evaluation metrics
  - How accurate is an Optical Music Recognition algorithm?
  - Precision, Recall, etc. – how to interpret?
- Machine learning on symbolic music
  - Longer sequences: high GPU memory usage
  - Large vocabulary: rare token problem
  - Loss functions between two musical sequences
    - Imbalanced class ratio alters training characteristics

How do I choose a representation that…

Represents musical differences succinctly?

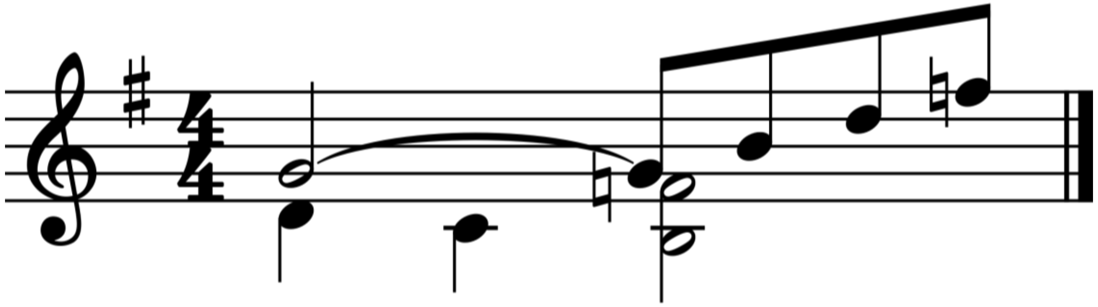
Is suitable for machine learning?

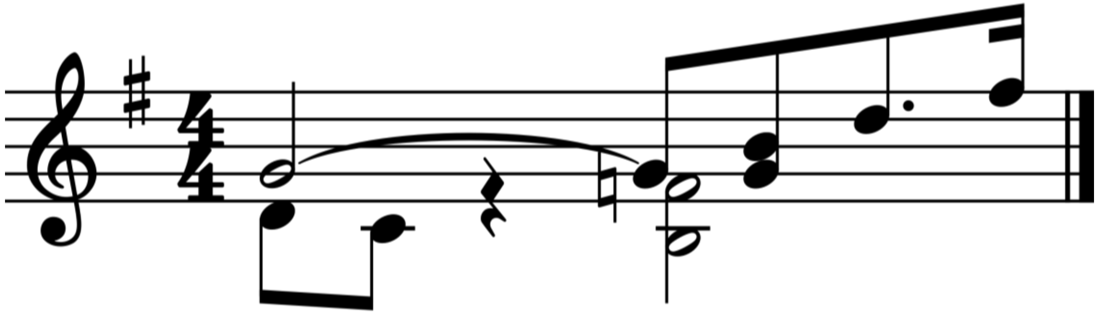
Doesn’t need a huge vocabulary of possible tokens?

Can encode a polyphonic score without loss of information?

A demonstration

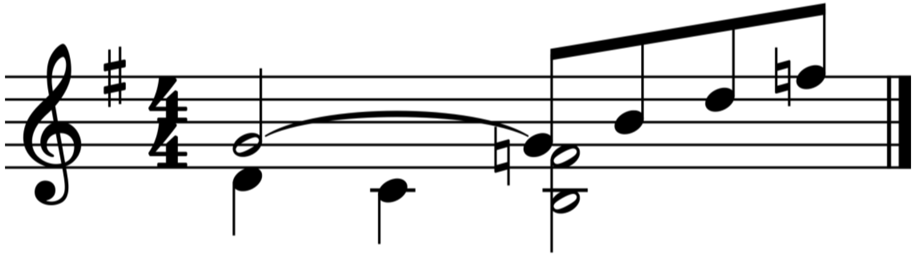
- I will define four possible string-like musical representations
- **How does the difference between these two scores change under different representations?**





#1: NoteTuple Representation

- Represent a score as a sequence of triples: (`type, delta, duration`)
    - "delta" is time until next onset
    - Chords = consecutive elements with delta 0
- 12 tokens to encode example measure

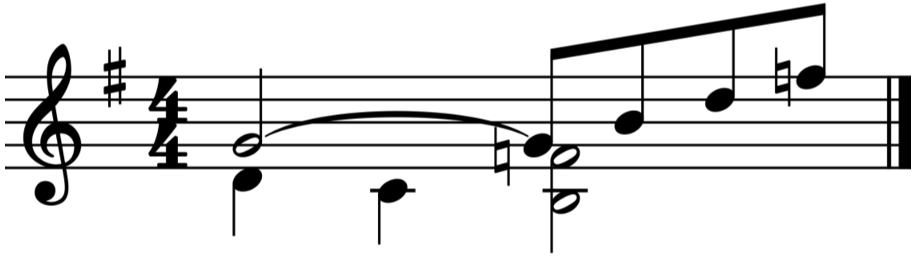


Encoding of example measure

```
(treble, 0.0, 0)
(keysig_1sharp, 0.0, 0)
(timesig_4/4, 0.0, 0)
(D4, 0.0, 1.0)
(G4, 1.0, 2.5)
(C4, 1.0, 1.0)
(B3, 0, 2.0)
(F4, 0, 2.0)
(B4, 0.5, 0.5)
(D5, 0.5, 0.5)
(F5, 0.5, 0.5)
(bar_final, 0.0, 0)
```

#2: MIDI-like Representation

- Sequence of triples:
  (`type, delta, on/off`)
  - Uses on and off “events” instead of durations
  - Like MIDI, but including clefs, time sigs, etc.
- 22 tokens to encode example measure



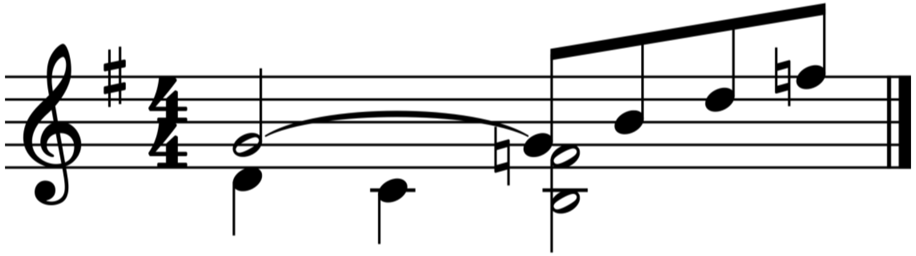
Encoding of example measure

```
(treble, 0.0, instant)
(keysig_1sharp, 0.0, instant)
(timesig_4/4, 0.0, instant)
(D4, 0.0, on)
(G4, 1.0, on)
(D4, 0.0, off)
(C4, 1.0, on)
(C4, 0.0, off)
(G4, 0.0, off)
(B3, 0.0, on)
(F4, 0.0, on)
. . .
```

#3: Event-Like Representation

- Separate deltas, notes, and on/off status into separate events
- Very verbose
    - E.g., A single eighth note encoded by:

    `notes_on, C4, delta_0.25, notes_off, C4`
- 41 tokens to encode example measure



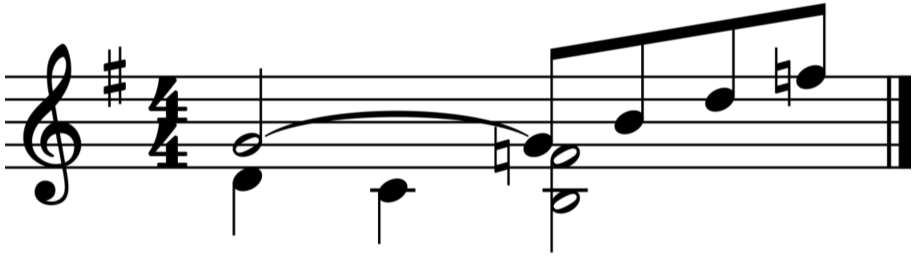
Encoding of example measure

```
treble
keysig_1sharp
timesig_4/4
notes_on
D4
G4
delta_1.0
notes_off
D4
notes_on
C4
delta_1.0
notes_off
C4
notes_on
   . . .
```

C. Hawthorne et al., “Sequence-to-Sequence Piano Transcription with Transformers.” in Proc. of the 22nd Int. Society for Music Information Retrieval Conf., Online, 2021.

#4: Agnostic Representation

- Literal reading of score
  - List glyphs from left to right, bottom to top
  - Not pitches, but staff positions
    - Lowest staff line = `pos1`
  - Add ^ token between notes in chord
- 20 tokens to encode example score



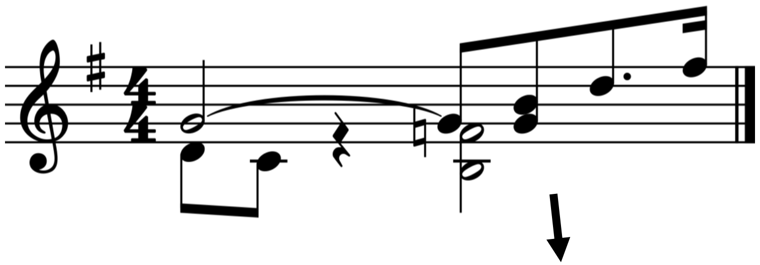
Encoding of example measure

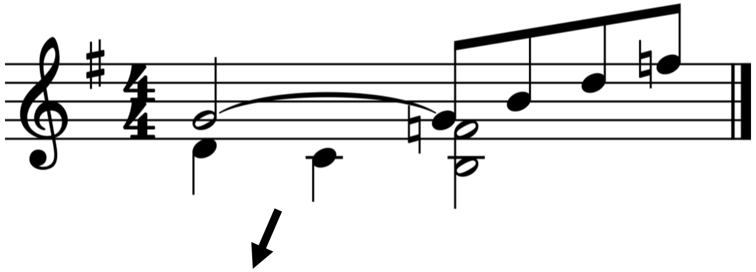
```
clef.treble
accid.sharp.pos9
timeSig.4/4
quarter.noBeam.down.pos-1
^
half.noBeam.up.pos2
tie.start.pos2
quarter.noBeam.down.pos-2
accid.natural.pos1
tie.end.pos2
half.noBeam.down.pos-3
^
half.noBeam.down.pos1
           ...
```

Sequence Alignment

- Needleman-Wunsch algorithm to define a difference operation
- Gives **minimal list of operations** to transform one sequence into another
    - Insertion, Deletion, or Replacement

Sequence Alignment





Measure with Errors	Original Measure
(treble, 0.0, 0)	(treble, 0.0, 0)
(keysig_1sharp, 0.0, 0)	(keysig_1sharp, 0.0, 0)
(timesig_4/4, 0.0, 0)	(timesig_4/4, 0.0, 0)
(D4, 0.0, 0.5)	_
(G4, 0.5, 2.5)	(D4, 0.0, 1.0)
(C4, 0.5, 0.5)	(G4, 1.0, 2.5)
(rest, 1.0, 1.0)	(C4, 1.0, 1.0)
(B3, 0, 2.0)	(B3, 0, 2.0)
(F4, 0, 2.0)	(F4, 0, 2.0)
(G4, 0, 0.5)	_
(B4, 0, 0.5)	(B4, 0.5, 0.5)
(D5, 0.75, 0.75)	(D5, 0.5, 0.5)
(F#5, 0.25, 0.25)	(F5, 0.5, 0.5)
(bar_final, 0.0, 0)	(bar_final, 0.0, 0)

Comparison between alignments

- Each type of representation:
  - Different encoding length
  - Different # operations necessary to correct
- Error Rate: # operations necessary / Total # of tokens

	Length of Encoding	Operations to Correct	Error Rate
NoteTuple	12	8	67%

Comparison between alignments

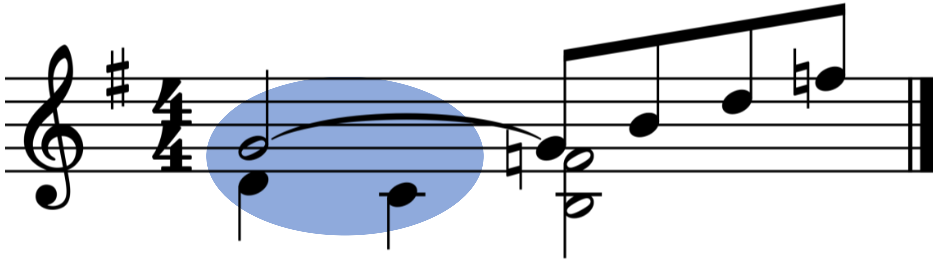
- Each representation causes the alignment to prescribe **different types of operations**

				Percentage of operations that are a...		
	Length of Encoding	Operations to Correct	Error Rate	Replacement	Insertion	Deletion
NoteTuple	**12**	8	67%	75%	25%	0%
MIDI-Like	27	11	41%	64%	36%	0%
Event-Like	48	16	**33%**	44%	50%	6%
Agnostic	22	**8**	36%	63%	38%	0%

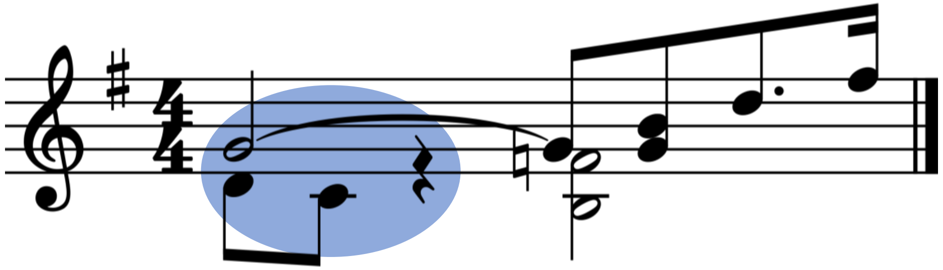
An example

- In the NoteTuple encoding, all the circled events are erroneous

Measure with Errors	Original Measure	Operation
	...	
(D4, 0.0, 0.5)	_	Delete
(G4, 0.5, 2.5)	(D4, 0.0, 1.0)	Replace
(C4, 0.5, 0.5)	(G4, 1.0, 2.5)	Replace
(rest, 1.0, 1.0)	(C4, 1.0, 1.0)	Replace
	...	



Original Measure

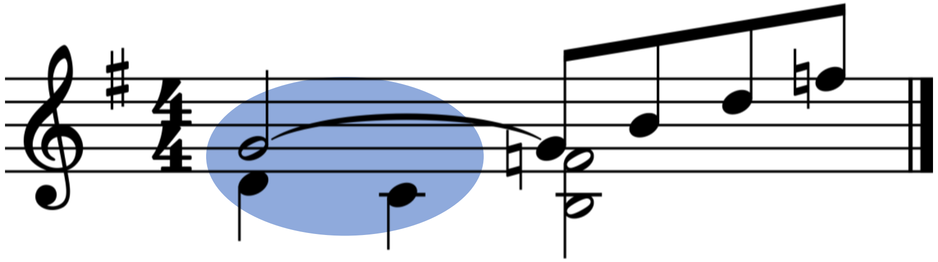


Measure with Errors

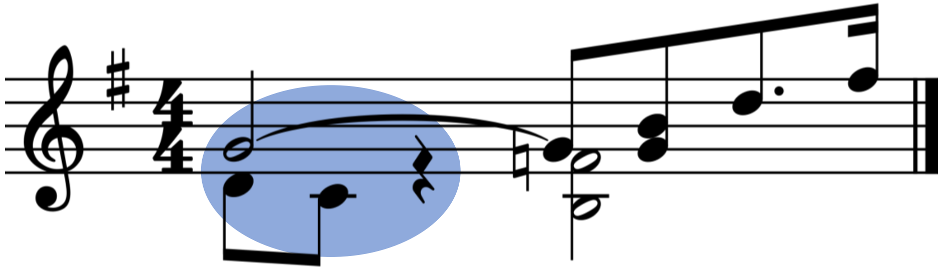
An example

- In the Event-like encoding, only the **deltas between notes** are erroneous; it's **mostly correct**

Measure with Errors	Original Measure	Operation
	...	
notes_on	notes_on	
D4	D4	
G4	G4	
delta_0.5	delta_1.0	Replace
notes_off	notes_off	
D4	D4	
delta_0.5	delta_1.0	Replace
C4	C4	
delta_1.0	_	Delete
notes_off	notes_off	
	...	



Original Measure



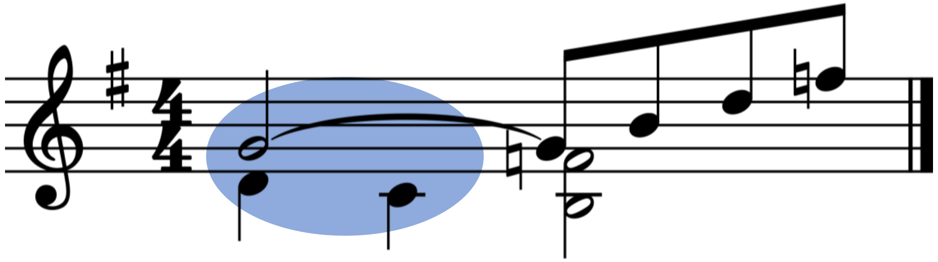
Measure with Errors

An example

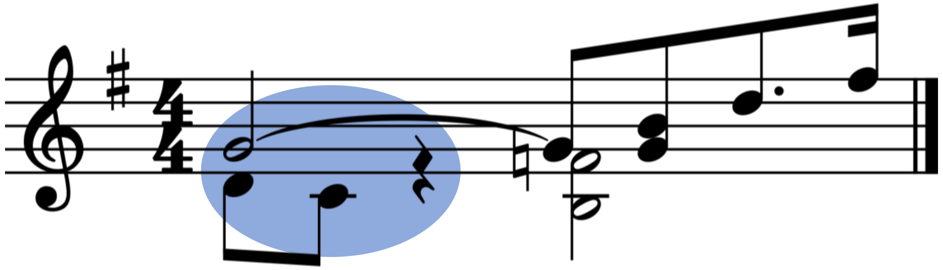
- The MIDI-Like encoding is in-between
  - Mostly incorrect, but some beginnings / ends of notes line up

Measure with Errors	Original Measure	Operation
	...	
(D4, 0.0, on)	(D4, 0.0, on)	
(G4, 0.5, on)	(G4, 1.0, on)	Replace
(D4, 0.0, off)	(D4, 0.0, off)	
(C4, 0.5, on)	(C4, 1.0, on)	Replace
(C4, 1.0, off)	(C4, 0.0, off)	Replace
(G4, 0.0, off)	(G4, 0.0, off)	
	...	

- Agnostic encoding behaves similarly



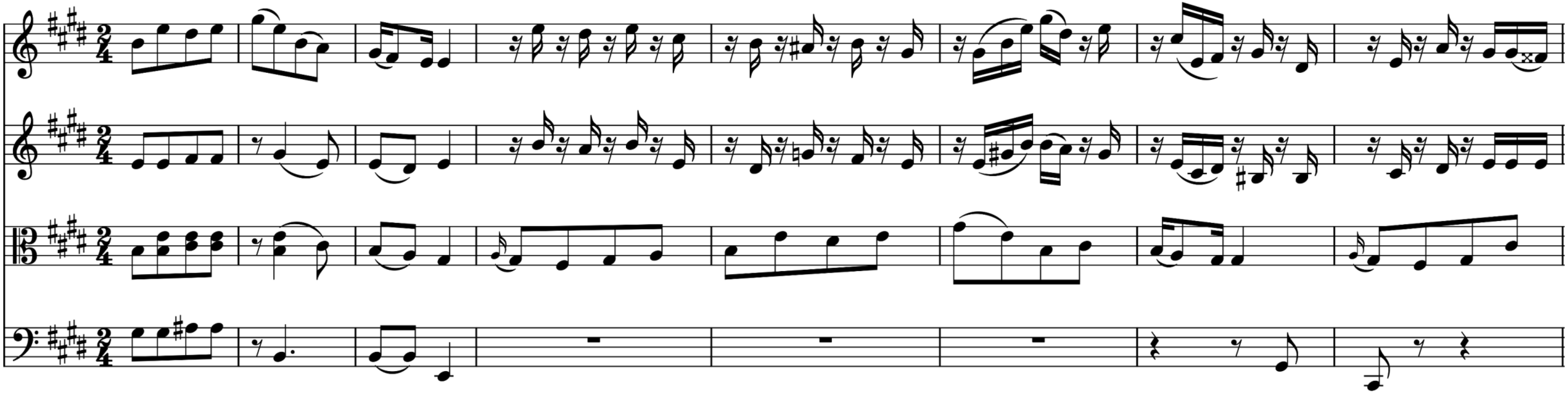
Original Measure



Measure with Errors

Using this method on a larger scale

- Now I perform this process on a whole string quartet movement:
  - One version corrected and reviewed by humans
  - One version the result of using Optical Music Recognition on a .pdf score



Felix Mendelssohn, Op 14, Mvt. 4: Andante in E Major for String Quartet, mm. 18-25

Comparison on String Quartet

	Vocabulary Size	Length of Encoding	Operations to Correct	Error Rate	Percentage of operations that are a…		
					Replacement	Insertion	Deletion
NoteTuple	592	**3897**	**753**	19%	53%	18%	29%
MIDI-Like	514	7537	2790	37%	32%	31%	38%
Event-Like	**144**	11865	3735	31%	23%	27%	50%
Agnostic	413	6498	1197	**18%**	26%	30%	44%

- NoteTuple has high vocabulary size (num. of possible tokens)
- MIDI-Like, Event-like: long encodings and high error rates
  - Event-Like has tiny vocabulary

Comparison on (a different) String Quartet

(Felix Mendelssohn, String Quartet in E-flat Major Op 12, Mvt. 4)

	Vocabulary	Length of	Operations to		Percentage of operations that are a…		
	Size	Encoding	Correct	Error Rate	Replacement	Insertion	Deletion
NoteTuple	844	**7548**	**1973**	**26%**	39%	23%	38%
MIDI-Like	650	12517	9323	74%	33%	37%	29%
Event-Like	**134**	21306	11829	56%	31%	37%	32%
Agnostic	477	15940	5263	33%	22%	38%	40%

- Similar statistics whenever errors are created by Optical Music Recognition

Closing Thoughts

- I chose the agnostic encoding for my “spellchecker”
  - Low error rate
  - Not too verbose
  - Not many rare words
  - Saw performance boost when I switched from NoteTuple
- But: this is only best for **the corpus I’m using** and the **problem I’m solving!**

Closing Thoughts

- Takeaway: when working with scores, **different representations emphasize different parts of the musical surface**
  - Consider: loss functions, alignments, evaluation
- Different representations admit concise representations of different types of errors
- Using a different representation can alter performance on common tasks
  - Event-Like rep. used by Hawthorne et al. (ISMIR 2021) for automatic music transcription

Thank you!

timothy.dereuse@mail.mcgill.ca

McGill

Schulich School of Music
École de musique Schulich

DDMAL DISTRIBUTED DIGITAL MUSIC ARCHIVES & LIBRARIES LAB

Fonds de recherche
Société et culture
Québec

Social Sciences and Humanities
Research Council of Canada

Conseil de recherches en
sciences humaines du Canada

Canada