6. Transformer Tokenizers

Generative Algorithms for Sound and Music

Why I am teaching this class

- Spent a whole year dissecting and reimplementing MIDITok
- I <3 u guys

What is a tokenizer?

- Translation layer between the transformer and its modality
- Encodes
- Decodes
- Is invertible*

Some terms

- Vocabulary
  - List of all recognized tokens
- Encode / Decode
  - Move from medium to tokens, and back.
- Token
  - Discrete element of information
- Id
  - Associated integer to a token

Why do we need a tokenizer anyway?

- Transformers operate on discrete data
- We need a reliable, reversible way to encode
- We don’t have infinite context lengths or resources
  - Need to be efficient and concise


LEFT
EXIT 12
Music
Text
imgflip.com

Tokenizers in NLP (Text-based models)

- The “foundation” is the set of all characters
- There are different philosophies and approaches to this problem

Tokenizer Philosophies in NLP

- Dictionary-based
- Character-based
- Byte-based
- Rule-based: Split dataset given pre-defined rules.
- Statistics-based: Start with a set of base tokens, create groupings based on recurrent patterns

Statistics-based tokenization in NLP

Start with a base vocabulary, add new tokens based on recurrent patterns

>>> What is a good starter set?

- Byte Pair Encoding
- WordPiece
- Any other methods?

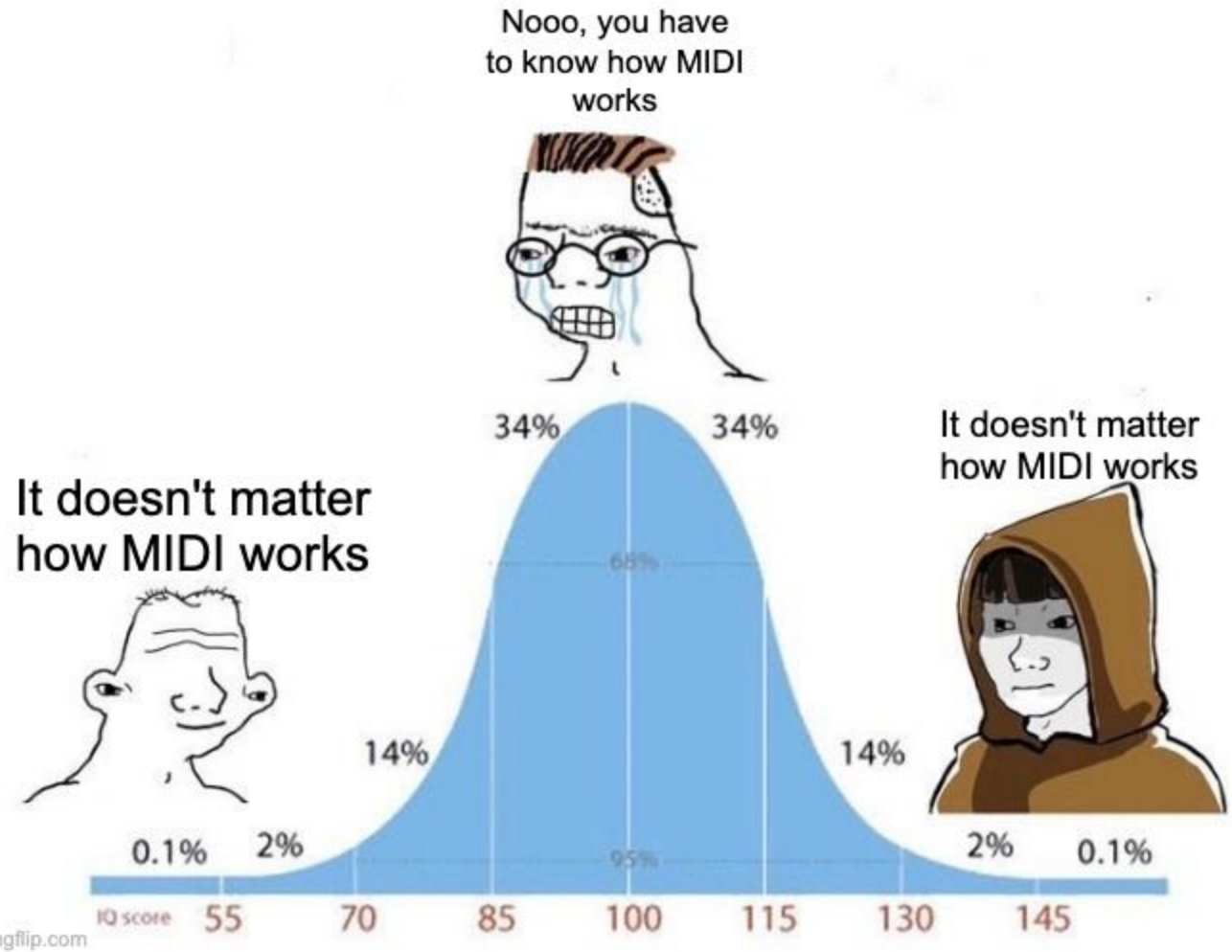
Text Tokenizers

Music Tokenizers

What musical information do we want to tokenize?

Initial Considerations

- Balance between available data, token variety, vocabulary size
- Purpose of model: What are we generating?


Nooo, you have to know how MIDI works
34%
34%
It doesn't matter how MIDI works
It doesn't matter how MIDI works
68%
14%
14%
95%
0.1%
2%
2%
0.1%
IQ score
55
70
85
100
115
130
145
imgflip.com

A History of MIDI

- Started as a communication between Synthesisers (1983)
- Need for saving files came later!
- MIDI file standard is more of a guideline…

MIDI files - Header

- Defines NumTracks, Division
- Division = Ticks per quarter note
  - Typical values: 384, 480, 960, 1920… (Why?)

MIDI files - Events

- Note Event: On / Off, Channel, Pitch, Velocity
- CC messages: Channel, ID, Value
- Ticks: Time in ticks between next event and previous
- Meta Events: TimeSig, Tempo, KeySig, Or even Text
- Track Events
- SysEx Events

MIDI files - What’s not in it?

- Explicit bar numbers
- Beat emphasis
- Phrasing information

All meta events are optional - Cannot assume they are always present.

What you need to know about MIDI files

- Two types: Performance, and Score based
- Notes are encoded with NoteOn and NoteOff messages
- Events are separated by Ticks
- Can include CC messages and even text

MIDI files - Implications for Tokenizers

- Don’t have necessarily the information we want
- Need to consider the quality of the files, not just quantity
- Tick-based timing needs processing
- May have information we do not want!

Tokenizing notes

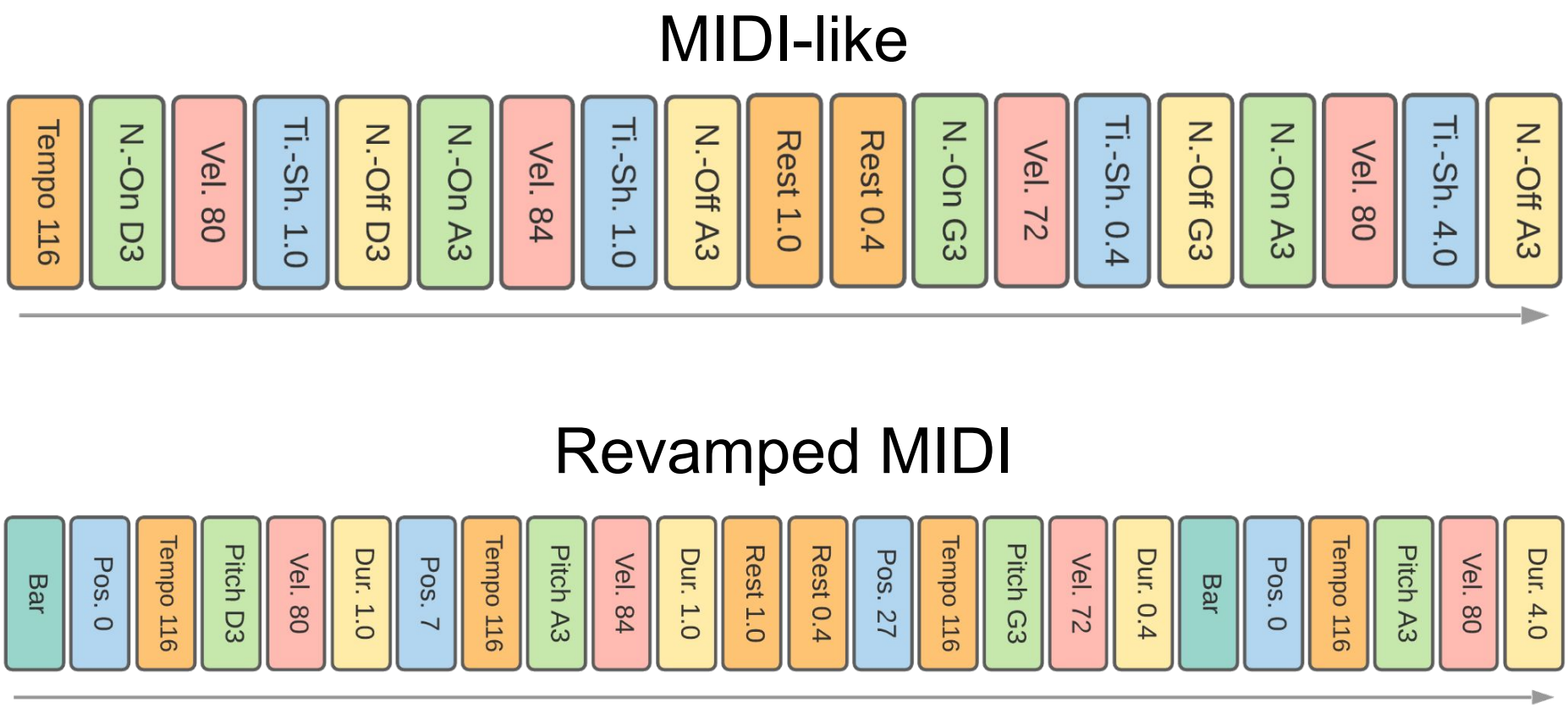
- Crucially, Notes have a pitch
- Definite Pitch vs Pitch Intervals?
- Duration vs TimeShift + NoteOff?
- Velocity Token?
- Chord Token?

Tokenizing time

- Duration
- Rest
- TimeShift
- Position
- Bar
- MicroTiming?

Tokenizing Meta Events

- Tempo
- Time Signatures
- PitchBends
- Pedals
- Program
- Track
- CC?


MIDI-like
Tempo 116
N.-On D3
Vel. 80
Ti.-Sh. 1.0
N.-Off D3
N.-On A3
Vel. 84
Ti.-Sh. 1.0
N.-Off A3
Rest 1.0
Rest 0.4
N.-On G3
Vel. 72
Ti.-Sh. 0.4
N.-Off G3
N.-On A3
Vel. 80
Ti.-Sh. 4.0
N.-Off A3
Revamped MIDI
Bar
Pos. 0
Tempo 116
Pitch D3
Vel. 80
Dur. 1.0
Pos. 7
Tempo 116
Pitch A3
Vel. 84
Dur. 1.0
Rest 1.0
Rest 0.4
Pos. 27
Tempo 116
Pitch G3
Vel. 72
Dur. 0.4
Bar
Pos. 0
Tempo 116
Pitch A3
Vel. 80
Dur. 4.0

Tokenizing Polyphony

Hierarchical - Program token before each note

- Longer token sequence
- Notes encoded sporadically
- Simultaneous notes are close together
- Interrupts melodic flow

Flat - Track Separators every 1-2 bars

- Shorter token Sequence
- Organized notes
- Simultaneous notes are far apart
- Interrupts harmonic flow

Revamped MIDI Plus

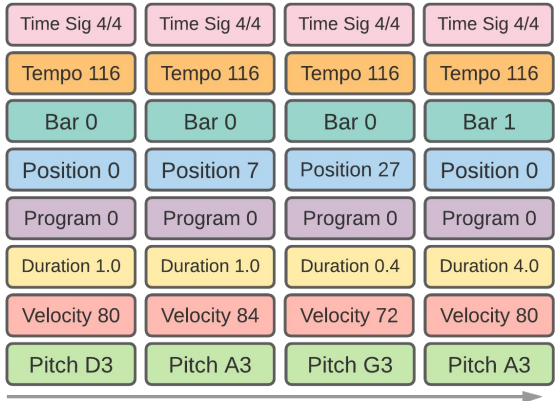
To Pool or Not To Pool?

Pooling = Process of merging tokens into new tokens

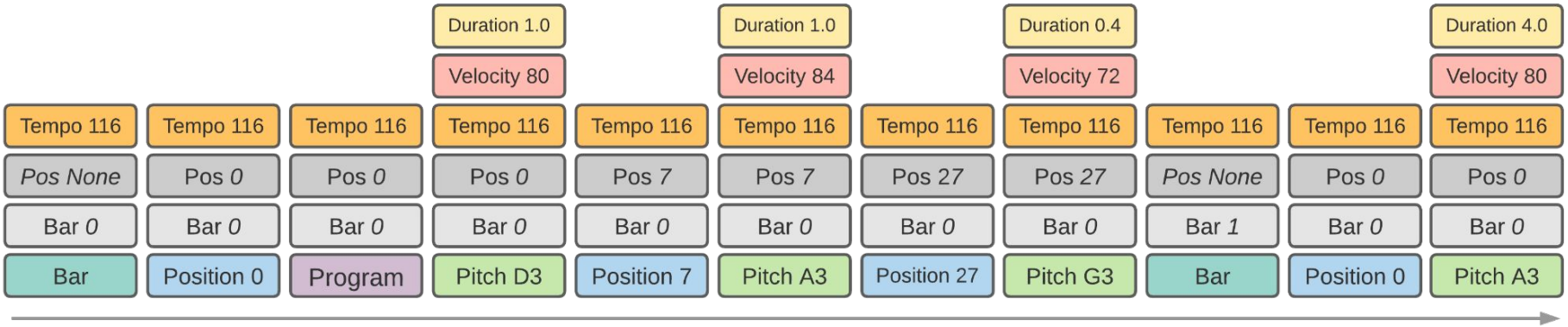
- Can be done statistically: BPE, WordPiece…
  - WordPiece equivalent for music?
- Can be done algorithmically:
  - Can group all note-related tokens in a pool

What goes in a pool? Is it necessary? Can it hurt?

Octuple



MuMIDI



Special Tokens

Special Tokens = Indirect information, don't have a concrete symbol.

- You know some already: [BOS, EOS, PAD, MASK]
- Steer the direction of the generations
- When generated, we ignore them
- Any ideas?

Adjusting for misplaced generated tokens

- Transformers are unpredictable
- Need to have predefined rules for failure cases
- Some failures are worse than others:
  - Missing Velocity token?
  - Missing Pitch token?
  - Early / Late / Missing Bar token?
- If you are getting too many, your token resolution may not be adequate to your data

Activity: Explore phrase-based pooling

If Notes are to music what letters are to speech, it would stand to reason that musical phrase-based pooling could improve our tokenizers, as seen in NLP.

- What's a phrase?
- Can we tokenize all phrases?
- Notes are already quite complex; what info do we want in the pool, and what not?
- Rule-based pooling or statistics-based?
- Pick a fragment of sheet music and tokenize it by hand

In clusters of 4 - 6 people

MIDITok

- Python Library - Compilation of tokenizer implementations
- Built on top of SyMusic for Score-based representations of MIDI
- HuggingFace hub integration
- Makes it easy to write your own tokenizer within the framework

MIDITok Implemented Tokenizers

- link

Why implement your tokenizer in MIDITok

- Handles Saving - Loading of your configs
- Provides helpful methods to use
- Easy training for BPE
- Clean, with pre-built pipelines for testing
- Built-in token validation
- Can act as a robust boilerplate