Hierarchical Token Merging (HTM)

A New Architecture to Learn Semantically Meaningful Representations

Alec Zorin Durgheu - pwtn63

Domain

Representation Learning:

$$\mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)$$

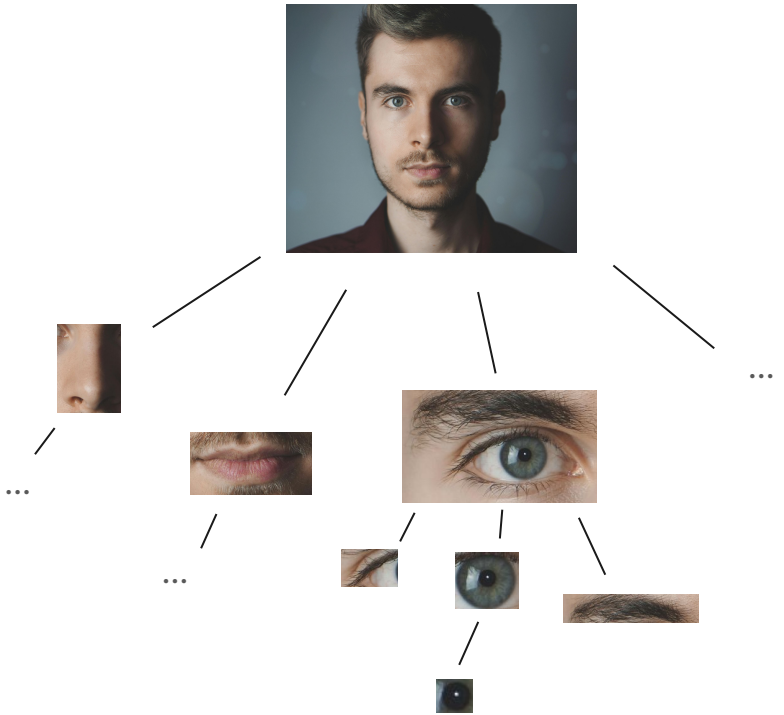
- ...where $\mathbf{x}_i$ is the input instance, $\mathbf{y}_i$ is the representation, and $\mathbf{f}$ is a learned function approximator (e.g., a neural network)
- Implemented iteratively in DL settings for feature extraction
    - Facilitates downstream tasks, such as image reconstruction and generation (as explored by HTM)

Problems

- Fails to fully capture data semantics
    - Frequently outputs hallucinations
- Black box problem
    - Difficult to understand how/why the model got to the output
- Other challenges:
    - Slow training/inference speed
    - Memory intensive
    - ...

Aims

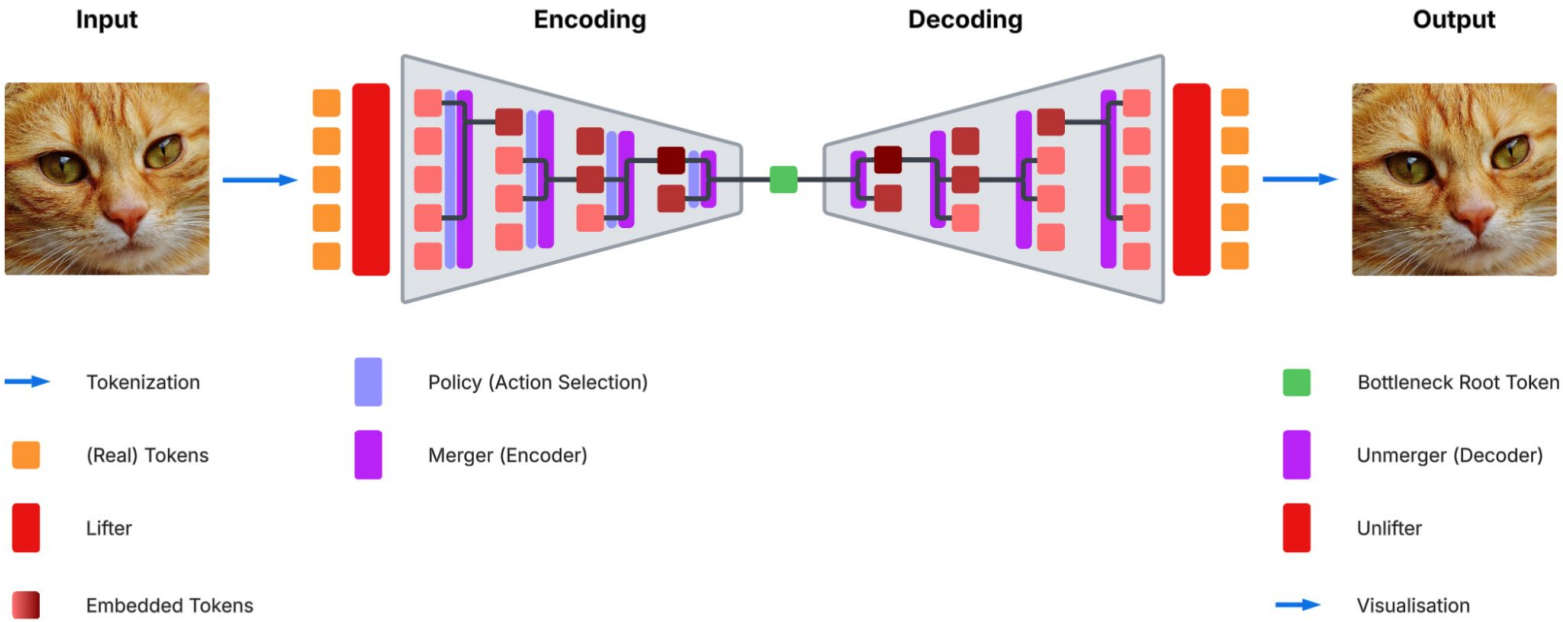
- Architectural alignment to the unique hierarchical data structure of each input
  - ⇒ recursive autoencoder design
- Semantically consistent organisation & clustering of the embedding space
  - ⇒ iterative merging of embedded tokens to the root-level bottleneck
- Interpretable forward pass & outputs
  - ⇒ learnable merging trajectories (actions) inline with an RL-inspired policy



Architecture

- Lifter
  - Maps real input tokens into leaf-level embedded tokens
- Unlifter
  - Maps leaf-level embedded tokens into real output tokens
- Merger (Encoder)
  - Merges two embedded tokens into one
  - Depth wise recurrence for abstraction of varying data granularity
- Unmerger (Decoder)
  - Similar to the merger, but in reverse
  - *Unmerges* one embedding into two
- Policy Network
  - Outputs logits from input pairs of embedded tokens
  - Logits reflect the quality of the merge between those tokens
- Classifier
  - Outputs two logits for each input embedded token
  - Logits correspond to class labels: the first for leaf-level tokens, the second for internal nodes

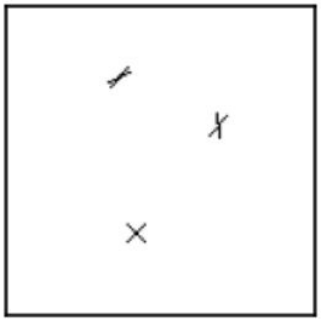
Forward Pass

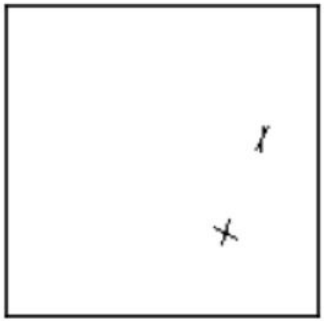


Datasets

Crosses:

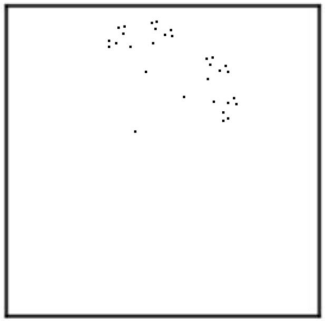
- Line segments: (x_1, y_1, x_2, y_2)
  - Two lines intersect at their midpoint to form crosses





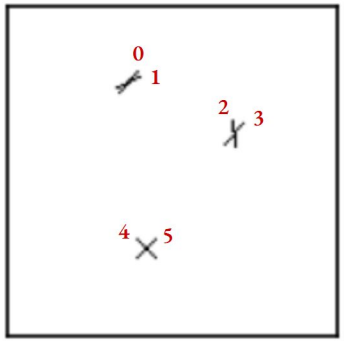
Trees:

- Nodes: (x, y)
  - Complete binary trees with shrinking distances

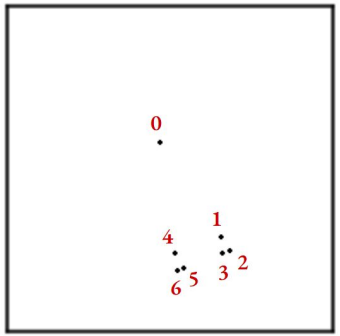


Policy Diagrams

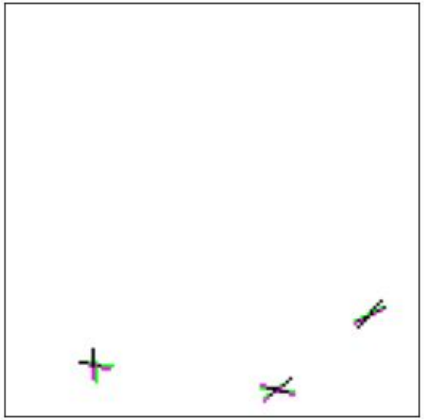
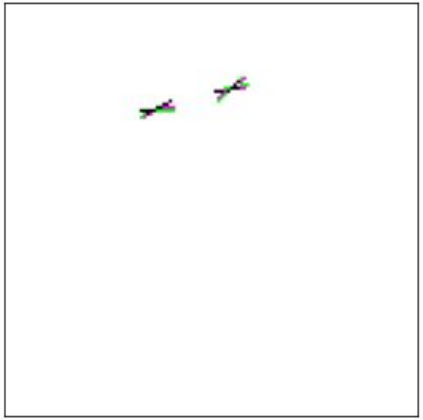
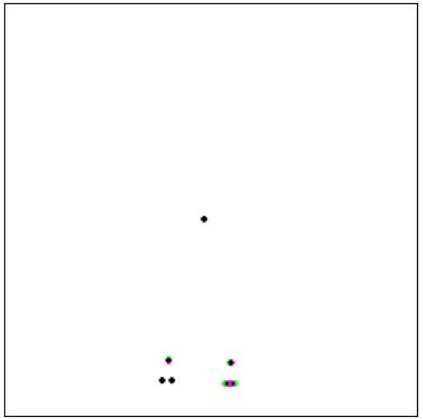
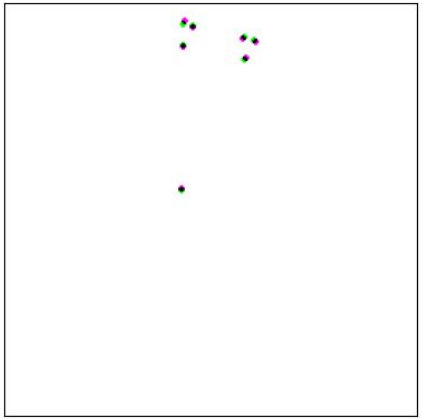
6	[[[1, 0],	[[[2, 3],	[[[2, 3],
7	[4, 5],	[4, 5],	[1, 0],
8	[7, 6],	[7, 6],	[6, 7],
9	[2, 3],	[0, 1],	[5, 4],
10	[8, 9]]]	[8, 9]]]	[8, 9]]]



7	[[[2, 3],	[[[5, 6],	[[[5, 2],
8	[5, 6],	[4, 7],	[4, 1],
9	[4, 8],	[3, 2],	[6, 3],
10	[1, 7],	[0, 8],	[0, 7],
11	[0, 10],	[1, 9],	[9, 8],
12	[9, 11]]]	[11, 10]]]	[10, 11]]]

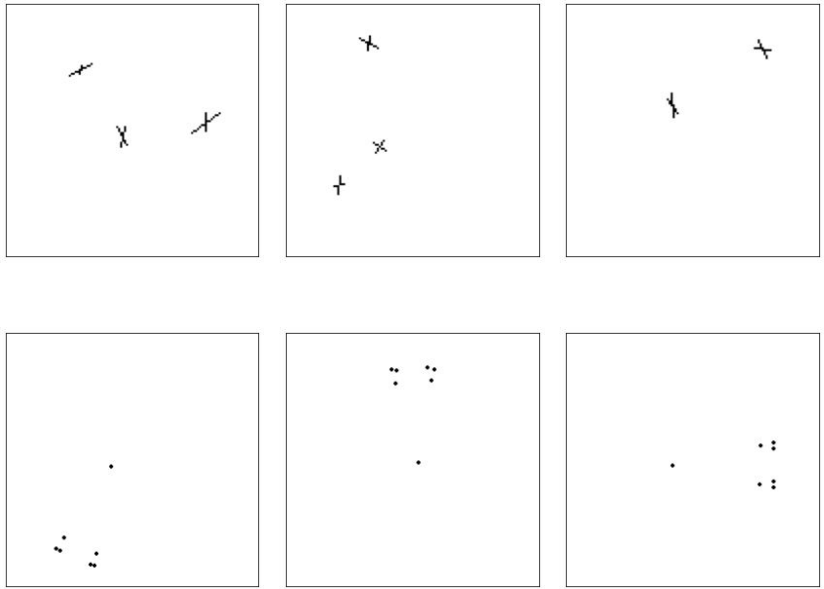


Reconstruction Examples

Generation

- Fit GMM to root node embeddings from trained HTM model
  - Samplable latent distribution
- Data generation procedure:
  - Sample root from trained GMM
  - Obtain classification result:
    - If internal node, unmerge and append outputs to generation tree
    - If leaf, extract it
    - Repeat until full traversal of generation tree
  - Return unlifted leaves from hierarchical decoding of sampled root



Conclusion

Successes:

- Promising results from simple proof-of-concept implementations
  - Great reconstruction fidelity
  - Interpretable policy *mostly* inline with expectations
  - Functional generative pipeline with reasonable results

Challenges & Some Future Directions:

- Slow training and inference speed
  - $\mathcal{O}(P^2)$ memory complexity from adjacency matrices (where P is the token population size)
    - ⇒ Development of efficient graphical data structures
- Noisy policy learning environment (despite curriculum learning and baselining)
  - On-policy and credit assignment problems from REINFORCE implementation
    - ⇒ Importance sampling from replay buffers
- Pairwise merging myopia
  - Actions are not contextualised within their active token population
    - ⇒ Self attention mechanism?
- …