ML for Ab Initio Data:

A Tour of Knots and Natural Language

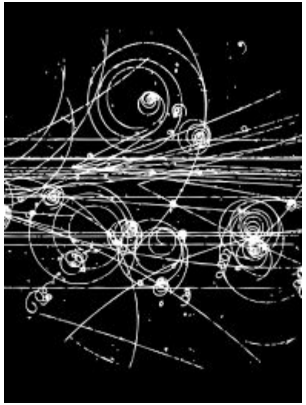
Jim Halverson

based on: 2010.16263 and work in progress
with Gukov, Ruehle, and Sulkowski

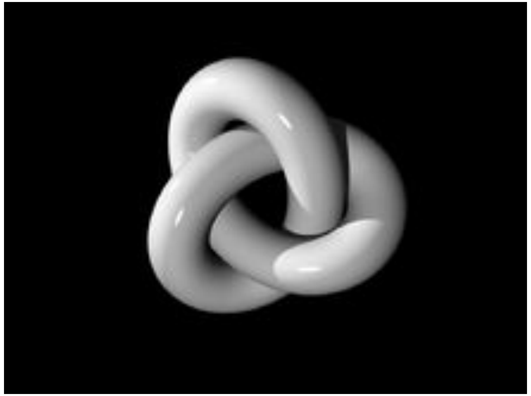
Northeastern University

What is data?

Do we create it?



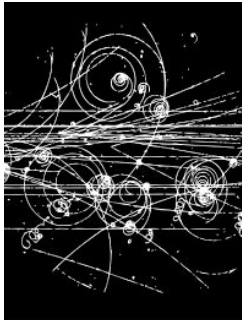
Do we collect it?

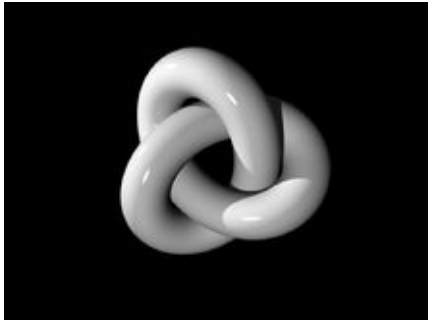


Do we define it?

How to differentiate?

Q: would ET have this data?





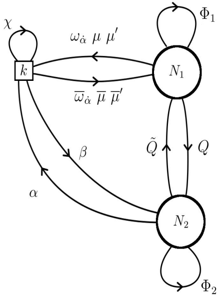
ET could have collided protons,
determined physics invariant under spacetime translations.

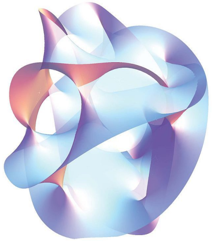
ET could have come up with a definition of a
mathematical knot.

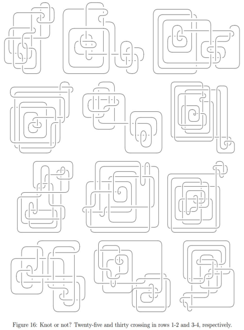
ET probably does not have these
handwritten digits.

Ab Initio Data and Ab Initio Everything

Ab Initio Data:







Data with definitions!

Not just “ML for Math,”
but ML for subjects with
rich, mine-able theoretical data.

Large, complex, structured datasets.

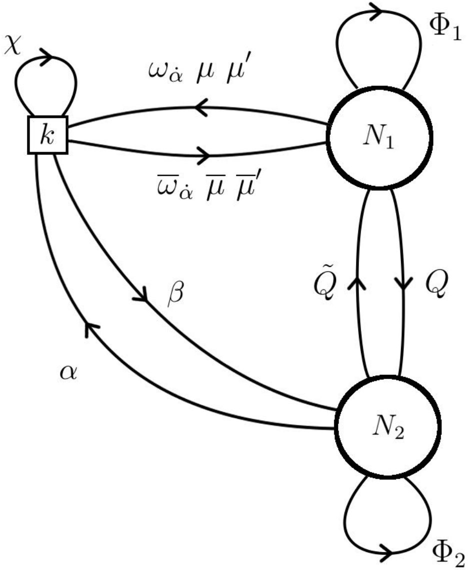
Ab Initio AI:

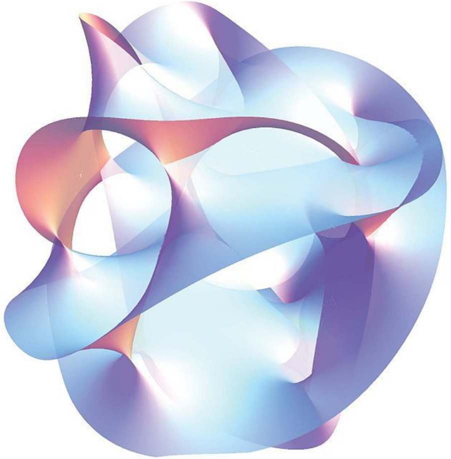
@IAIFI, we want ab initio everything!

Theme:
advance knowledge of fundamental interactions using innovative methods in AI built upon *ab initio* physics principles, while also advancing foundations of AI.

Today: Ab Initio Data

Ab Initio Data: Examples





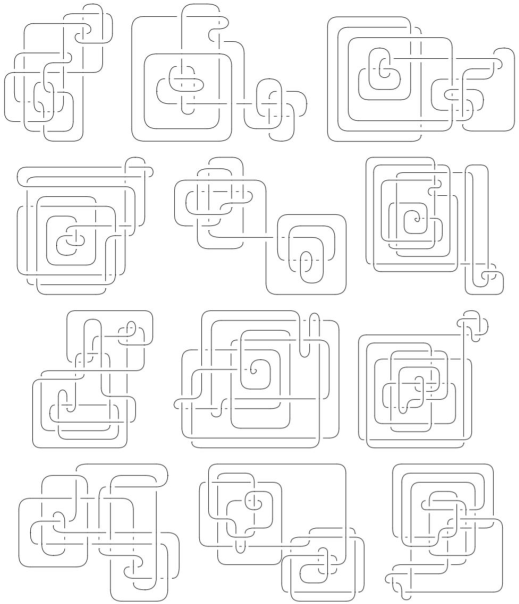


Figure 16: Knot or not? Twenty-five and thirty crossing in rows 1-2 and 3-4, respectively.

Chiral Gauge Theories:
Decorated graphs representing constrained theories of particle physics.

Calabi-Yau Manifolds:
Enormous network of manifolds related by topological transitions, of interest for string theory and algebraic geometry.

Mathematical Knots:
The subject of this talk!

Outline: An Example of ML for Ab Initio Data

- Knots and Natural Language

- Learning to Unknot

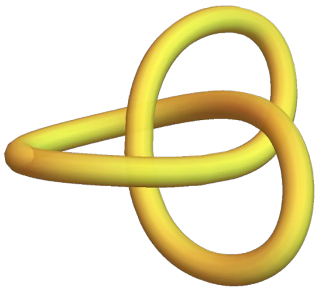
- The General Topological Problem

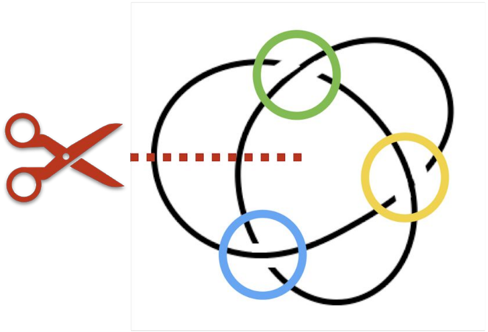
Unifying Q: Can machine learning automate topology?

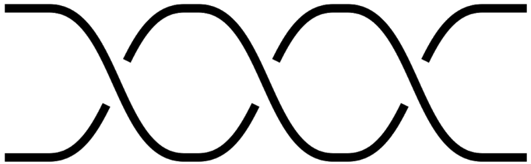
Knots and Natural Language

Topological Objects as Words

Knots and Braids



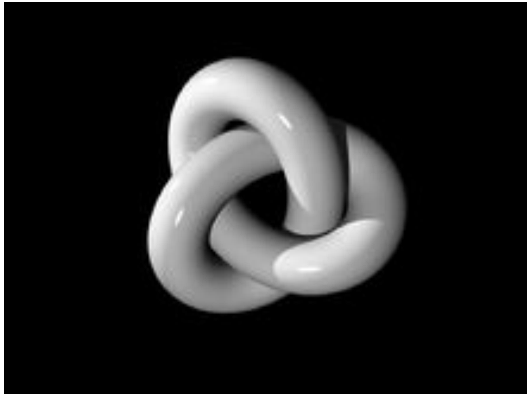




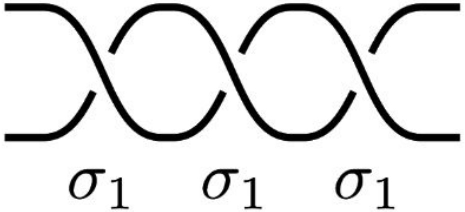
$\sigma_1 \quad \sigma_1 \quad \sigma_1$

$$\sigma_1 \sigma_1 \sigma_1$$

Representing the Trefoil Knot



Trefoil Knot:
Our old friend from
the first slide.

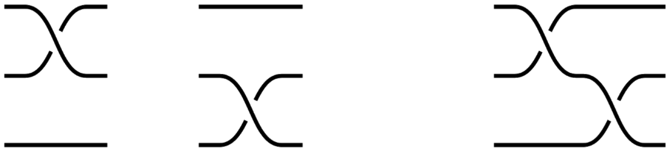


1 1 1

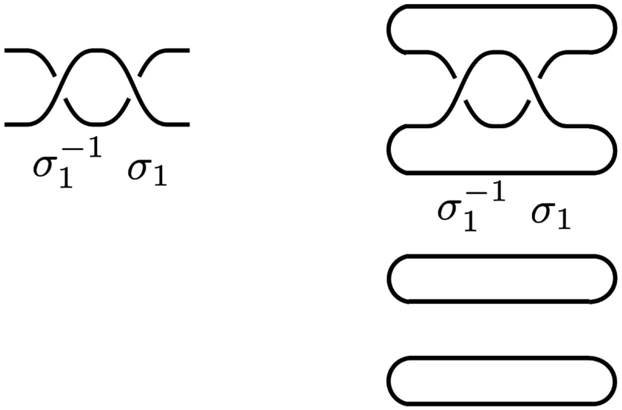
Braids: They Form a Group!

Identity:

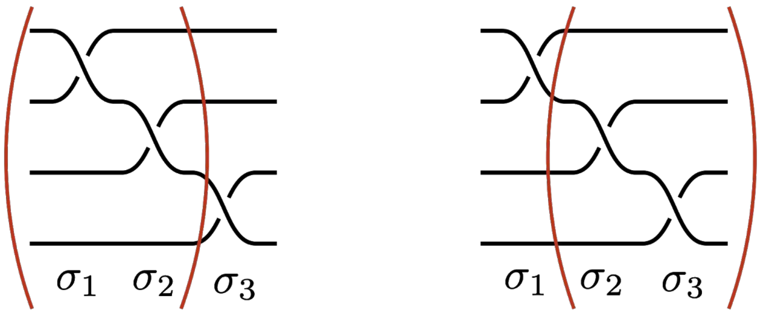
Composition:



Inverse:

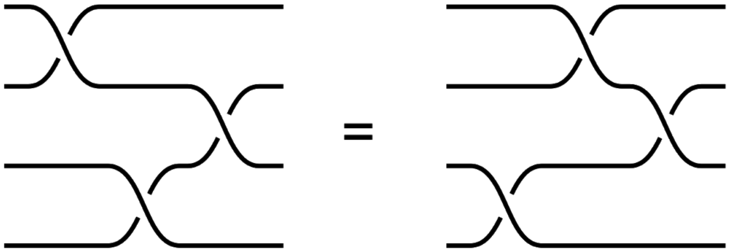


Associativity:



Braid Equivalence

Braid Relation 1:

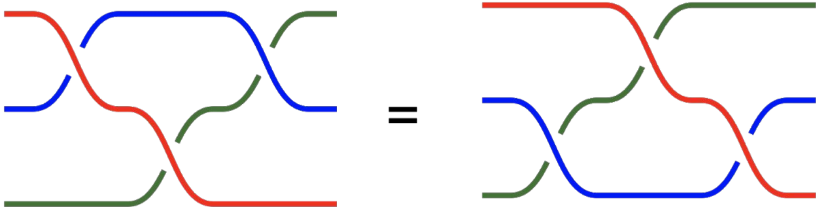


$\sigma_1 \; \sigma_3 \; \sigma_2$ = $\sigma_3 \; \sigma_1 \; \sigma_2$

(i.e. some generators commute)

$$\sigma_i \sigma_j = \sigma_j \sigma_i \quad \text{if} \quad |i - j| > 1$$

Braid Relation 2:

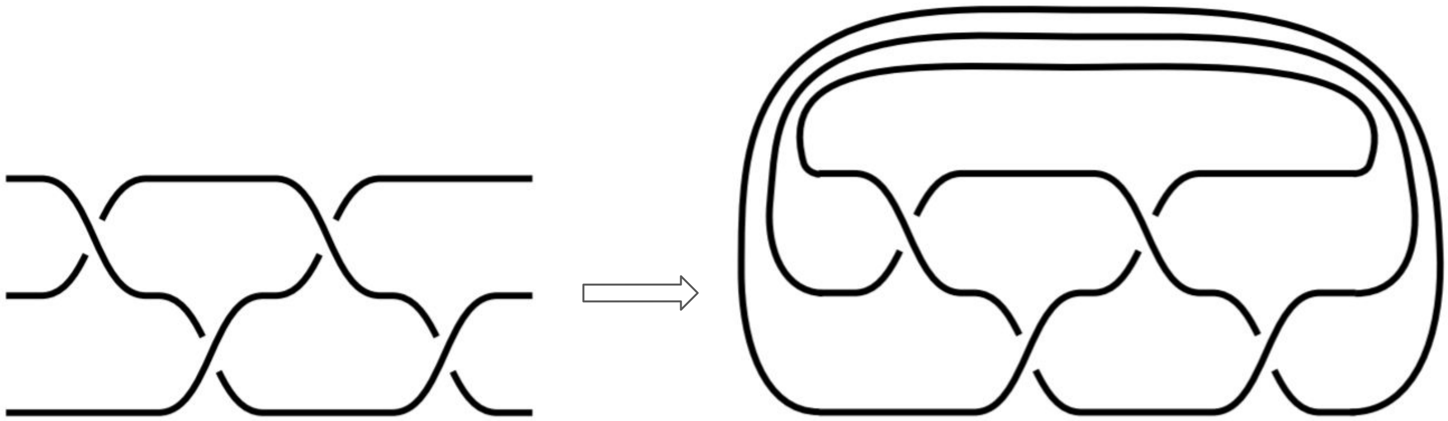


$\sigma_1 \; \sigma_2 \; \sigma_1$ = $\sigma_2 \; \sigma_1 \; \sigma_2$

(i.e. can rearrange braid)

$$\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1}$$

Knots from Braids via Closure

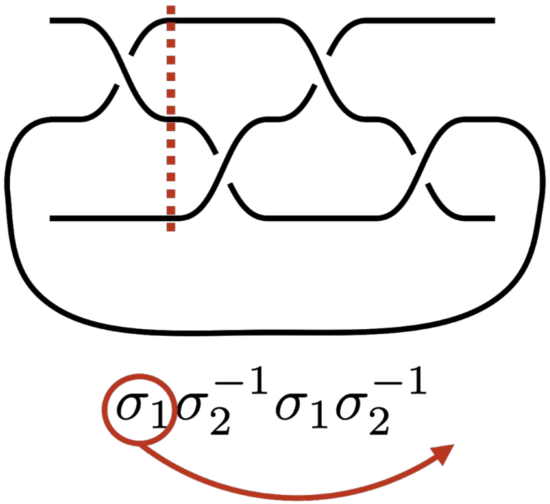


Can turn braids back into knots!

The same knot can be represented as the closure of many different braids.

Knot Equivalence of Braid Closures

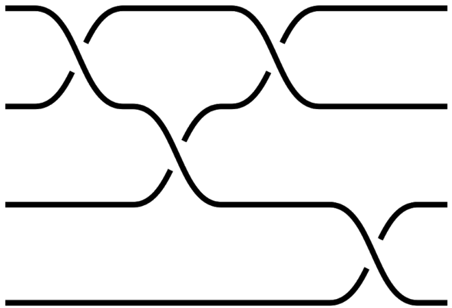
Markov Move 1: Conjugation



$\sigma_1 \sigma_2^{-1} \sigma_1 \sigma_2^{-1}$

$$w \to \sigma_i^{\pm 1}\, w\, \sigma_i^{\mp 1}$$

Markov Move 2: Stabilization



$$w \to w\, \sigma_{N+1}^{\pm 1}$$

The Upshot

Thought of as braids, knots are represented by **words**.

Knot equivalence becomes equivalences between different words.

How do we determine when two words carry the same meaning?

These questions are the domain of ***natural language processing.***

Natural Language Processing

Learning Semantics:

E embeds words into vector space.
e.g. E(king) - E(man) + E(woman) = E(queen)

Generative language models:

bold text: human-written prompt.
rest: generated by GPT-3 (OpenAI).

Below is a screenplay for a film-noir hard boiled detective story by Raymond Chandler about the boy wizard Harry Potter:

“Harry Potter”
By Raymond Chandler

Harry Potter, Private Eye.
Scene: A small dingy office, early morning, furniture of the Salvation Army store variety. Sordid atmosphere. Harry Potter, in ratty tweed suit, unpressed shirt, and unshined shoes, sits behind the desk looking haggard, rumpled, and embittered. In one corner of the office a torn shade rattles at the wind gusts outside. Harry has one long index finger on the butt of a .38 and he broods over a photograph. The photograph shows a gorgeous blonde wearing an evening gown. She is holding a cocktail in one hand and looking pensively at a cigarette in the other. A tear adorns her cheek. Harry looks at her for a long moment, sighs, then puts his gun down and stands. He straightens his clothes, scratches his head, then with more resolution turns around, picks up his pistol, and leaves the office. He passes through the front room and looks out the dirty window. An old woman is washing the sidewalk outside with a mop. A young man in a double breasted gray suit is leaning against the building. Harry sighs again and goes out the door. He walks up to the young man and without ceremony punches him in the jaw.

NLP in Our Context

Learn commutativity and non-commutativity:

He's sometimes right = Sometimes he's right $\sigma_1\sigma_3 = \sigma_3\sigma_1$

The chicken eats the scientist ≠ The chicken eats the scientist $\sigma_1\sigma_2 \neq \sigma_2\sigma_1$

Learn equivalences:

The scientists read the paper = The paper was read by the scientists

$$w = w\sigma_{N+1}$$

$$w = \sigma_i^{-1} w \sigma_i$$

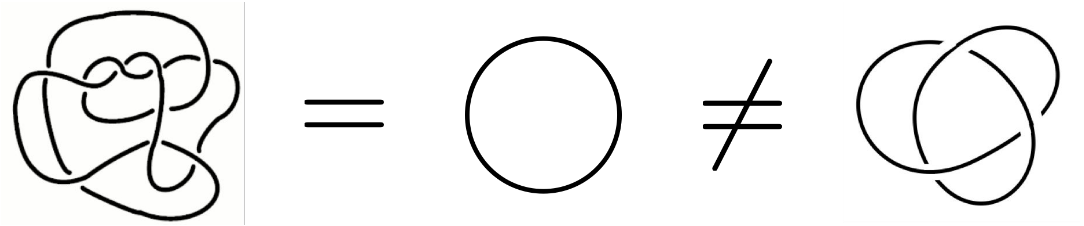
Learning to Unknot

Supervised Learning with Transformers
Emergent Notions of Hardness

Reinforcement Learning for Explicit Unknotting

The Unknot Problem

Q: is a given knot K the unknot?



Knot invariants?

Alexander Polynomial: 1 for unknot, converse is not true, + fast.
Jones Polynomial: 1 for unknot, converse not known to be true but is true for up to 24 crossings, but slow #P-hard.
Khovanov Homology: detects the unknot, but slow (fast would contradict #P-hard Jones).

No known fast invariant that detects the unknot.

Knot or not? A game for children

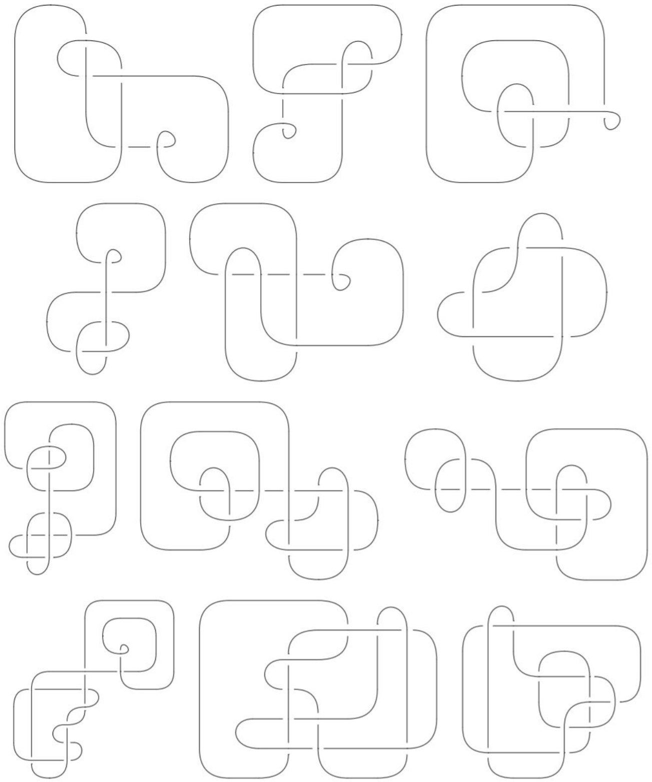


Figure 14: Knot or not? Five and ten crossing in rows 1-2 and 3-4, respectively.

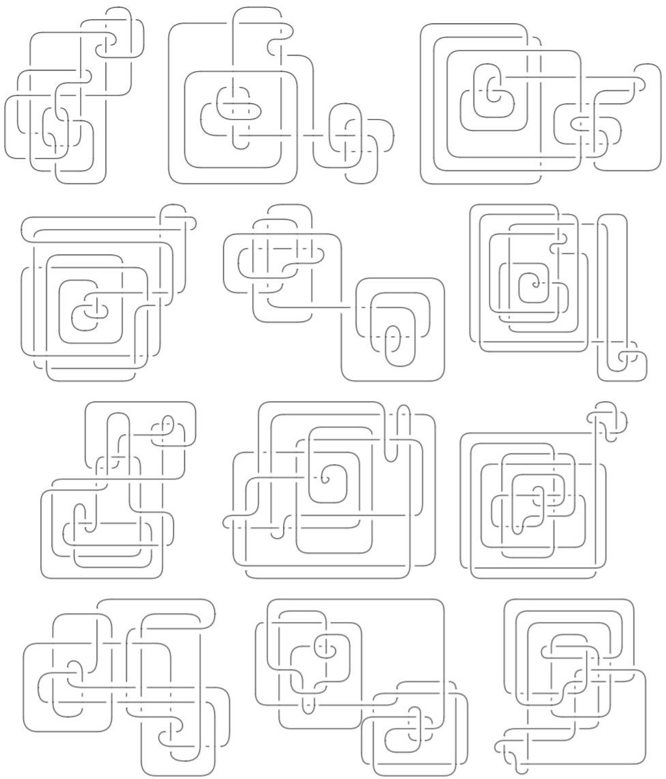


Figure 16: Knot or not? Twenty-five and thirty crossing in rows 1-2 and 3-4, respectively.

[12]Solutions are presented left-to-right, top-to-bottom, with K and U denoting non-trivial knots and unknots, respectively. Fig. 14: KUUKUKUUKUKK. Fig. 15 UKUKKUKKUKUU. Fig. 16 KUUKUKUUKKUK.

point: difficulty increases with crossings.

Generating Data: Priors for Random Knots and Unknots

Algorithm 5 RandomUnknot: generate random unknot representative.

Require: $n_{\text{letters}}, M \in \mathbb{Z}$.

```
Braid B ← empty braid word.
while |B| ≠ n_letters do
    if |B| > n_letters then
        B ← empty braid word.
    end if
    for k ∈ {1, ..., M} do
        B ← RandomMarkovMove(B).
        if |B| − 1 ≥ 0 then
            B ← BraidRelation2(B, start position ∼ U({1, ..., |B|})).
        end if
    end for
    B ← SmartCollapse(B).
end while
return B.
```

Algorithm 6 RandomKnot: generate random non-trivial knot representative.

Require: $n_{\text{letters}}, n_{\text{strands}}, M \in \mathbb{Z}$.

```
Braid B ← empty braid word [].
while |B| ≠ n_letters do
    if |B| > n_letters then
        B ← empty braid word.
    end if
    while |B| < n_letters do
        i ∼ U({0, 1}).
        j ∼ U({0, ..., n_strands − 1}).
        B ← B + [(−1)^i j]
    end while
    B ← Knotify(B)
    if B ≠ [] then                    ▷ Knotify sometimes yields an empty word.
        for k ∈ {1, ..., M} do
            B ← RandomMarkovMove(B).
            B ← BraidRelation2(B, start position ∼ U({1, ..., |B|})).
        end for
        B ← SmartCollapse(B).
    end if
end while
return B.
```

Q: how do we generate examples?

Attention, Attention! Meet Reformers.

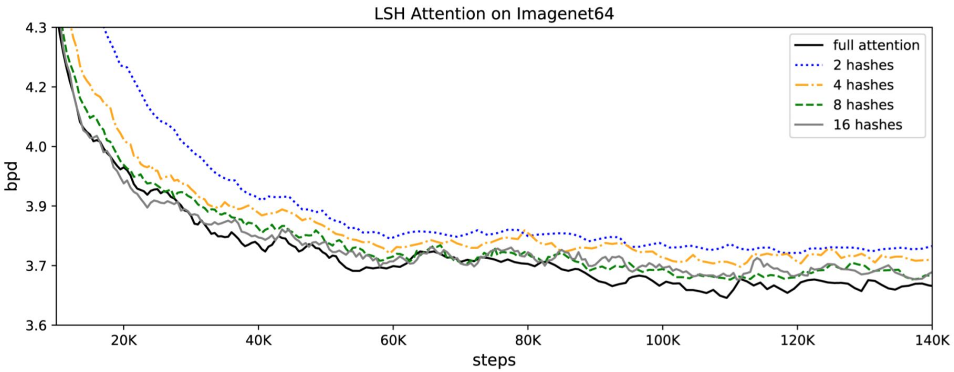
Attention Mechanism: learn what in the sequence carries the most meaning, i.e. pay attention to it.

"Attention is all you need" [1706.03762, Vaswani et al.]

Reformer: The Efficient Transformer

Upshot: efficiency gains due to replacing scaled dot-product attention with locality-sensitive hashing (LSH) attention. $O(L^2) \rightarrow O(L \log L)$

"Reformer: The Efficient Transformer" [2001.04451, Kitaev et al.]
Good library: reformer-pytorch



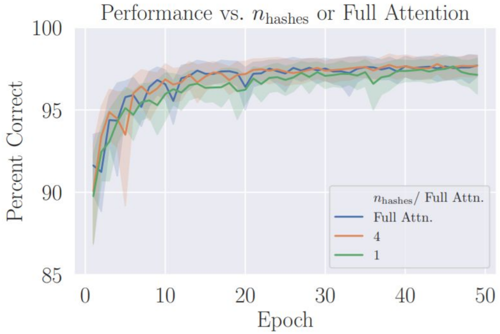
Decisions, Decisions

binary class. on unknot decision.

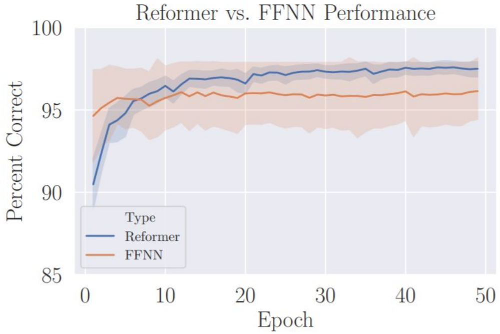
trained on thousands of knots and unknots with diff. #s of crossings.

Comments:

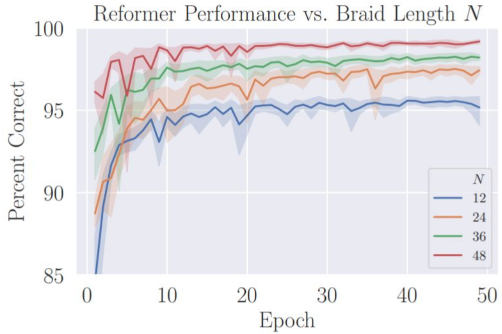
1) NLP wins, but barely. (b/c easy?)
2) Reformers ~ Transformers
3) performance up with N, a lot of fixed # words, less so for fixed # letters.



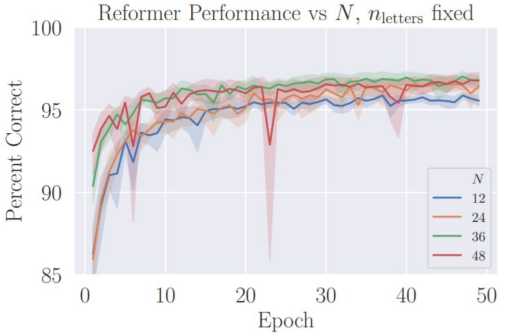
(a) Performance dependence on the number of locality sensitive hashes.



(b) Performance comparison between reformer and feedforward network.



(c) Performance dependence on the braid length. Performance increases with N.



(d) Performance when number of braid letters, rather than number of braid words, is fixed.

Hardness and Jones Correlations

Hardness:

@ right, note some small peaks in wrong spot, networks quite sure of their wrong predictions!

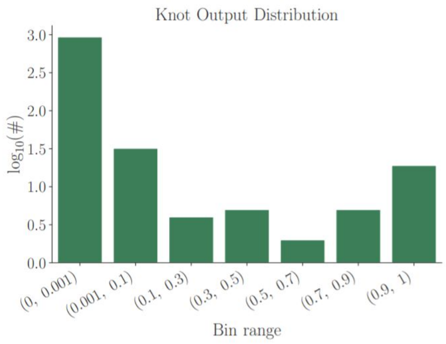
hardness of knot ***persists*** across diff. inits.

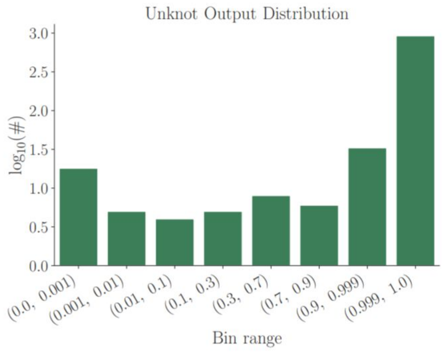
e.g. 1000 N<=9 test braids have 30 hard instances, 19 of which are trefoils, despite ~ ¼ knots being trefoils. **Knots with fewer crossings harder!?**

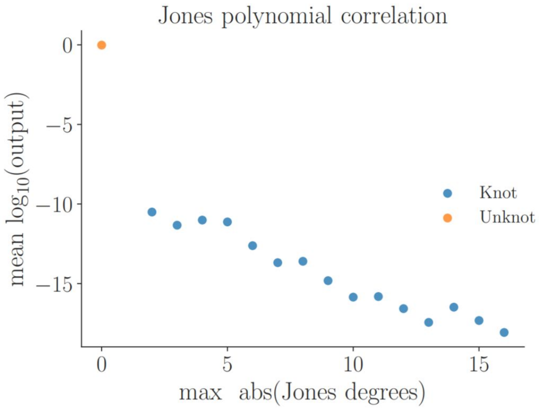
Jones Polynomial Correlations:

@ right, network confidence on correctly labelling knots correlated with Jones degree.

Jones not used in training at all! Learned feature.







Can we explicitly unknot?

Find a sequence of moves (knot equivalences) that reduce the braid word to nothing.

Reinforcement Learning

- an *agent* interacts in an *environment*.
- it perceives a *state* from state space.
- its *policy* picks an action, given state.

$$\pi : \mathcal{S} \to \mathcal{A}$$

- arrives in new state, receives *reward*.
- successive rewards accum. to *return*.
  future rewards penalized by *discount*.

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

- *state-* and *action-value* functions:

$$v(s) = \mathbb{E}[G_t | S_t = s]$$

$$q(s, a) = \mathbb{E}[G_t | S_t = s, A_t = a]$$

Famous Example: AlphaZero

“Mastering the game of Go without Human Knowledge”
- Silver et al, Nature 2017

RL with ***no human data.***

“A general reinforcement learning algorithms that masters chess, shogi, and Go through self-play.”
- Silver at al, Science, 2018

Unknotting with RL

State Space:
zero-padded braids of length 2N.

Action Space: dim = N+5

1) shift left
2) shift right
3) BR1 and shift right
4) BR2 and shift right
5) Markov 1, conjugate by arb. gen.
6) SmartCollapse: destabilize and remove inverses until unchanged

Reward: negative braid length

End of game: empty braid or 300 moves.

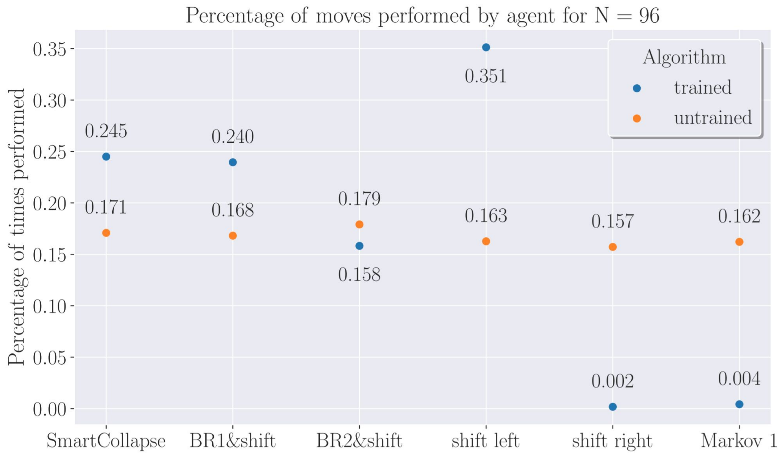
RL Algorithms:
A3C: asynch. advantage actor-critic, worker bees report back to 2 A/C nets.

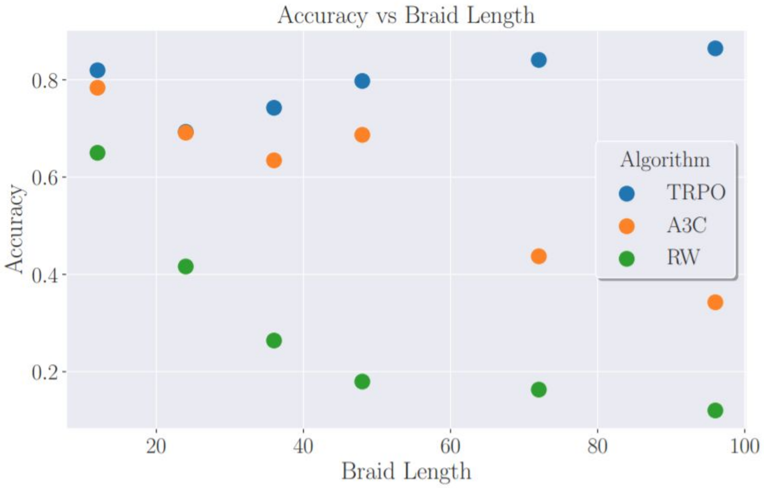
TRPO: trust region policy optimization. Policy updates and steps depend on loss curvature.

Results + Interpretability

RL wins.
RW decreases rapidly.
TRPO crushes: flat in N.

Wins not just in % solved in <= 300 moves (below), but also in number of moves performed.





Interpretability via rollout.
i.e. learning is a flow in the state-dependent distribution on action space. **how does it change?**

SmartCollapse: only action that reduces N.

Shift left / right asymmetry b/c many shift rights come for free. **See paper for more interpretation!**

The General Topological Problem

Automated Learning of Topological Equivalence Classes

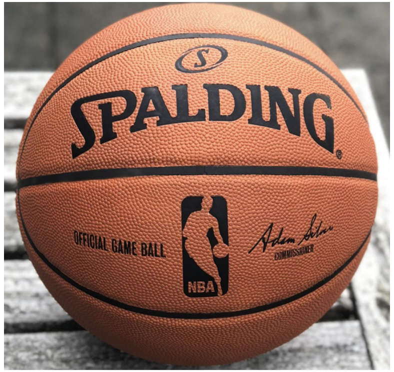
Actually doing it for knots

(Preliminary results)

Can machine learning automate topology?

=

≠



≠

Topology is about equivalence classes of objects,
e.g. under deformation

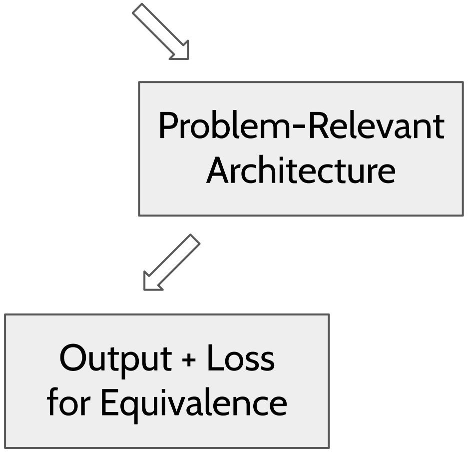
Goal: learn latent space embedding that encodes equivalence.

Ideally tight, well-separated clusters.
Complete topological invariant?

Example: genus of Riemann surface

Learning Topological Equivalence: General Schematic

$\{e_{k1}, \ldots, e_{kd}\} \in [e_k]$
$k = 1, \ldots, N$
d reps of N eq. classes



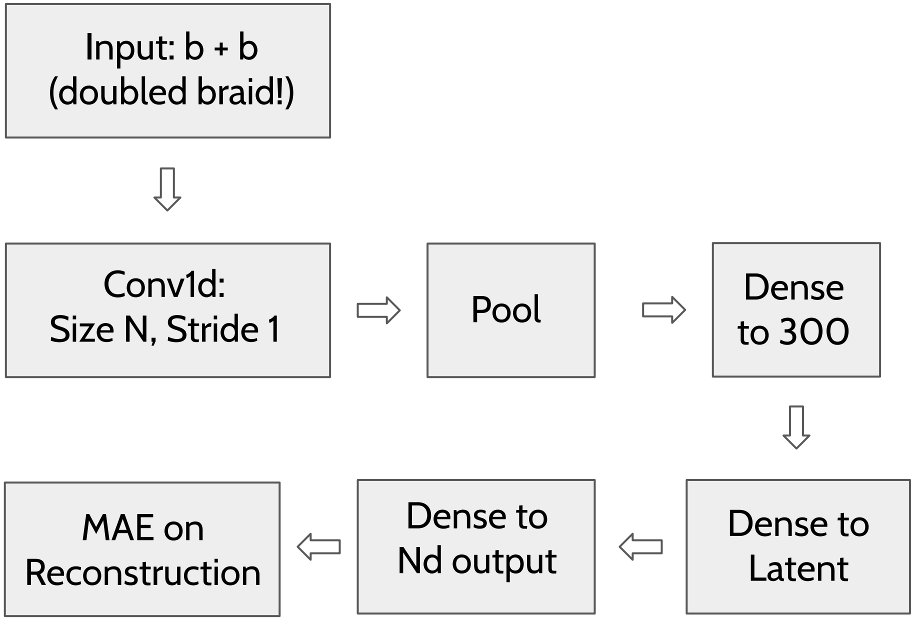
Lots of possible ways to do this, similarity learning.

e.g., Triplet Loss:
Input is (A,P,N), A the anchor, P a positive with [P] = [A], and N a negative with [N] != [A].

Minimize distance of P to A, maximize distance of N to A

e.g., Simple Reconstruction:
Pick distinguished member of class, require all reps reconstruct that member at output, look for clustering in latent layer before output.

Knot Application: Learning Topological Equivalence

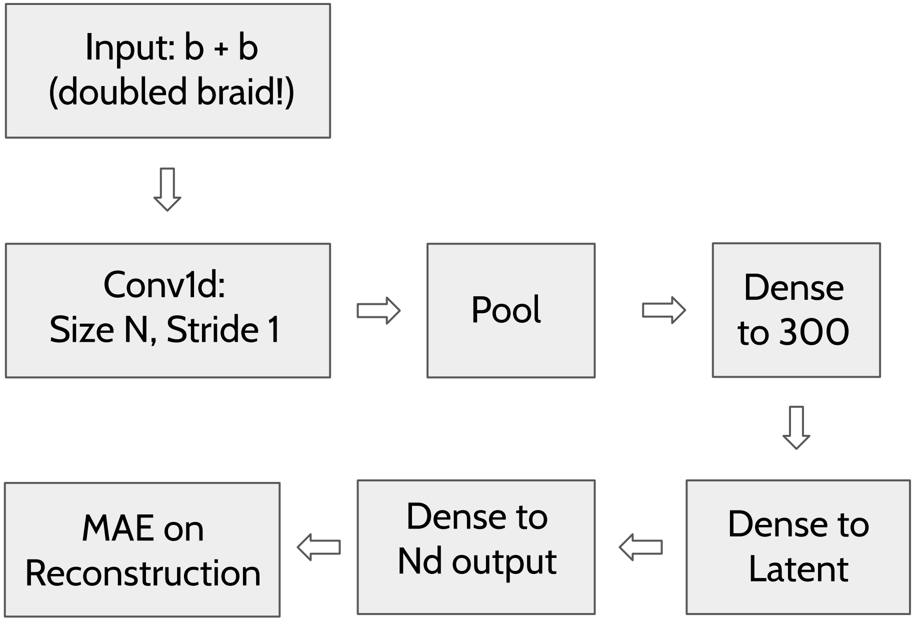


Input: instead of trefoil as [1 2 1 2]
double it due to periodic b.c.
[1 2 1 2 1 2 1 2]

conv filter is braid length, stride 1,
sees all translates of braid!

Reconstruction: network tries to
reconstruction distinguished rep of
class, using MAE loss

Knot Application: Learning Topological Equivalence

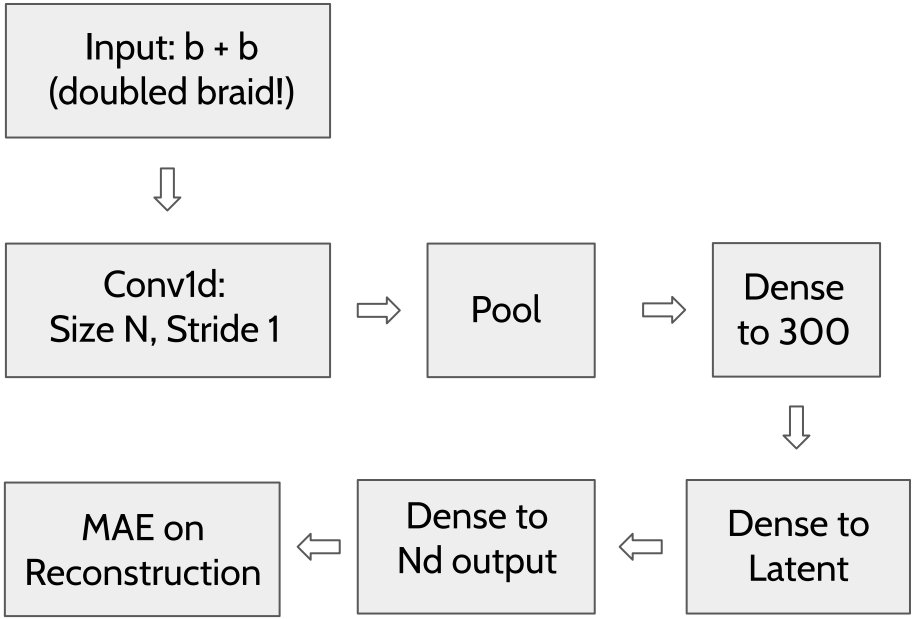


Input: instead of trefoil as [1 2 1 2]
double it due to periodic b.c.
[1 2 1 2 1 2 1 2]

conv filter is braid length, stride 1,
sees all translates of braid!

Reconstruction: network tries to
reconstruction distinguished rep of
class, using MAE loss

Knot Application: Learning Topological Equivalence

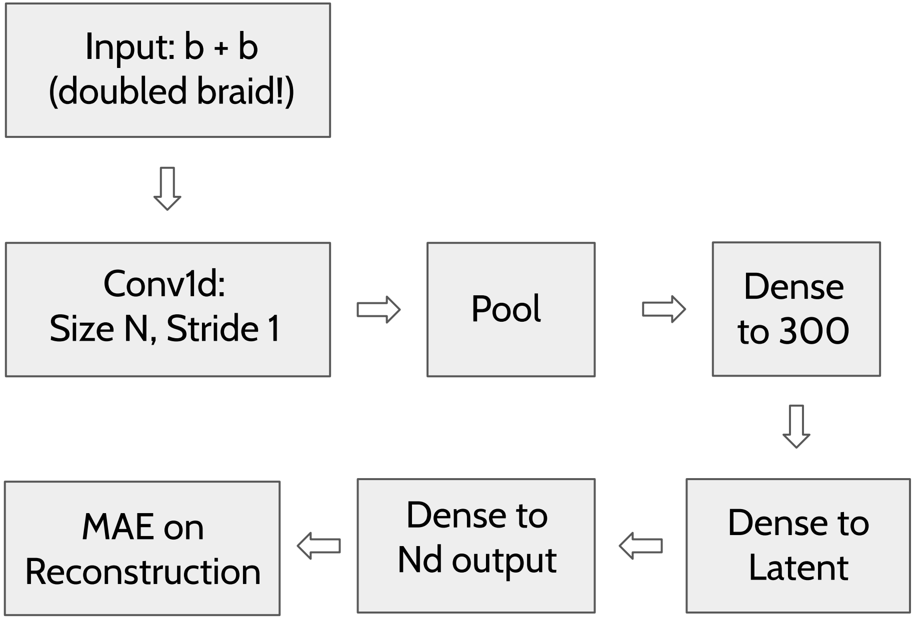


Input: instead of trefoil as [1 2 1 2]
double it due to periodic b.c.
[1 2 1 2 1 2 1 2]

conv filter is braid length, stride 1,
sees all translates of braid!

Reconstruction: network tries to
reconstruction distinguished rep of
class, using MAE loss

Knot Application: Learning Topological Equivalence

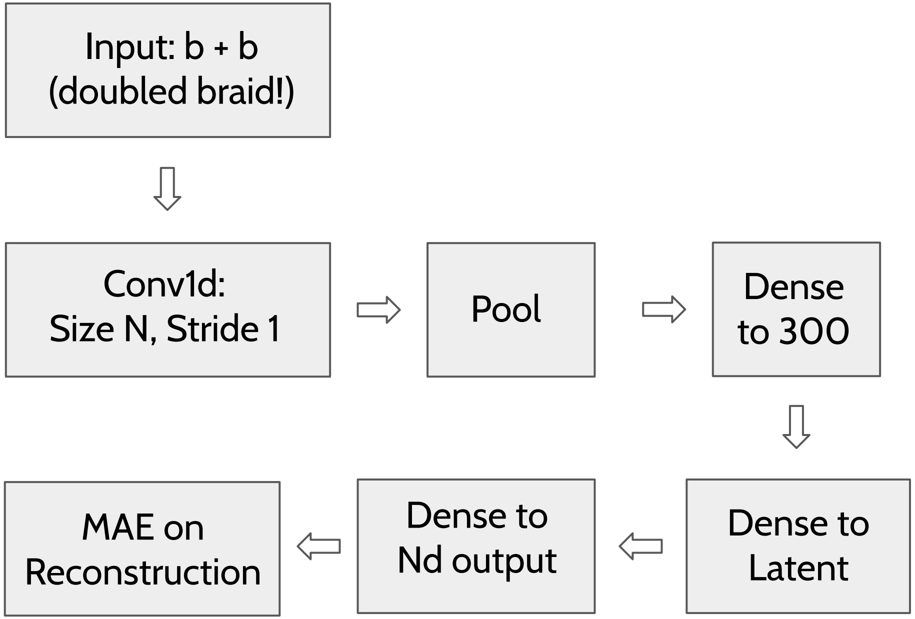


Input: instead of trefoil as [1 2 1 2]
double it due to periodic b.c.
[1 2 1 2 1 2 1 2]

conv filter is braid length, stride 1,
sees all translates of braid!

Reconstruction: network tries to
reconstruction distinguished rep of
class, using MAE loss

Knot Application: Learning Topological Equivalence



Input: instead of trefoil as [1 2 1 2] double it due to periodic b.c.
[1 2 1 2 1 2 1 2]

conv filter is braid length, stride 1, sees all translates of braid!

Reconstruction: network tries to reconstruction distinguished rep of class, using MAE loss

Knot Application: Learning Topological Equivalence

Training details:

200 equivalences classes of knots,
realized as 5-strand length 12 braids.
12 reps each class, 5 moves each from initial knot.

Embedding dims: 6, 4, 2

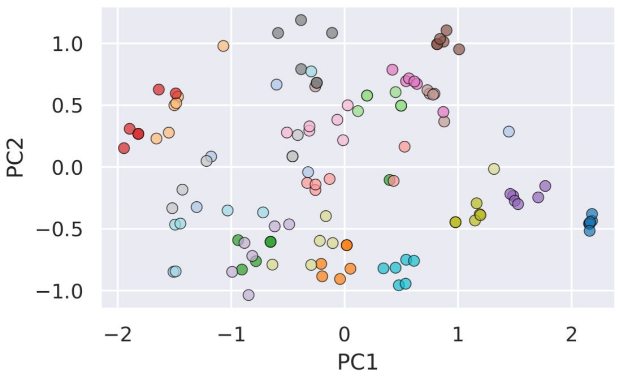
(90,10) train-test split.
400 epochs.

Trains in a minute or two.

Clustering details:

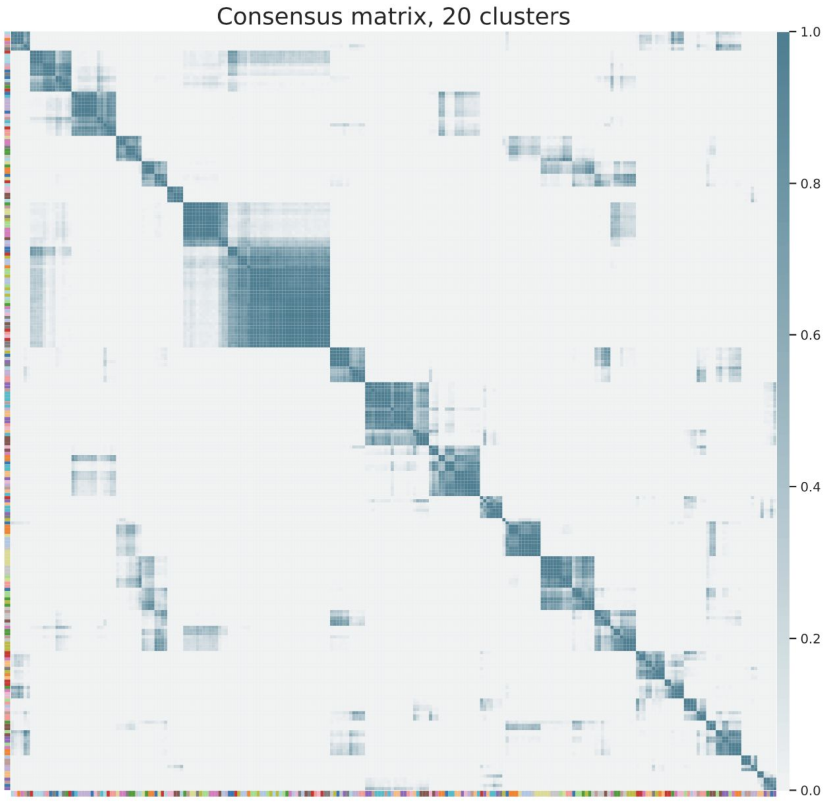
Want to cluster trained latent reps of braids.
Agglomerative clustering and dendrograms.

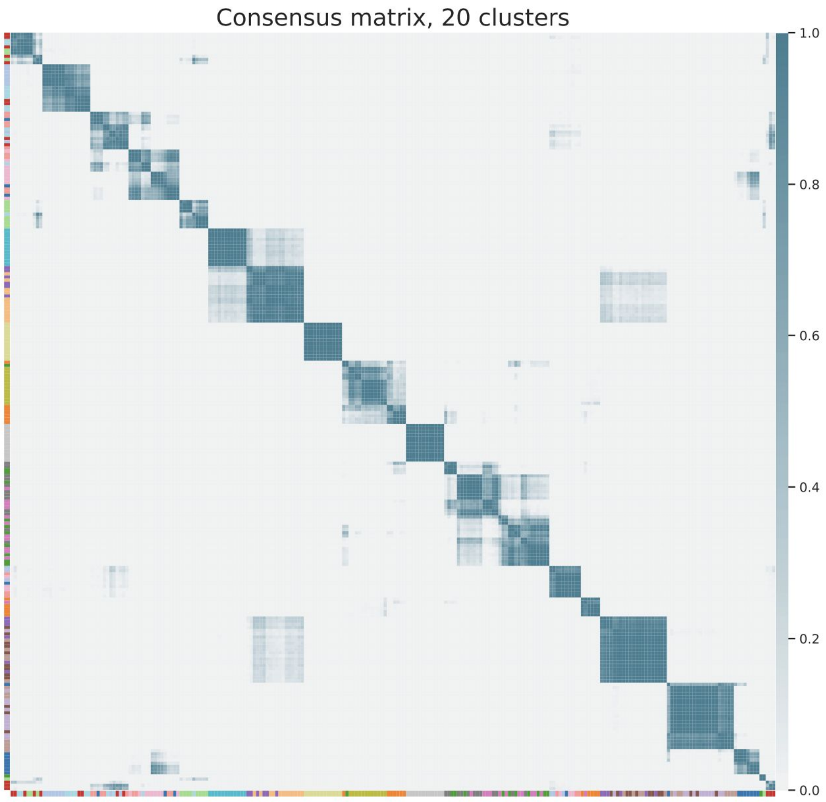
Consensus clustering with Kmeans,
600 resamples with resample proportion 0.9.



PCA for
Embedding Dim 4,
Explained Variance 80%

Consensus + Hierarchical Clustering for Knots





Untrained Latent Space

Consensus clustering on latent space, Hierarchical on Consensus
Ground truth topological labels on axes

Trained Latent Space

Conclusions for Knots

Knots as Ab Initio Data:

Data from definitions,
prior for sampling
knots and unknots.

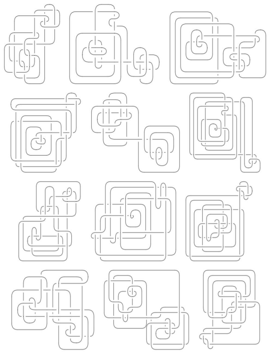
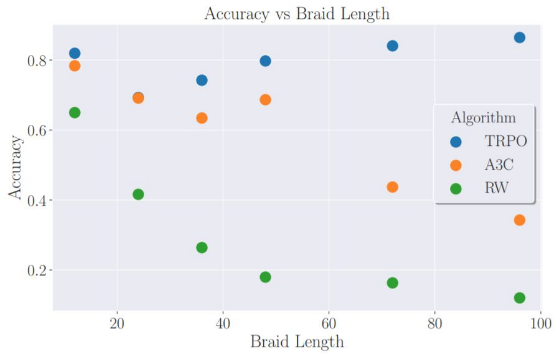
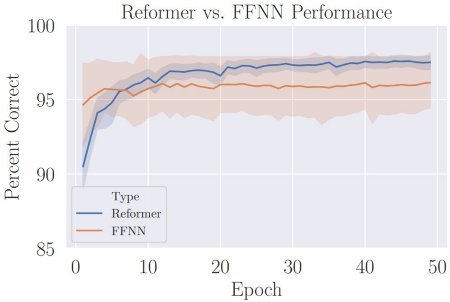


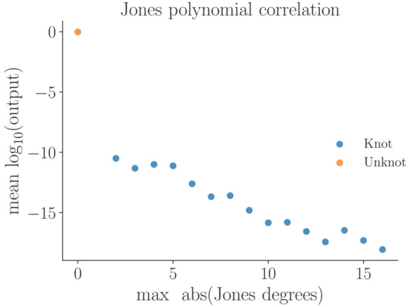
Figure 16: Knot or not? Twenty-five and thirty crossing in rows 1-2 and 3-4, respectively.

Explicit Unknotting with Reinforcement Learning:



Unknotting Hardness and Supervised Learning:



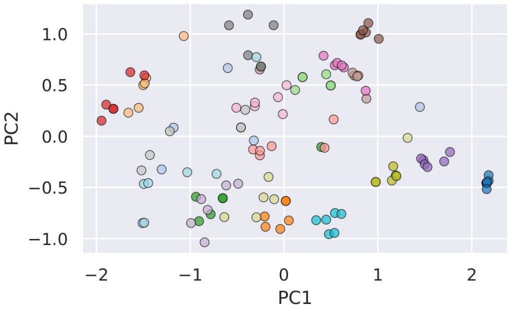


Binary classification for Unknot succeeds,
Learned correlation with Jones polynomial

General Topological Equivalence:

Latent space clustering
of topologically
equivalent objects?

Here: knots.



General Conclusion:

Data is more than what we create or collect.

It can stem directly from definitions, *ab initio data.*
It can be enormous and structured.

ML can help us understand it better.

Thanks!

Questions?

Or get in touch after:

e-mail: jhh@neu.edu
Twitter: @jhhalverson
web: www.jhhalverson.com