Probing for Structural Representations in Language Models

Lecture for LING 3850 / 7850

Papers discussed:

- **Hewitt & Manning 2019**
- **Diego-Simón et al. 2024**
- **Orhan et al. 2026**

Zhenghao Herbert Zhou, Apr 6 & 8, 2026

Overarching Questions

Overarching Questions

- We know that **syntactic information is decodable** from language models' (LM) hidden representations (e.g., via probing classifiers);
  - Part-of-speech tags, morphological features (number, gender), etc.

Overarching Questions

- We know that **syntactic information is decodable** from language models' (LM) hidden representations (e.g., via probing classifiers);
  - Part-of-speech tags, morphological features (number, gender), etc.
- However, we also want to know:
  1. Whether richer representations beyond individual features, such as syntactic structures of entire sentences, are also encoded;
  2. If so, whether structural information is **geometrically organized** (in a human interpretable way) in the hidden states.

Overarching Questions

- We know that **syntactic information is decodable** from language models' (LM) hidden representations (e.g., via probing classifiers);
  - Part-of-speech tags, morphological features (number, gender), etc.
- However, we also want to know:
  1. Whether richer representations beyond individual features, such as syntactic structures of entire sentences, are also encoded;
  2. If so, whether structural information is **geometrically organized** (in a human interpretable way) in the hidden states.
- Do LMs have abstract, hierarchical representations of sentence structures? Answering this question helps us build intuitions about **information organization in language models' intermediate hidden states**.

Outline

Outline

- **Hewitt & Manning 2019 [structural probe]**:
  - *distance* information in syntactic trees could be recoverable from a linear subspace of contextual embeddings;

Outline

- **Hewitt & Manning 2019 [structural probe]**:
  - *distance* information in syntactic trees could be recoverable from a linear subspace of contextual embeddings;
- **Diego-Simón et al. 2024 [polar probe]**:
  - *direction/head* and *type* information of dependency parses could also be recovered geometrically;

Outline

- **Hewitt & Manning 2019 [structural probe]**:
  - *distance* information in syntactic trees could be recoverable from a linear subspace of contextual embeddings;
- **Diego-Simón et al. 2024 [polar probe]**:
  - *direction/head* and *type* information of dependency parses could also be recovered geometrically;
- **Orhan et al. 2026 [follow-up on structural probe]**:
  - generalizing this geometric-probe logic beyond syntax to *phonemic (🤫)* and *semantic* information, and showing that those properties emerge over training;

Hewitt & Manning 2019

Structural Probe

One way to represent syntax: Dependency Parsing

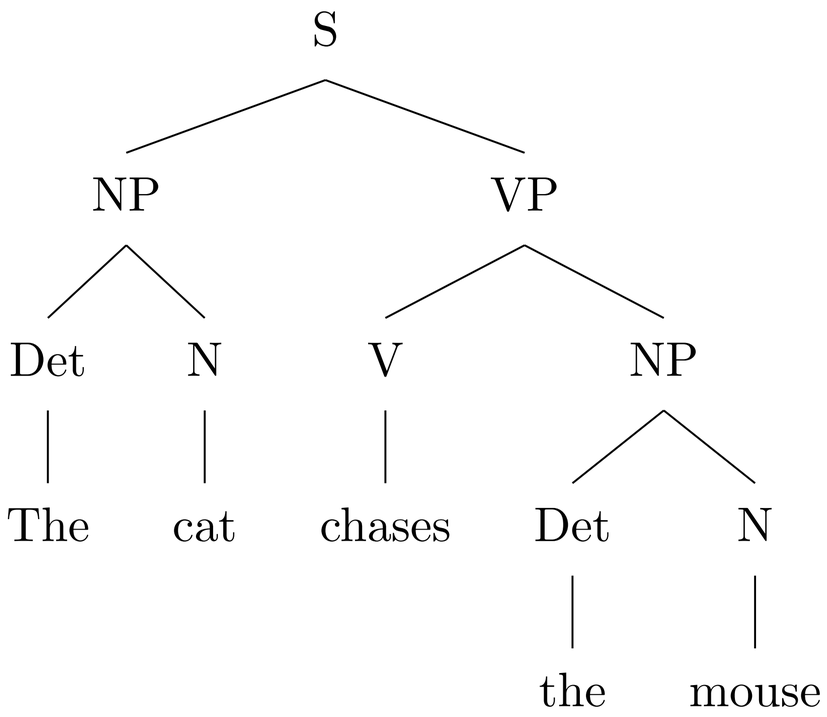
- Consider how you would represent the syntactic structure of sentence

The cat chases the mouse.

One way to represent syntax: Dependency Parsing

- Consider how you would represent the syntactic structure of sentence

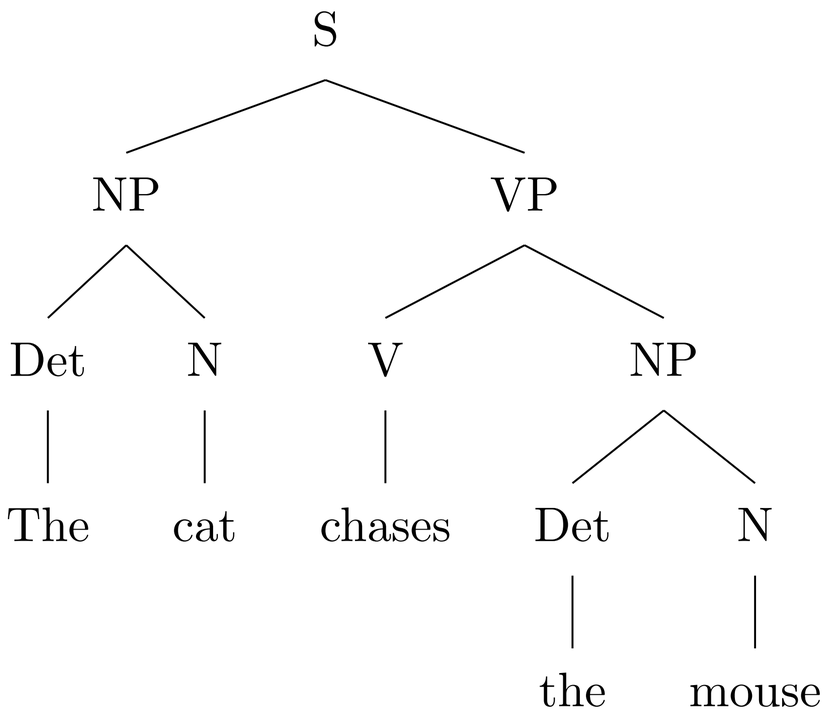
The cat chases the mouse.

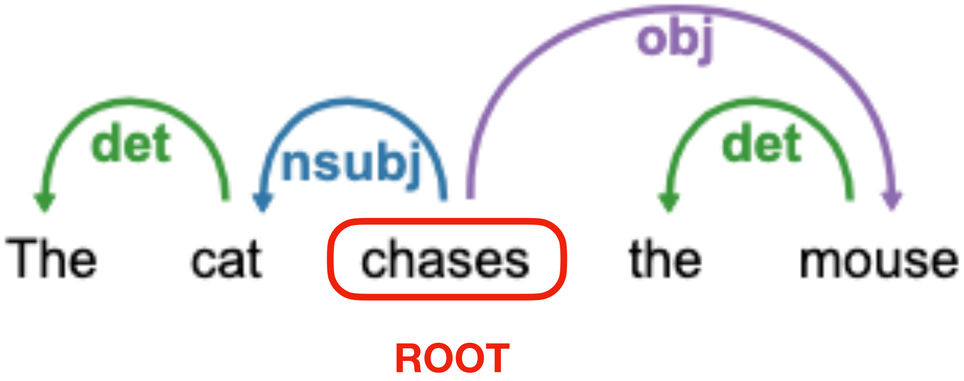


One way to represent syntax: Dependency Parsing

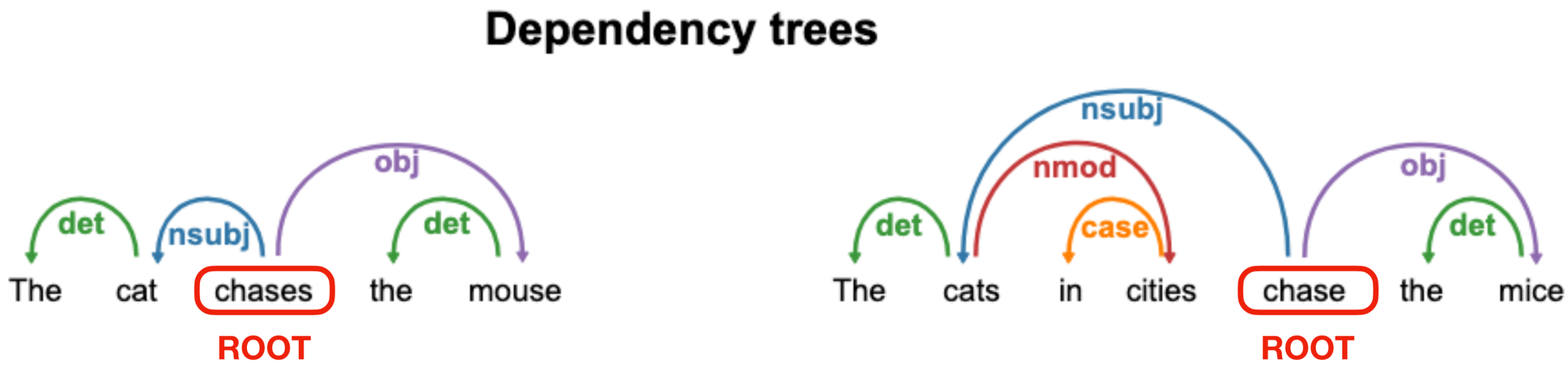
- Consider how you would represent the syntactic structure of sentence

The cat chases the mouse.



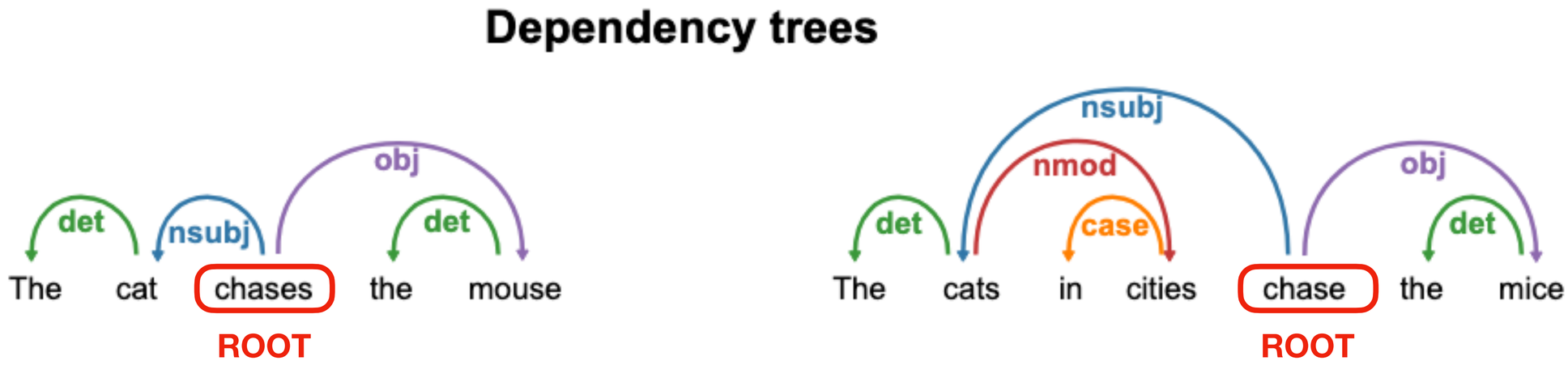


One way to represent syntax: Dependency Parsing



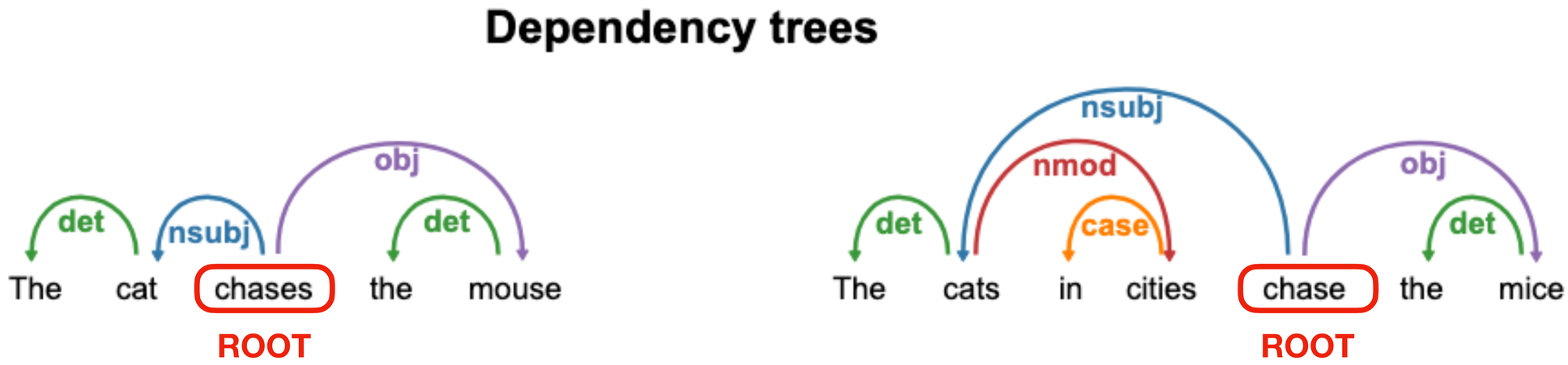
One way to represent syntax: Dependency Parsing

- Dependency parsing represents the grammatical relations of **<head, dependent>** pairs with a *directed, acyclic graph* (tree);
  - Exactly 1 root; every other word has exactly one incoming dependency link;

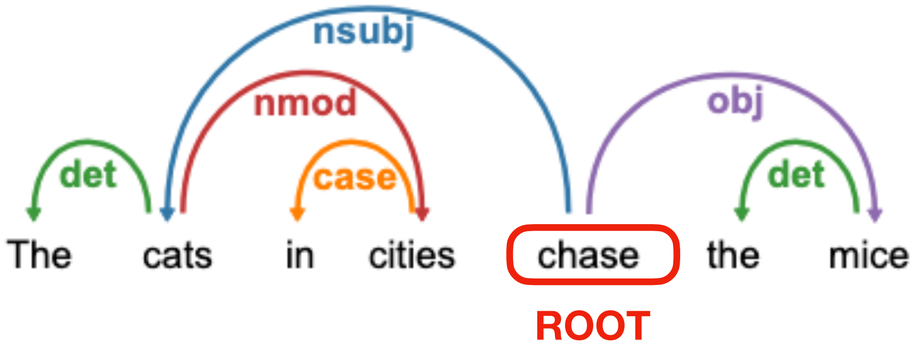


One way to represent syntax: Dependency Parsing

- Dependency parsing represents the grammatical relations of **<head, dependent>** pairs with a *directed, acyclic graph* (tree);
  - Exactly 1 root; every other word has exactly one incoming dependency link;
- A grammatical relation is the function that the dependent plays with respect to the head.
  - The head determines the grammatical role of the phrase.
  - An edge points from the head to the dependent.

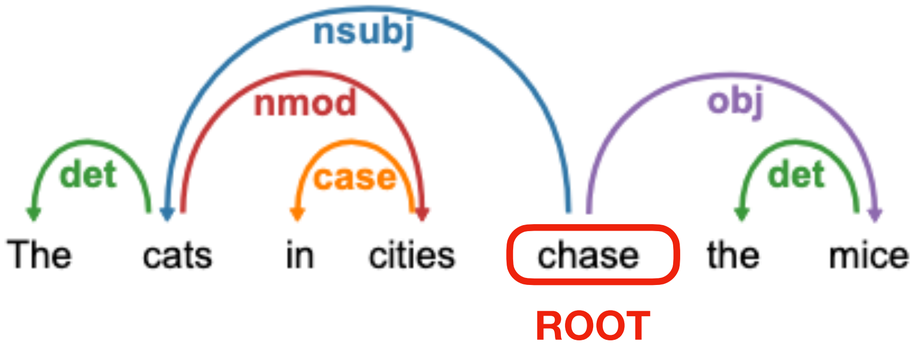


Notations


nsubj
nmod
obj
det
case
det
The
cats
in
cities
chase
the
mice
ROOT

Notations

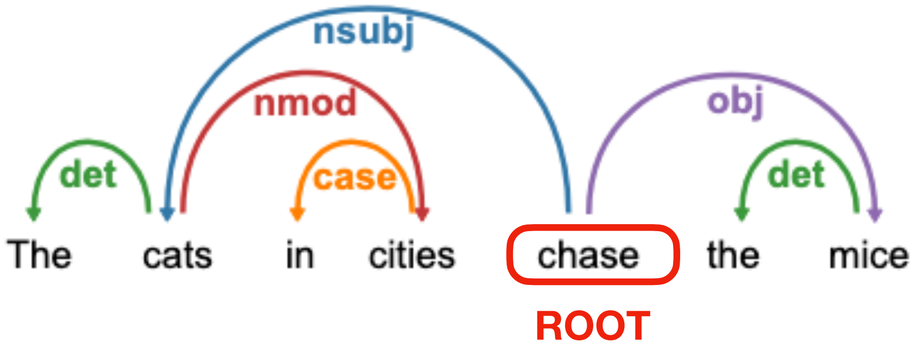
Information represented in a dependency tree:



Notations

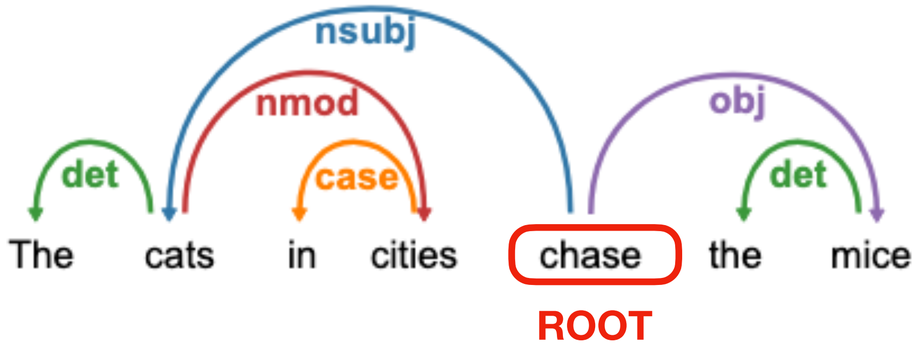
Information represented in a dependency tree:

- w_i: the i^{th} words in the sentence;



Notations

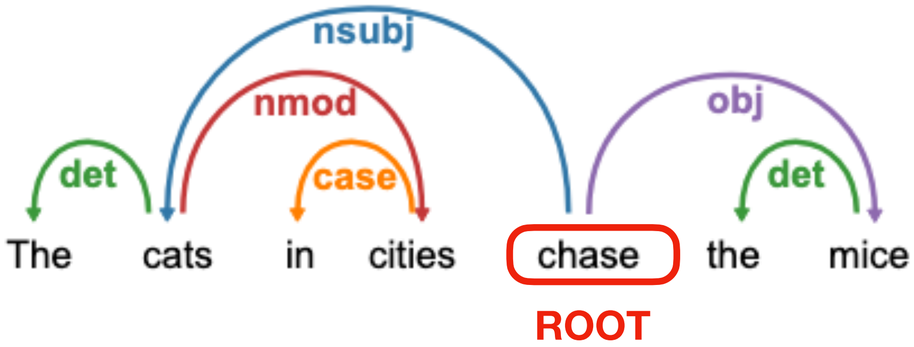
Information represented in a dependency tree:



- w_i: the i^{th} words in the sentence;
- $t : w_i, w_j \rightarrow C$ indicates the type of dependency relations between two words, where C is the set of syntactic types;

Notations

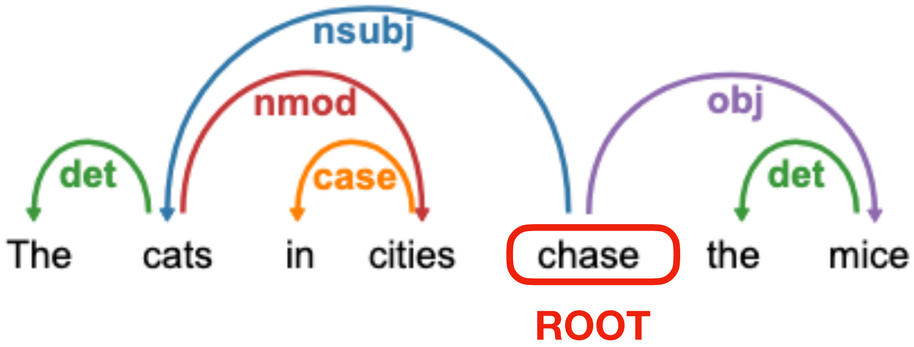
Information represented in a dependency tree:



- w_i: the i^{th} words in the sentence;
- $t : w_i, w_j \rightarrow C$ indicates the type of dependency relations between two words, where C is the set of syntactic types;
- $u : w_i, w_j \rightarrow \{w_i, w_j\}$ indicates the head word (i.e., direction) of the dependency relation between two words.

Notations

Information represented in a dependency tree:



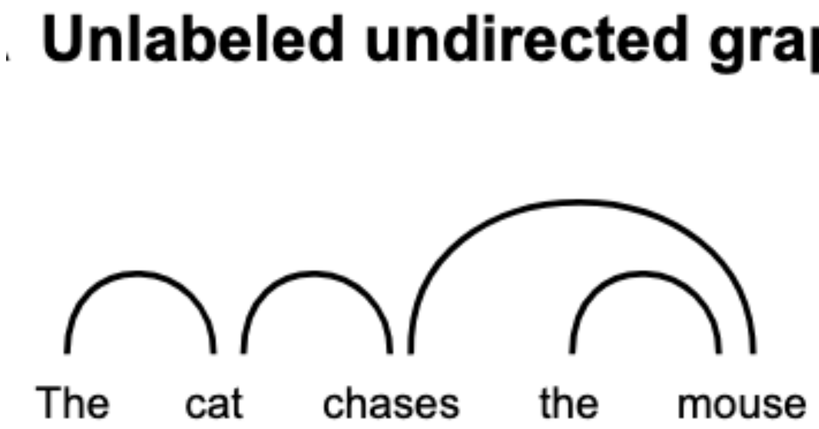
- w_i: the i^{th} words in the sentence;
- $t : w_i, w_j \rightarrow C$ indicates the type of dependency relations between two words, where C is the set of syntactic types;
- $u : w_i, w_j \rightarrow \{w_i, w_j\}$ indicates the head word (i.e., direction) of the dependency relation between two words.
- Also…. $d : w_i, w_j \rightarrow \mathbb{Z}^+$ is the syntactic distance between any pair of words;

Starting with distance

- For each pair of words, we want to measure the distance between them in the dependency tree without direction / type information.
- In the undirected graph, there is a unique shorest path between any two words — take **the number of edges on that path** to be the distance.

Starting with distance

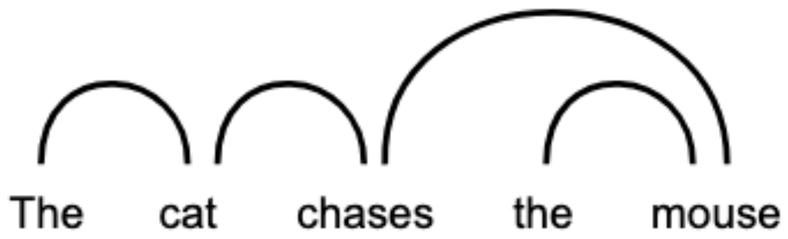
- For each pair of words, we want to measure the distance between them in the dependency tree without direction / type information.
- In the undirected graph, there is a unique shorest path between any two words — take **the number of edges on that path** to be the distance.



Starting with distance

- For each pair of words, we want to measure the distance between them in the dependency tree without direction / type information.
- In the undirected graph, there is a unique shorest path between any two words — take **the number of edges on that path** to be the distance.

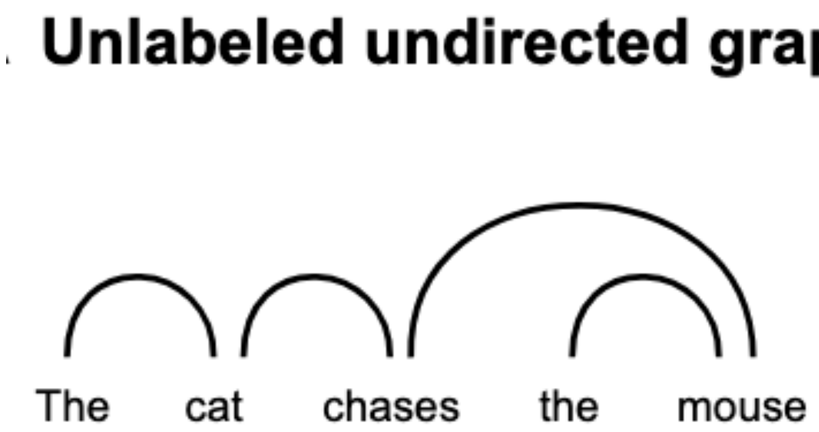
Unlabeled undirected graph



- $d(chases, cat) = d(chases, mouse) = 1$

Starting with distance

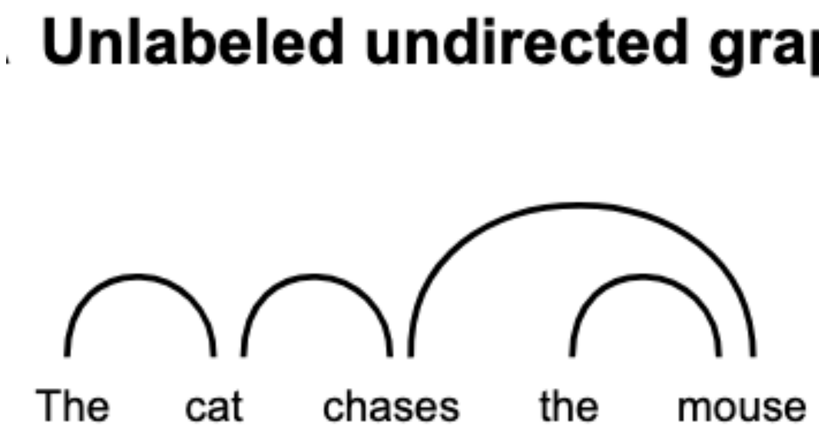
- For each pair of words, we want to measure the distance between them in the dependency tree without direction / type information.
- In the undirected graph, there is a unique shorest path between any two words — take **the number of edges on that path** to be the distance.



- $d(chases, cat) = d(chases, mouse) = 1$
- $d(cat, mouse) = 2$

Starting with distance

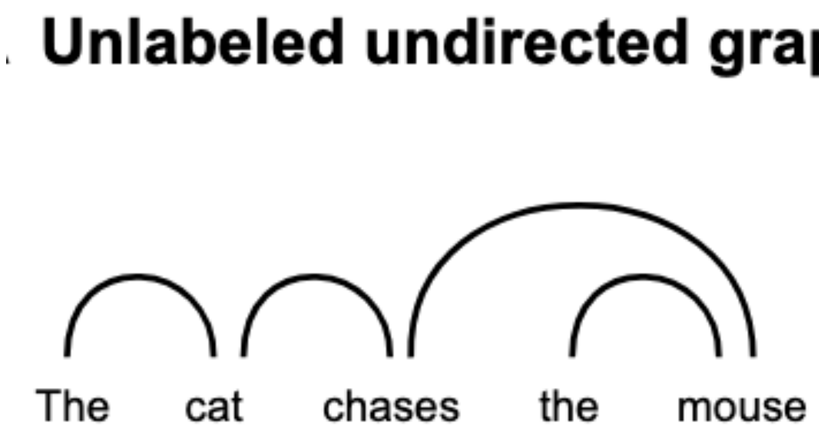
- For each pair of words, we want to measure the distance between them in the dependency tree without direction / type information.
- In the undirected graph, there is a unique shorest path between any two words — take **the number of edges on that path** to be the distance.



- $d(chases, cat) = d(chases, mouse) = 1$
- $d(cat, mouse) = 2$
- $d(The, cat) = 1$, $d(The, mouse) = 3$

Starting with distance

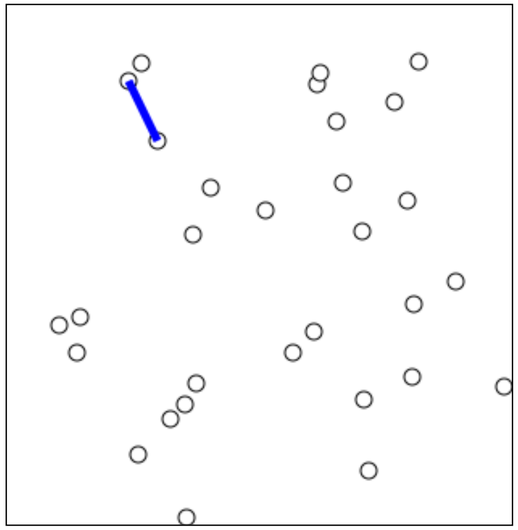
- For each pair of words, we want to measure the distance between them in the dependency tree without direction / type information.
- In the undirected graph, there is a unique shorest path between any two words — take **the number of edges on that path** to be the distance.



- $d(chases, cat) = d(chases, mouse) = 1$
- $d(cat, mouse) = 2$
- $d(The, cat) = 1$, $d(The, mouse) = 3$
- $d(The, the) = 4$

Reconstructing dependency tree given distances

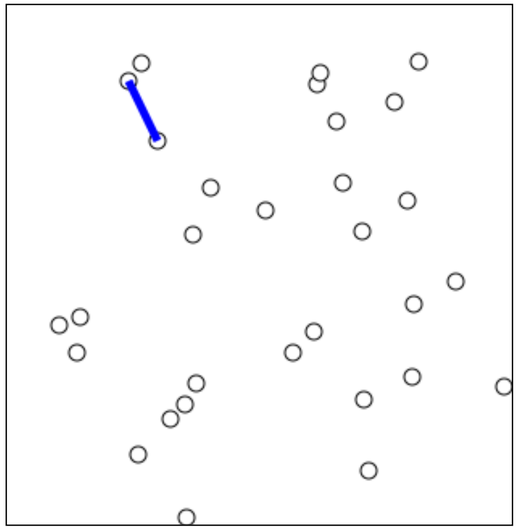
via Prim's minimum spanning tree algorithm



Reconstructing dependency tree given distances

via Prim's minimum spanning tree algorithm

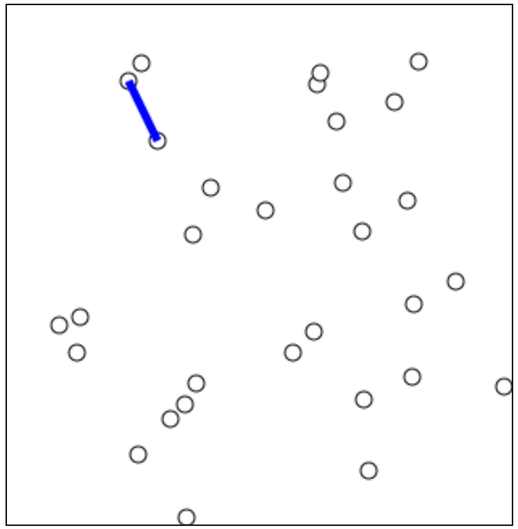
- Prim's algorithm greedily finds a minimum spanning tree (MST) for a weighted undirected graph:



Reconstructing dependency tree given distances

via Prim's minimum spanning tree algorithm

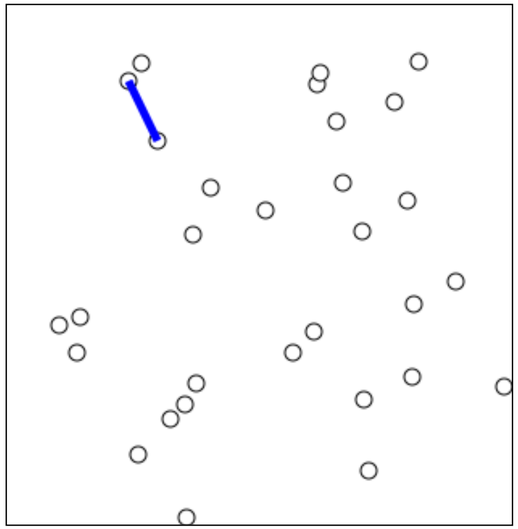
- Prim's algorithm greedily finds a minimum spanning tree (MST) for a weighted undirected graph:
  - MST: a subset of edges that connect all vertices (spanning) without cycle (tree), minimizing the total weight of the edges (minimum).



Reconstructing dependency tree given distances

via Prim's minimum spanning tree algorithm

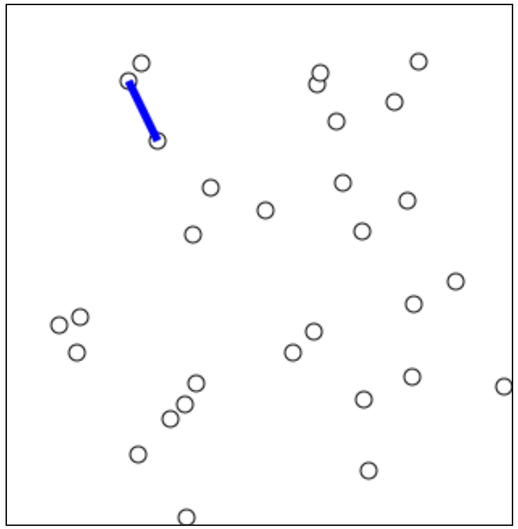
- Prim's algorithm greedily finds a minimum spanning tree (MST) for a weighted undirected graph:
  - MST: a subset of edges that connect all vertices (spanning) without cycle (tree), minimizing the total weight of the edges (minimum).
- Idea sketch:



Reconstructing dependency tree given distances

via Prim's minimum spanning tree algorithm

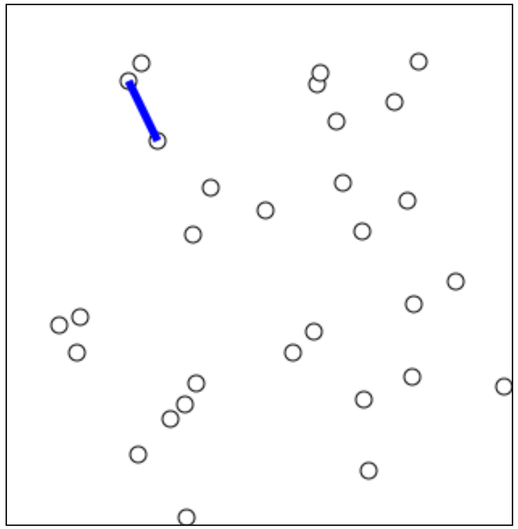
- Prim's algorithm greedily finds a minimum spanning tree (MST) for a weighted undirected graph:
  - MST: a subset of edges that connect all vertices (spanning) without cycle (tree), minimizing the total weight of the edges (minimum).
- Idea sketch:
  1. Randomly pick a vertex;



Reconstructing dependency tree given distances

via Prim's minimum spanning tree algorithm

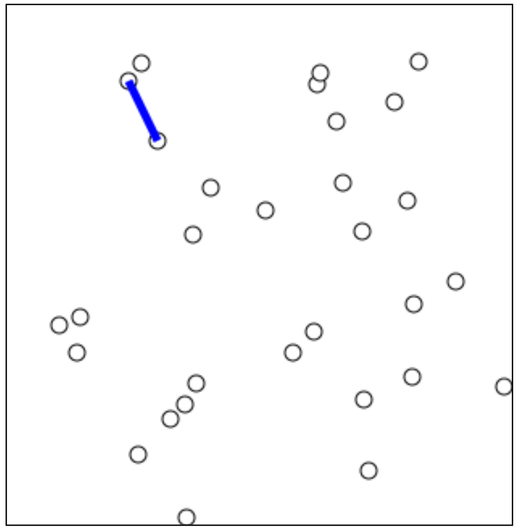
- Prim's algorithm greedily finds a minimum spanning tree (MST) for a weighted undirected graph:
  - MST: a subset of edges that connect all vertices (spanning) without cycle (tree), minimizing the total weight of the edges (minimum).
- Idea sketch:
  1. Randomly pick a vertex;
  2. Grow the tree by one edge: among edges that connect the current tree to vertices not yet in the tree, find the one with minimum edge, add to the tree.



Reconstructing dependency tree given distances

via Prim's minimum spanning tree algorithm

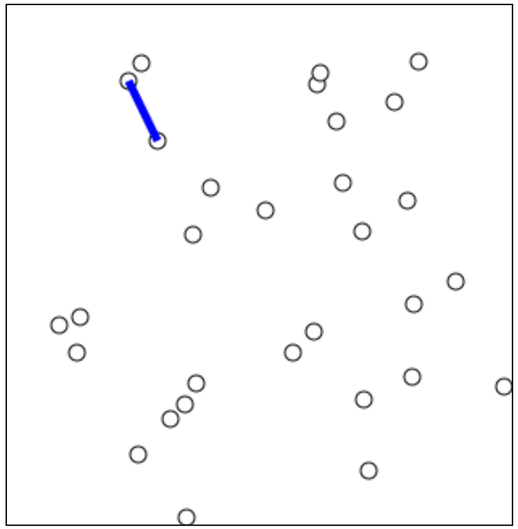
- Prim's algorithm greedily finds a minimum spanning tree (MST) for a weighted undirected graph:
  - MST: a subset of edges that connect all vertices (spanning) without cycle (tree), minimizing the total weight of the edges (minimum).
- Idea sketch:
  1. Randomly pick a vertex;
  2. Grow the tree by one edge: among edges that connect the current tree to vertices not yet in the tree, find the one with minimum edge, add to the tree.
  3. Keep track of vertices already in the tree; repeat step 2 until all vertices are in.



Reconstructing dependency tree given distances

via Prim's minimum spanning tree algorithm

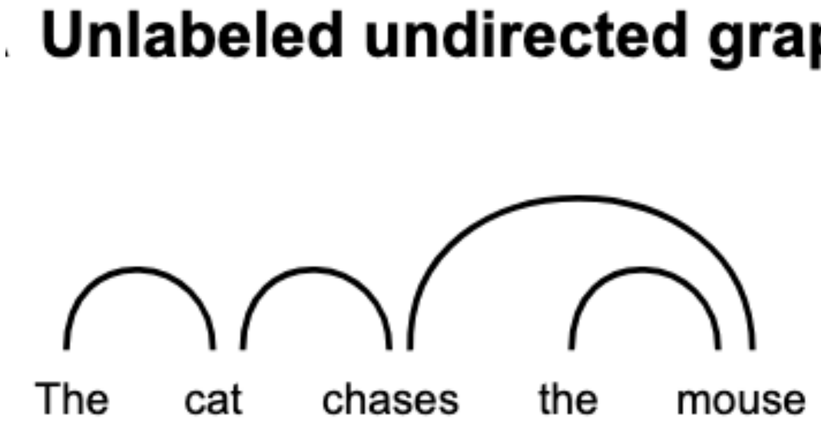
- Prim's algorithm greedily finds a minimum spanning tree (MST) for a weighted undirected graph:
  - MST: a subset of edges that connect all vertices (spanning) without cycle (tree), minimizing the total weight of the edges (minimum).
- Idea sketch:
  1. Randomly pick a vertex;
  2. Grow the tree by one edge: among edges that connect the current tree to vertices not yet in the tree, find the one with minimum edge, add to the tree.
  3. Keep track of vertices already in the tree; repeat step 2 until all vertices are in.



Applying Prim’s Algorithm to Dependency Trees

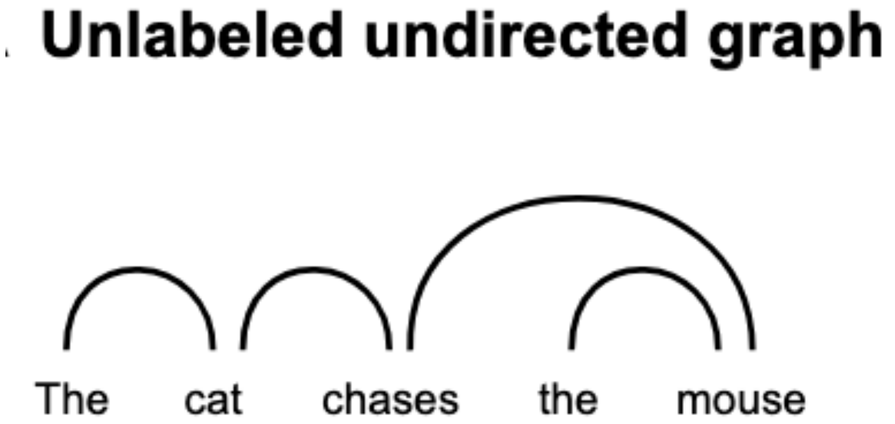
Applying Prim's Algorithm to Dependency Trees

- Are dependency trees spanning trees?



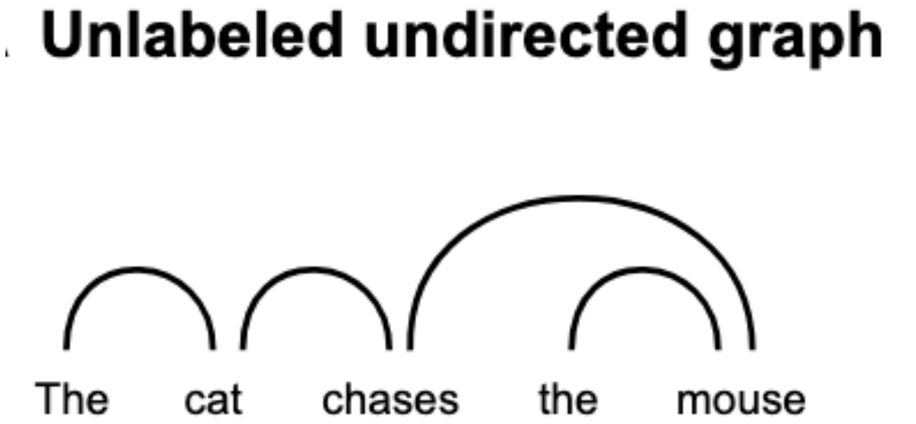
Applying Prim's Algorithm to Dependency Trees

- Are dependency trees spanning trees?
  - *yes*: *all words are linked without cycles!*



Applying Prim's Algorithm to Dependency Trees

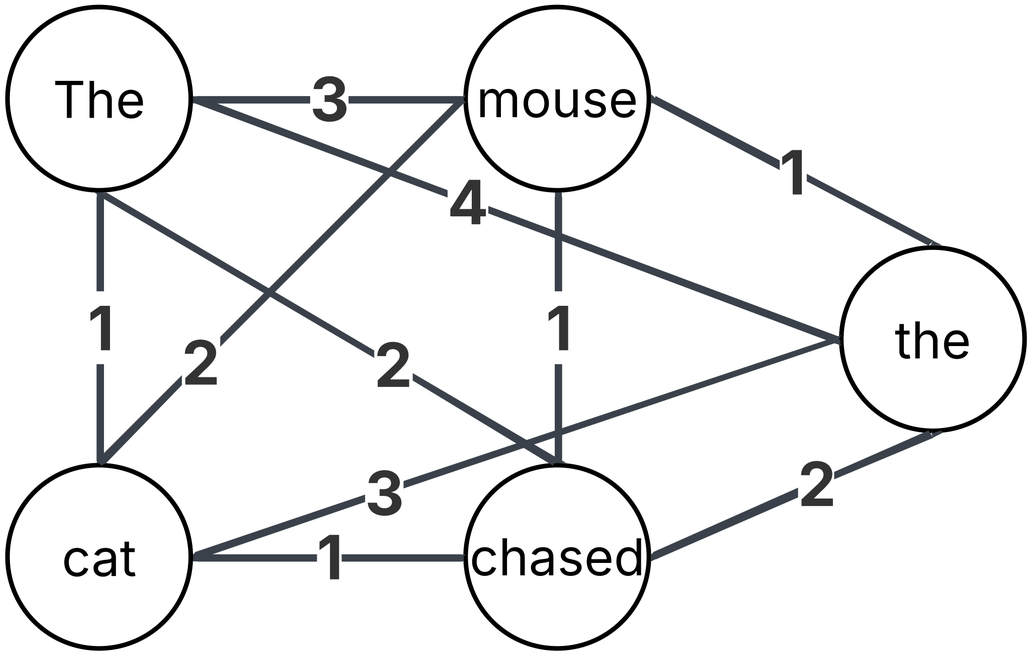
- Are dependency trees spanning trees?
  - *yes*: *all words are linked without cycles!*



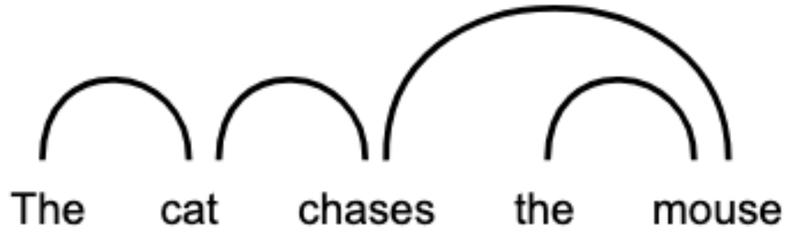
- d(*cat*, *mouse*) = d(*The*, *chases*) = d(*chases*, *the*) = 2
- d(the, *cat*) = d(*The*, *mouse*) = 3
- d(*The*, *the*) = 4

Applying Prim's Algorithm to Dependency Trees

- Are dependency trees spanning trees?
  - *yes*: *all words are linked without cycles!*

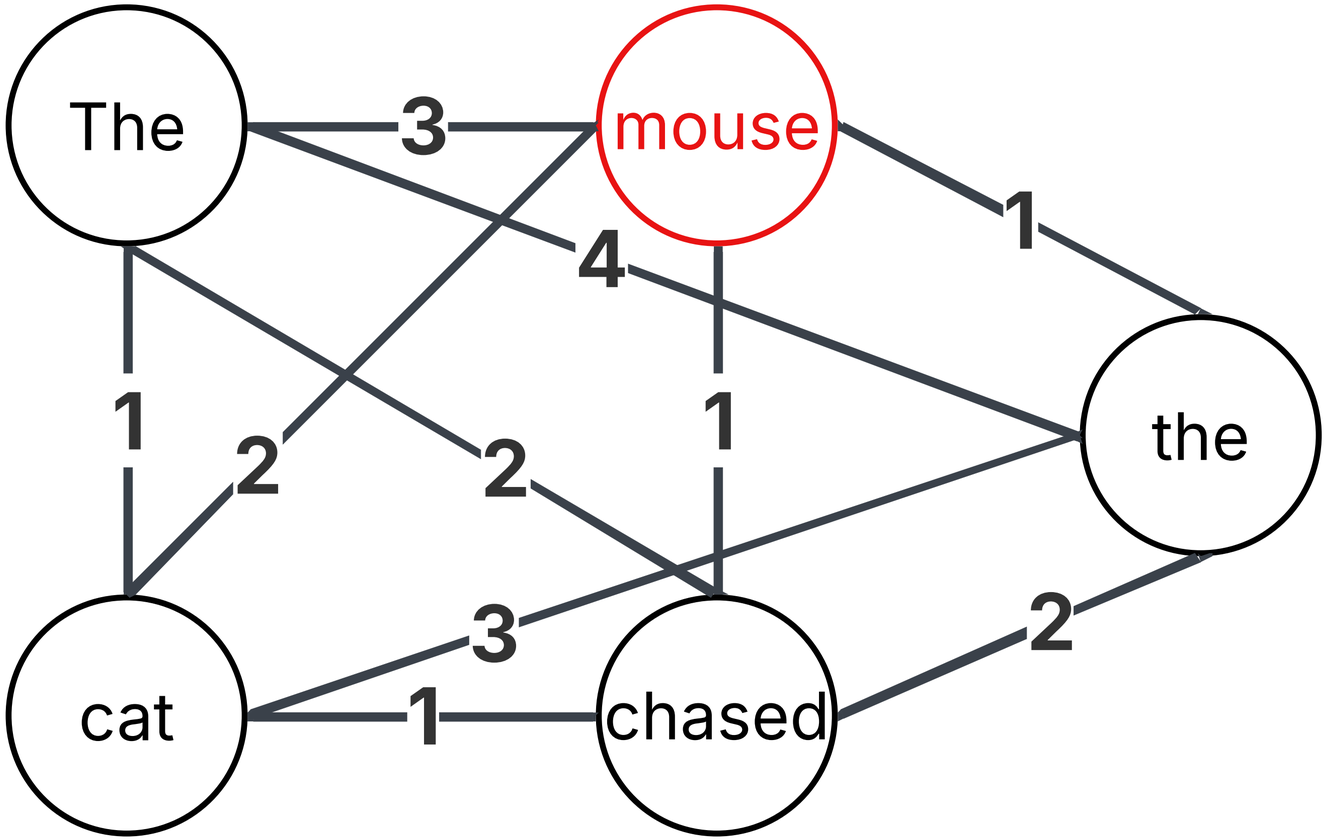


Unlabeled undirected graph

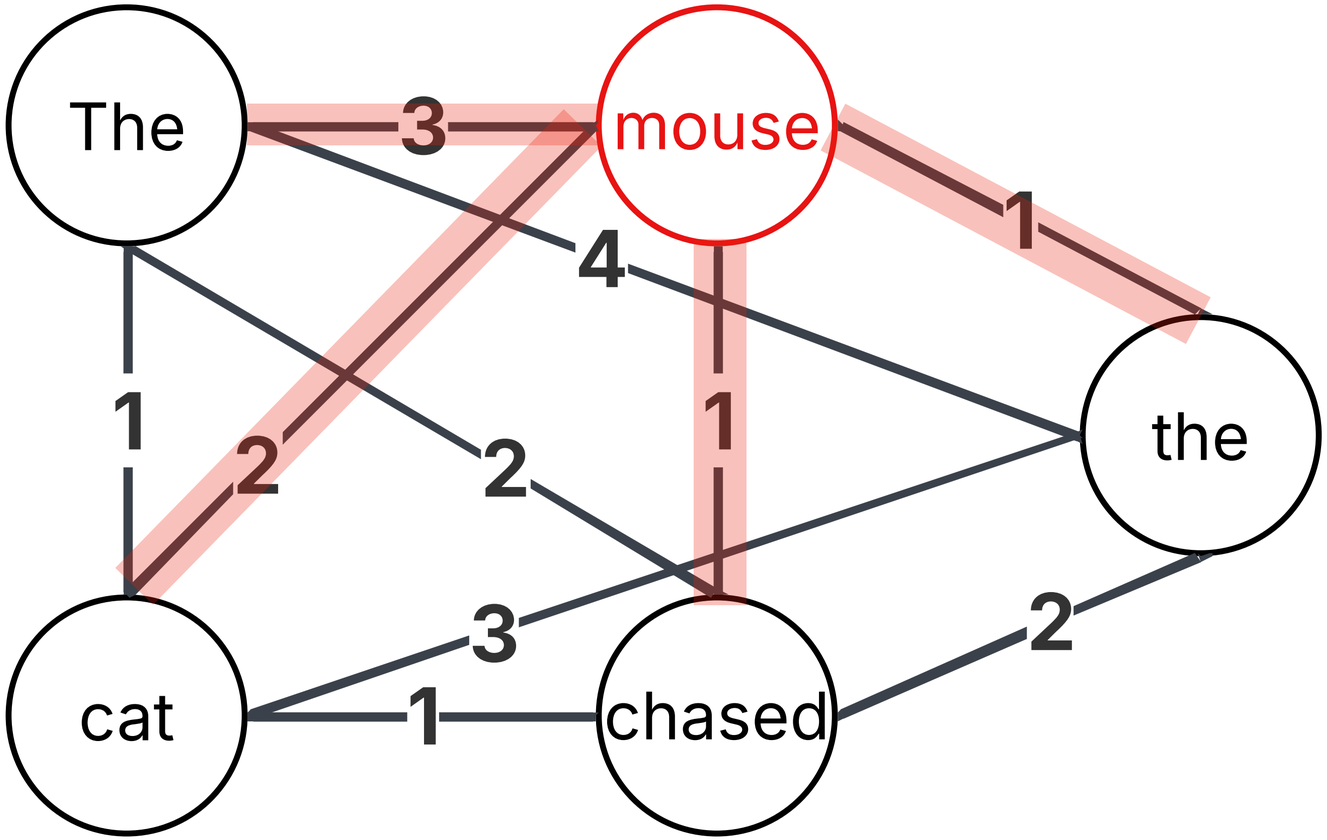


- d(*cat*, *mouse*) = d(*The*, *chases*) = d(*chases*, *the*) = 2
- d(*the*, *cat*) = d(*The*, *mouse*) = 3
- d(*The*, *the*) = 4

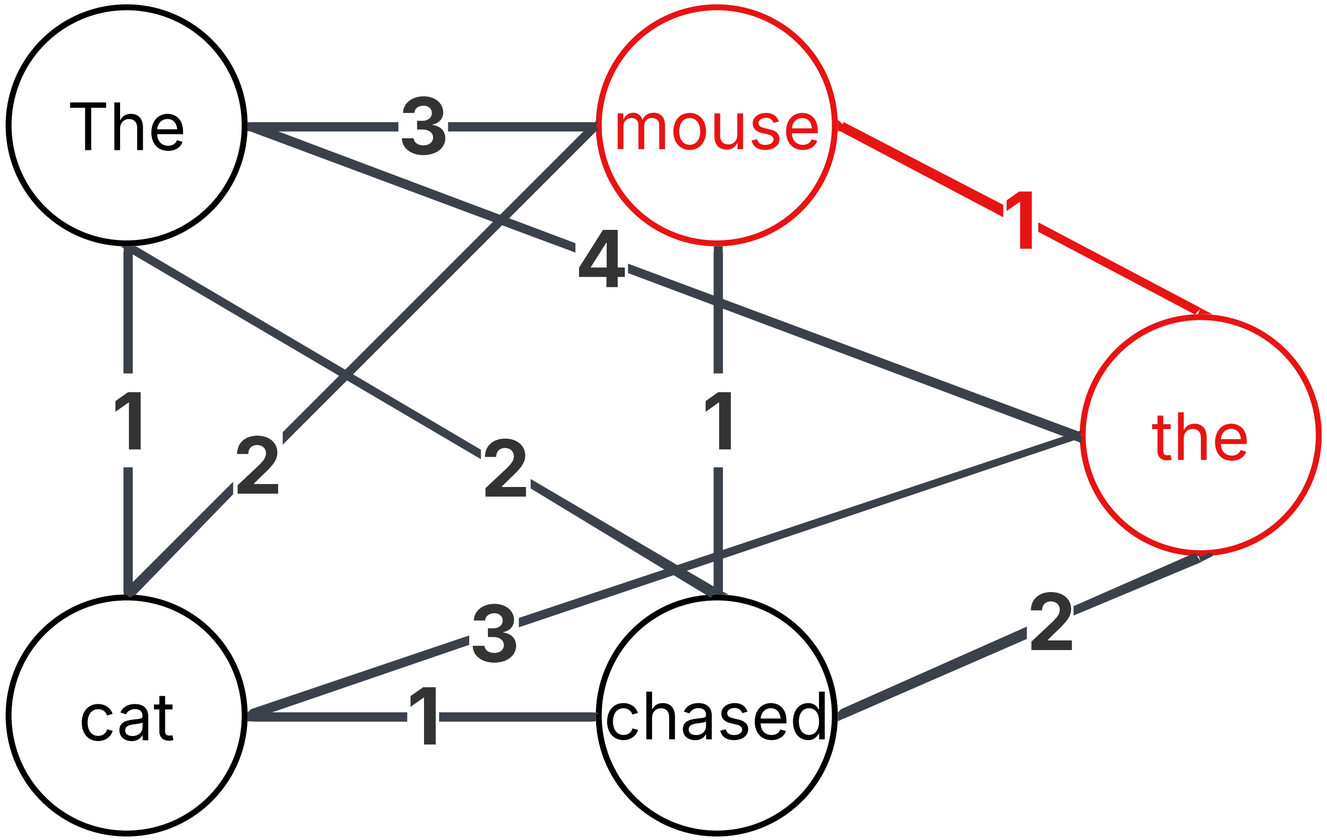
Applying Prim's Algorithm to Dependency Trees



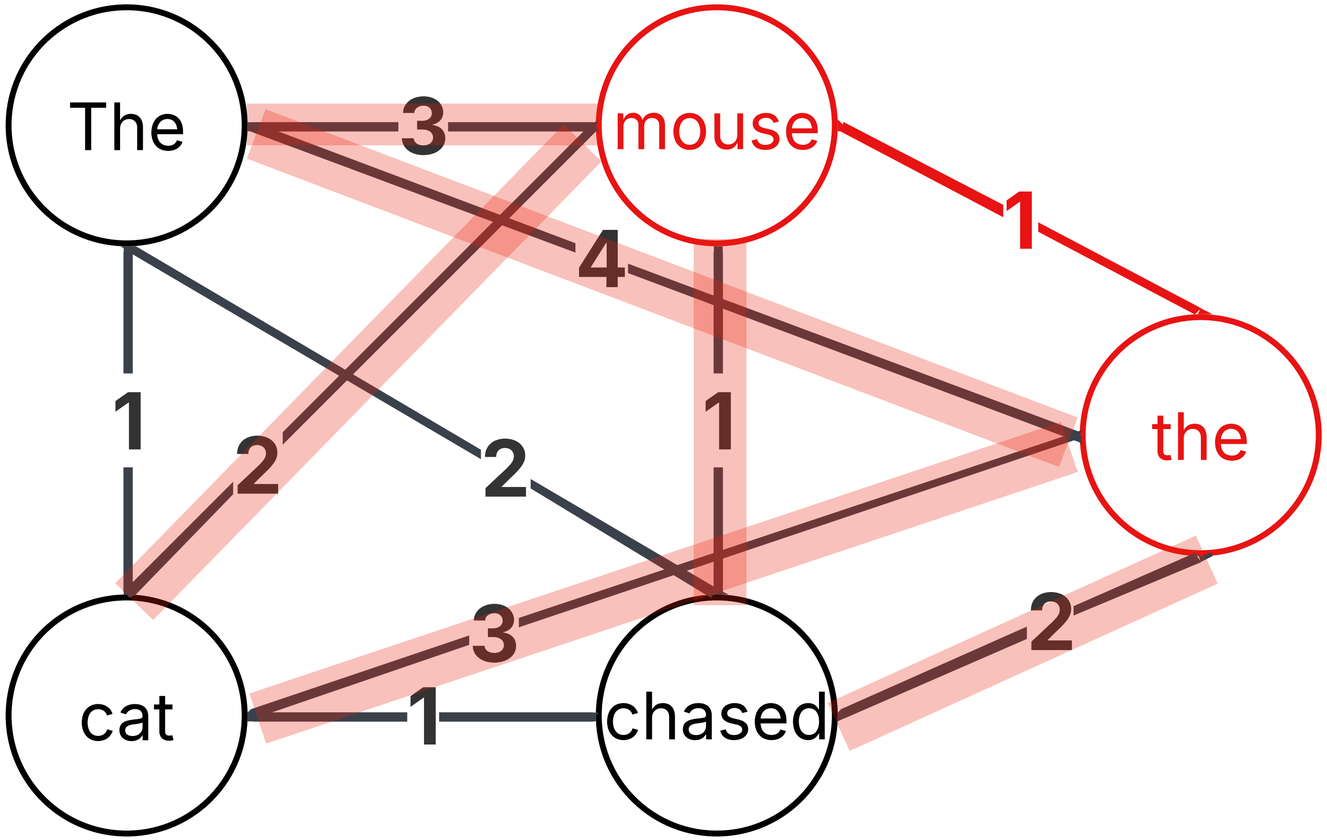
Applying Prim's Algorithm to Dependency Trees



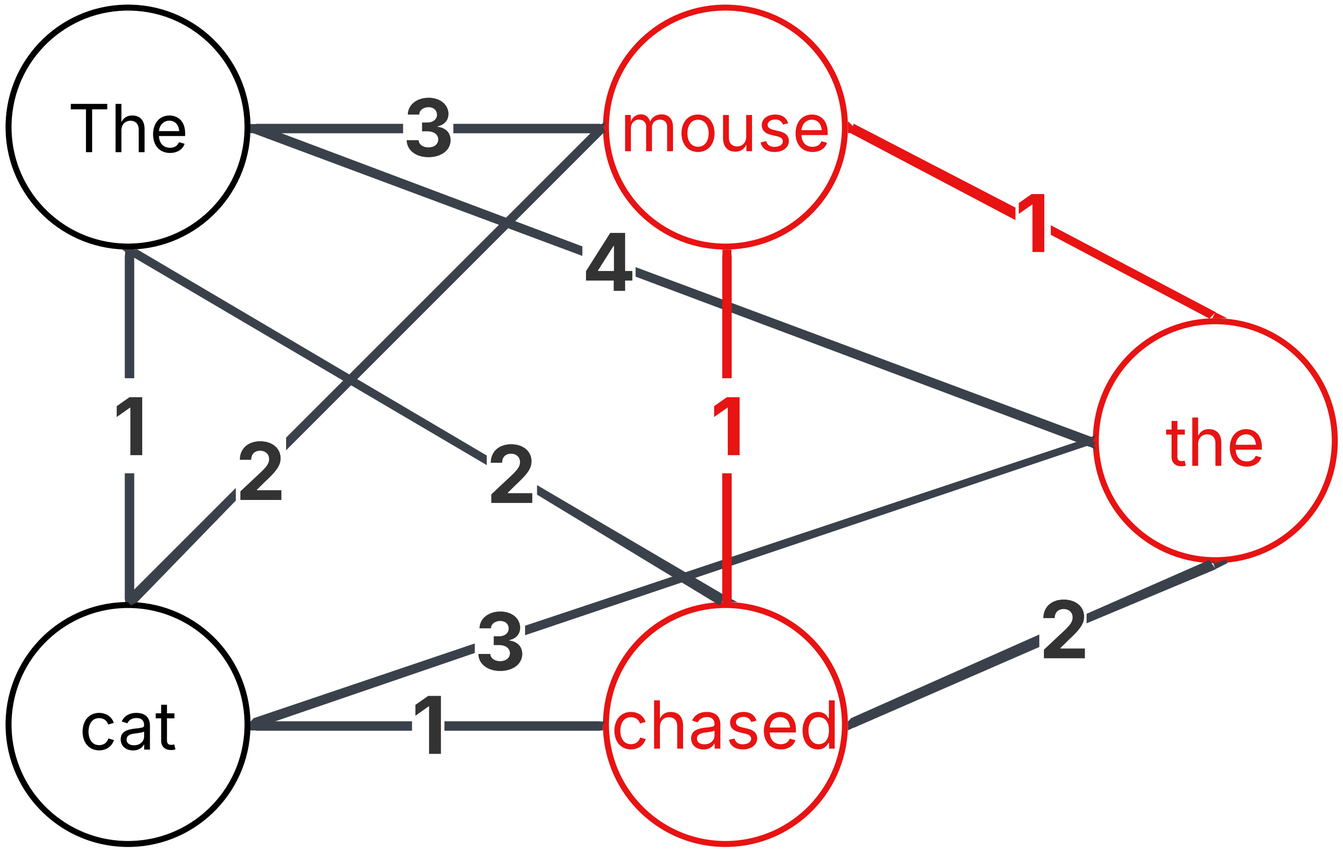
Applying Prim's Algorithm to Dependency Trees



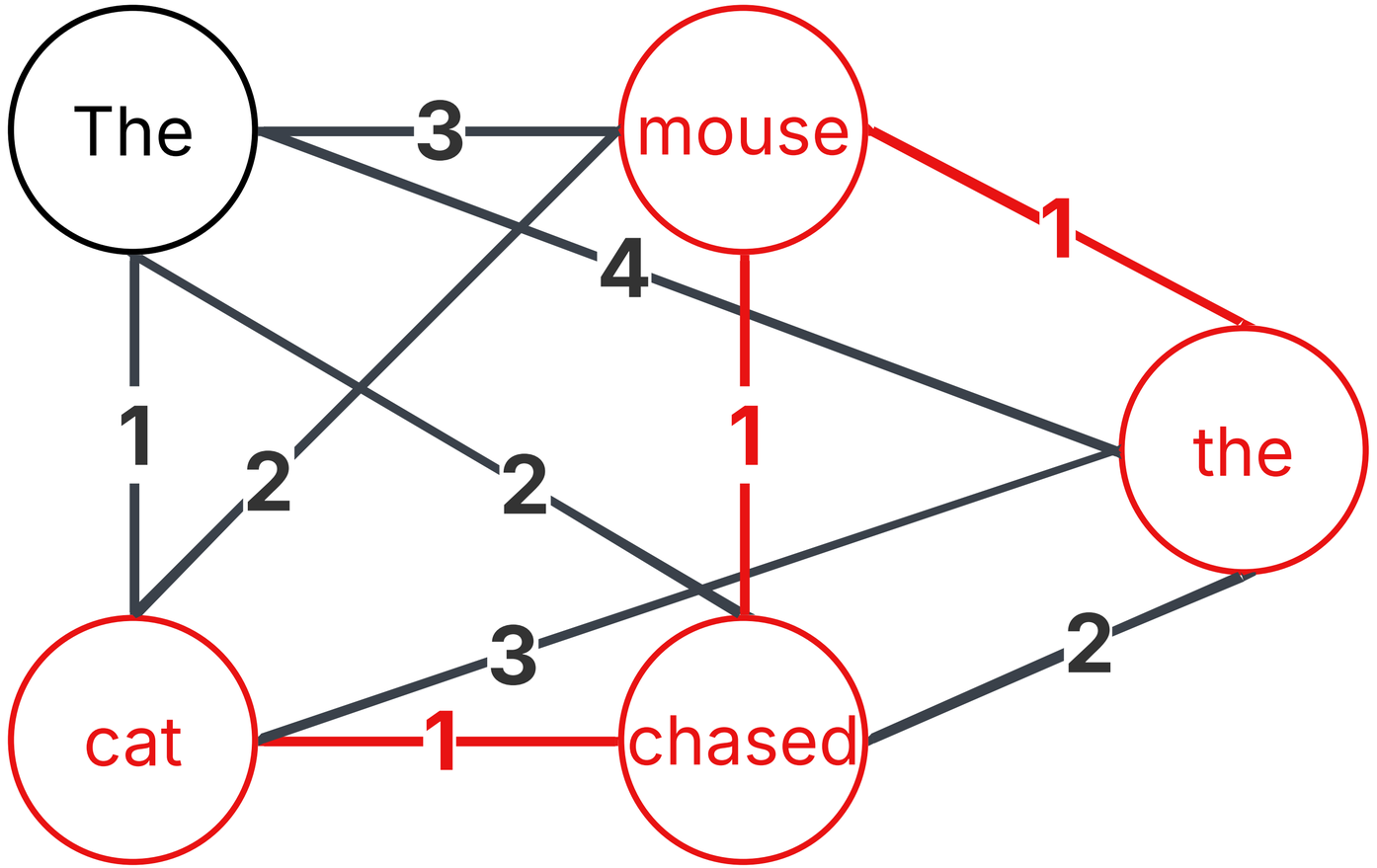
Applying Prim's Algorithm to Dependency Trees



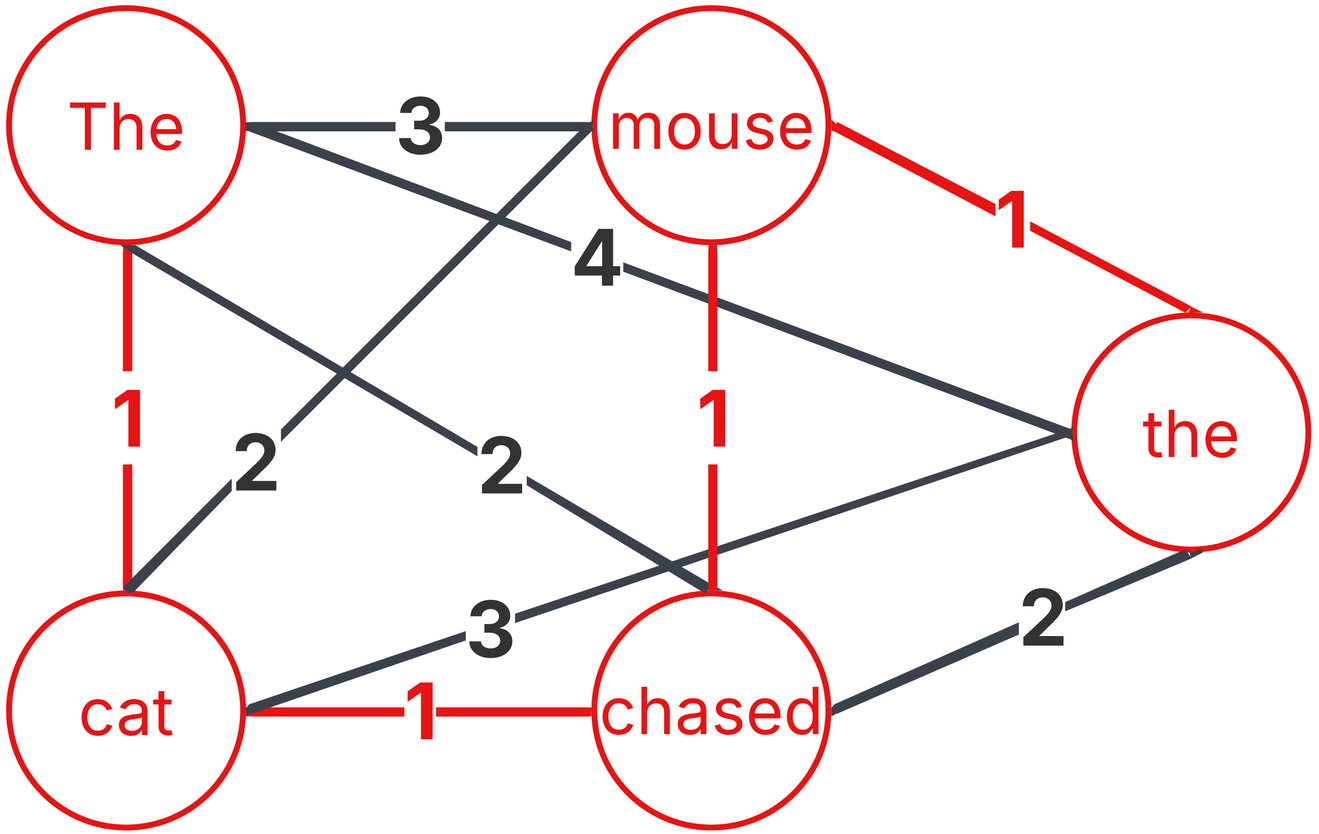
Applying Prim's Algorithm to Dependency Trees



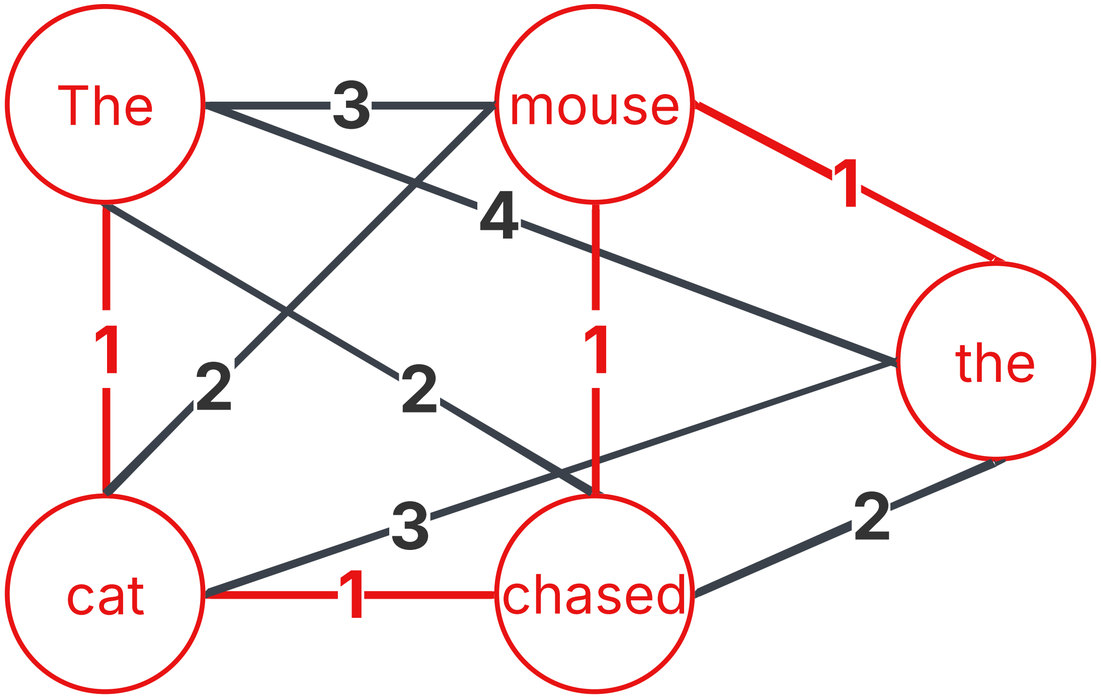
Applying Prim's Algorithm to Dependency Trees

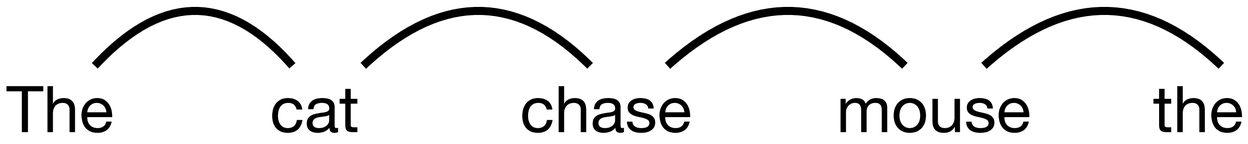


Applying Prim's Algorithm to Dependency Trees

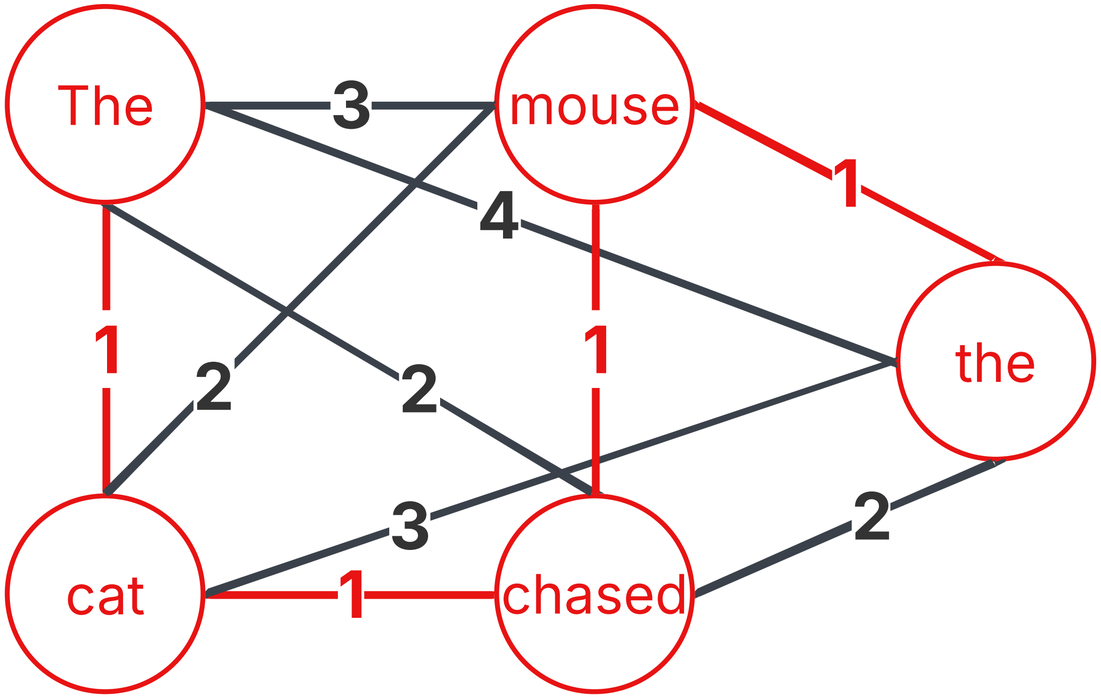


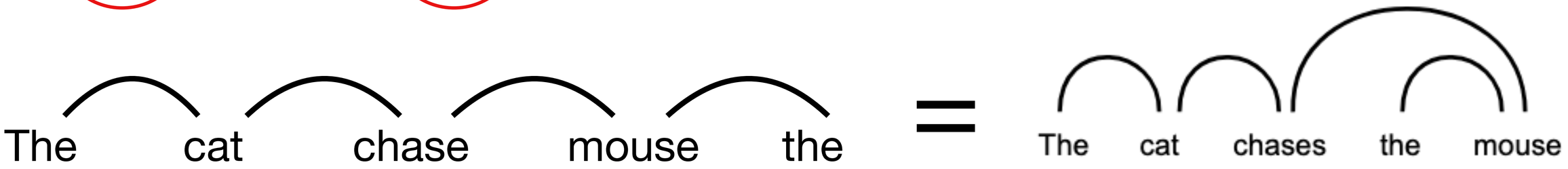
Applying Prim's Algorithm to Dependency Trees



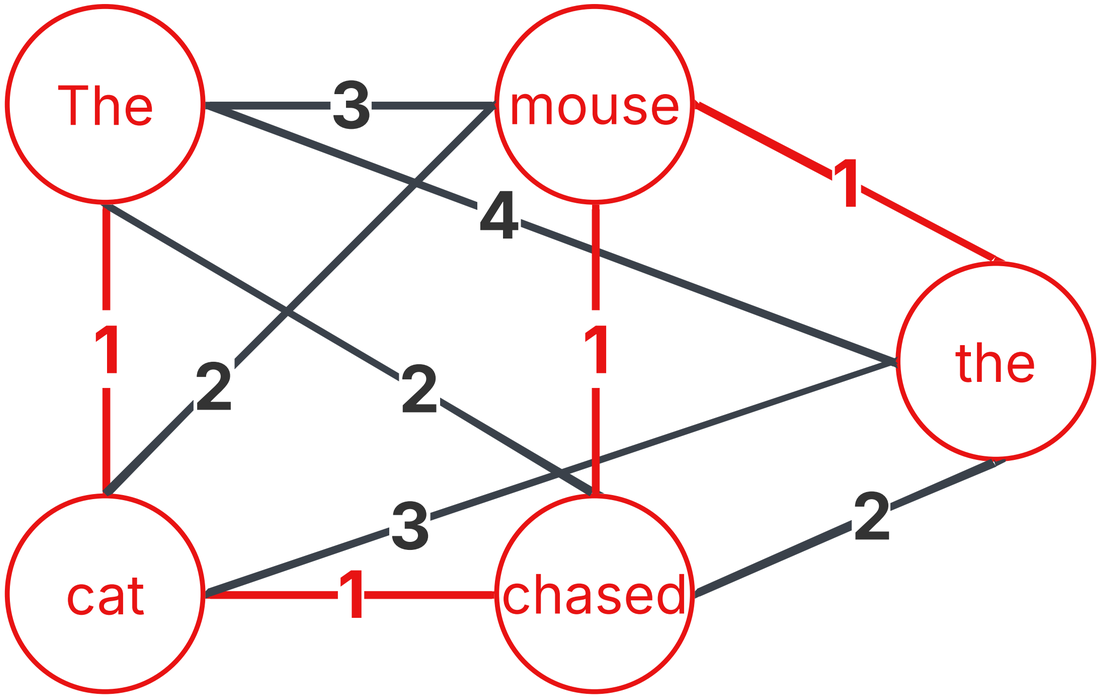


Applying Prim's Algorithm to Dependency Trees

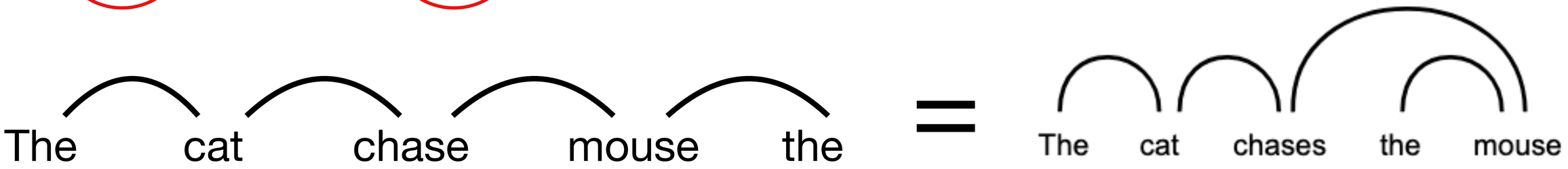




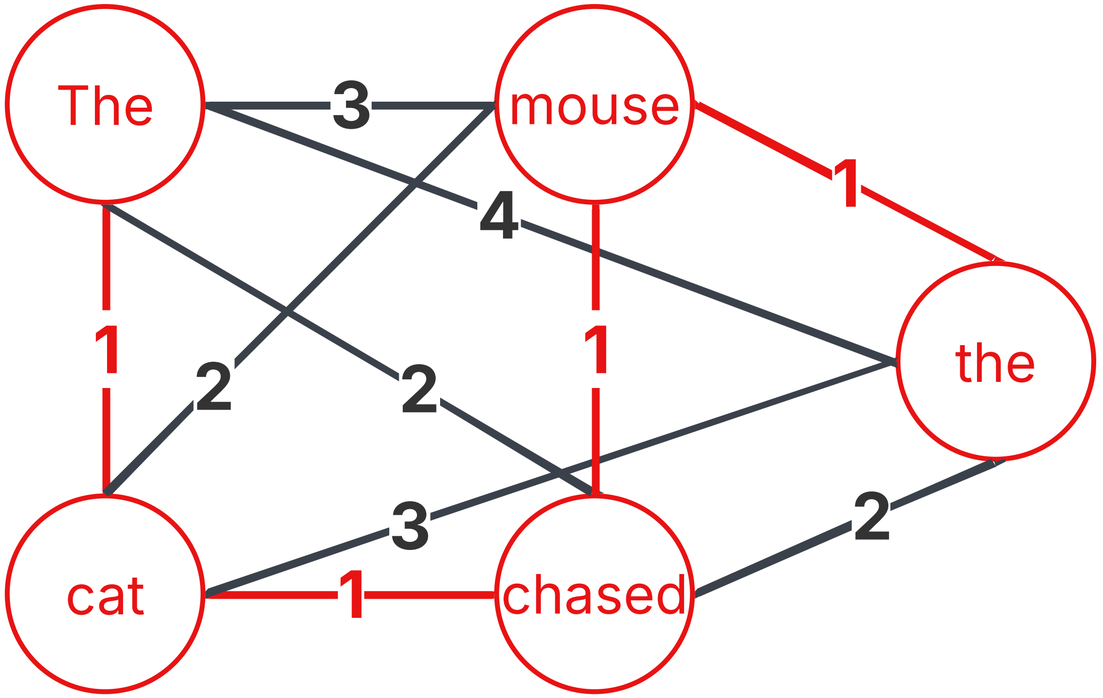
Applying Prim's Algorithm to Dependency Trees



Takeaways:

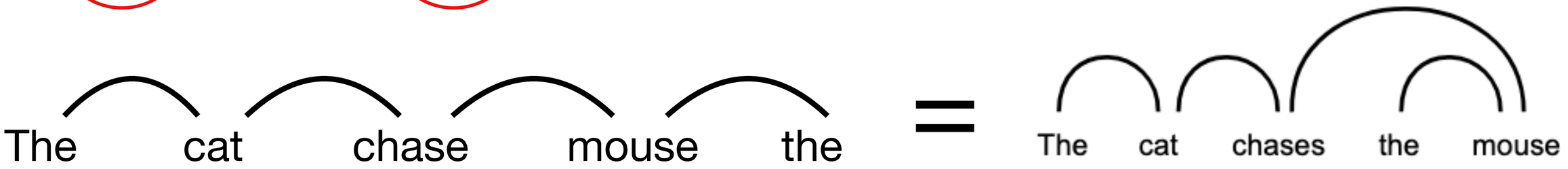


Applying Prim's Algorithm to Dependency Trees

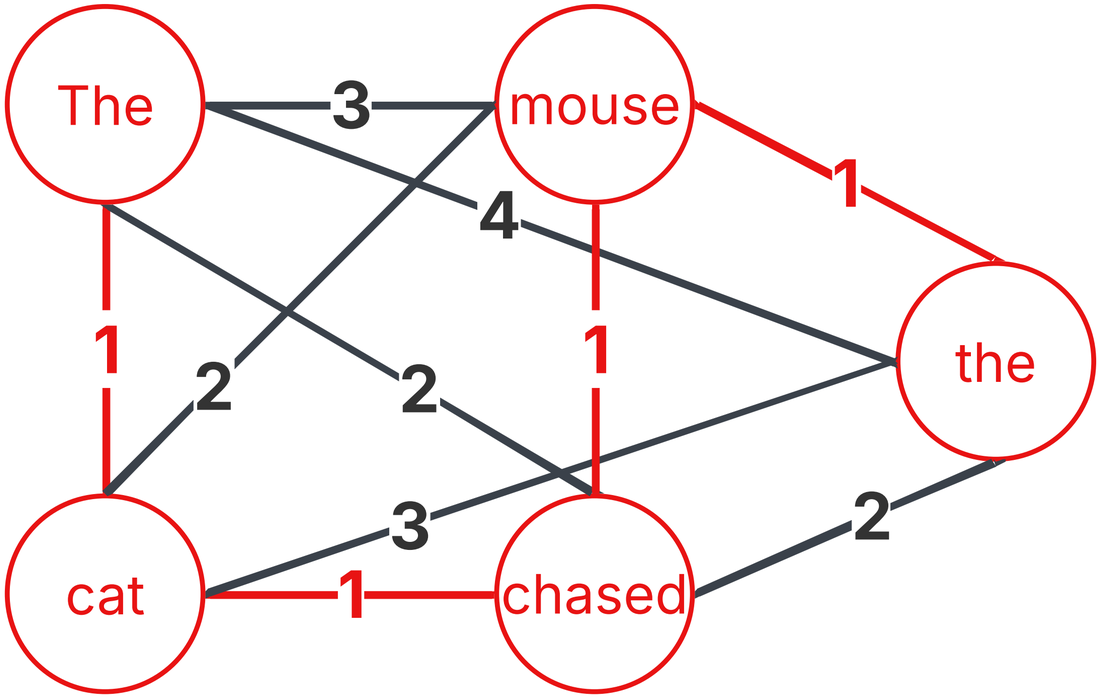


Takeaways:

- *Given distance information of all pairs of words in a sentence, we can reconstruct the dependency tree;*

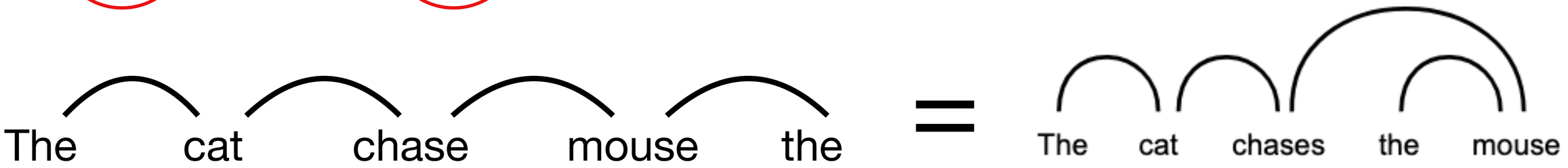


Applying Prim's Algorithm to Dependency Trees



Takeaways:

- *Given distance information of all pairs of words in a sentence, we can reconstruct the dependency tree;*
- *Linear order != Syntactic order: word directly lined in the dependency tree should be small distances.*



Hewitt & Manning’s Solution

Hewitt & Manning's Solution

- Given $h_i, h_j \in \mathbb{R}^k$ as the hidden states for w_i, w_j at some layer;

Hewitt & Manning's Solution

- Given $h_i, h_j \in \mathbb{R}^k$ as the hidden states for w_i, w_j at some layer;
- Goal = to find **linear** operator $\hat{d}$ such that $\hat{d}(h_i, h_j) \approx d(w_i, w_j)$

Hewitt & Manning's Solution

- Given $h_i, h_j \in \mathbb{R}^k$ as the hidden states for w_i, w_j at some layer;
- Goal = to find **linear** operator $\hat{d}$ such that $\hat{d}(h_i, h_j) \approx d(w_i, w_j)$
- Solution: $\hat{d}(h_i, h_j) := \|\mathbf{B}(h_i - h_j)\|^2$, where $\mathbf{B} \in \mathbb{R}^{m \times k}$ is a linear projection;
  - i.e., $\mathbf{B}$ projects the activation space into a lower dimension subspace for where syntactic (distance) information is represented.
  - the choice of m decides how small that projected subspace is — spolier: Hewitt & Manning found something interesting here!

Hewitt & Manning’s Solution, cont.

Hewitt & Manning's Solution, cont.

- Given a set of dependency parsed sentences, extract the set of all pairs of words Ω and distances between them;

Hewitt & Manning's Solution, cont.

- Given a set of dependency parsed sentences, extract the set of all pairs of words Ω and distances between them;

- A supervised learning objective: optimize $\mathbf{B}$ (the only learned parameter) to minimize the following loss:

Hewitt & Manning's Solution, cont.

- Given a set of dependency parsed sentences, extract the set of all pairs of words Ω and distances between them;

- A supervised learning objective: optimize $\mathbf{B}$ (the only learned parameter) to minimize the following loss:

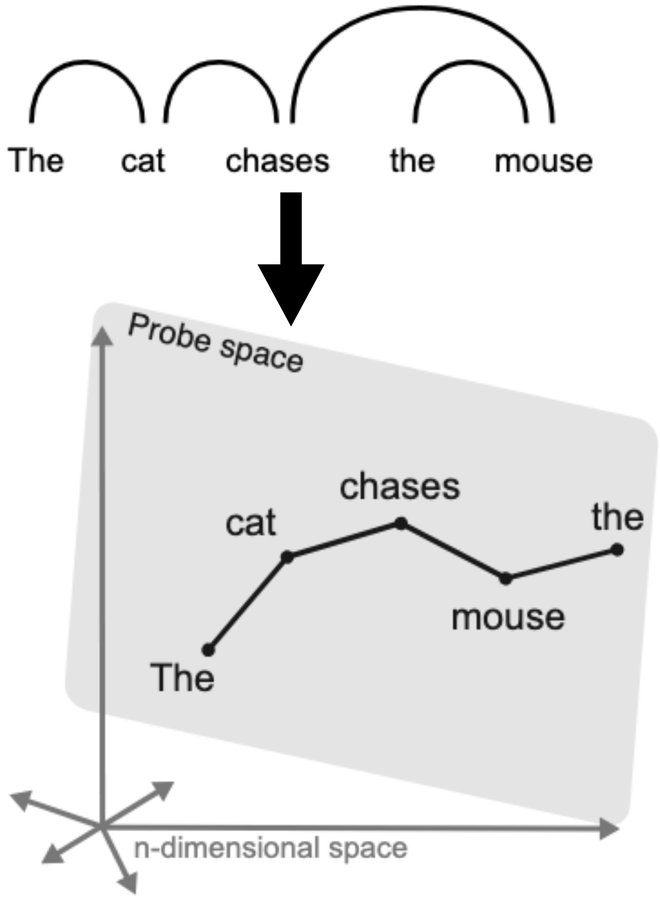
$$\mathscr{L}(\mathbf{B}) = \frac{1}{|\Omega|} \sum_{(w_i, w_j) \in \Omega} |\, d(w_i, w_j) - \hat{d}(h_i, h_j)^2 \,|$$

  - i.e., for all pairs of words, minimize the difference between the true distance $d(w_i, w_j)$ and the square of projected distance $\|\mathbf{B}(h_i - h_j)\|^2$;

  - normalized by the number of word pairs, aka sentence length squared.

Recap

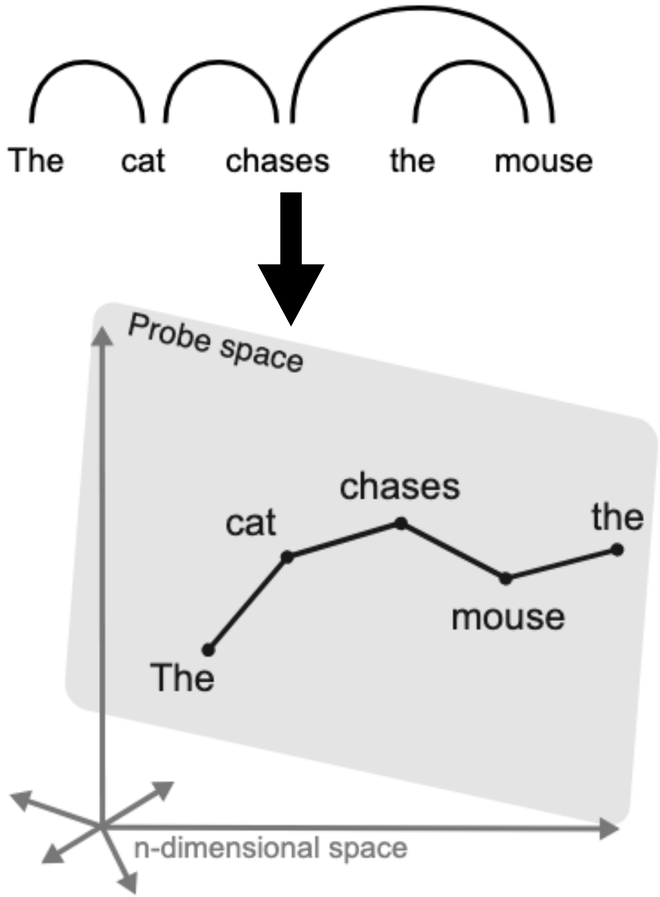
The structural probe goes beyond local probing classifiers



Recap

The structural probe goes beyond local probing classifiers

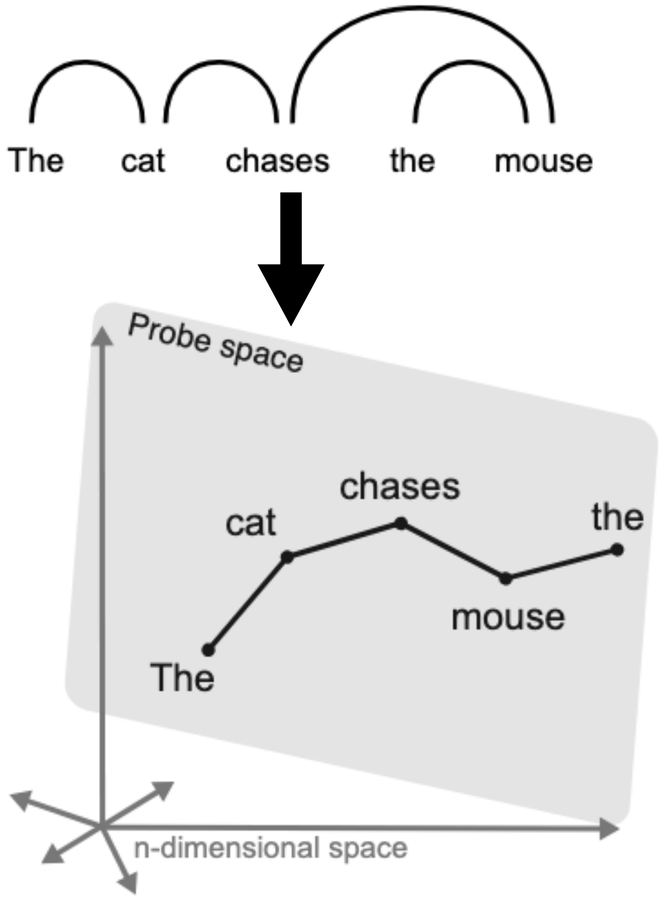
- The structural probe asks whether there exists **a global linear geometry** in which all pairwise dependency distances are represetend at once;



Recap

The structural probe goes beyond local probing classifiers

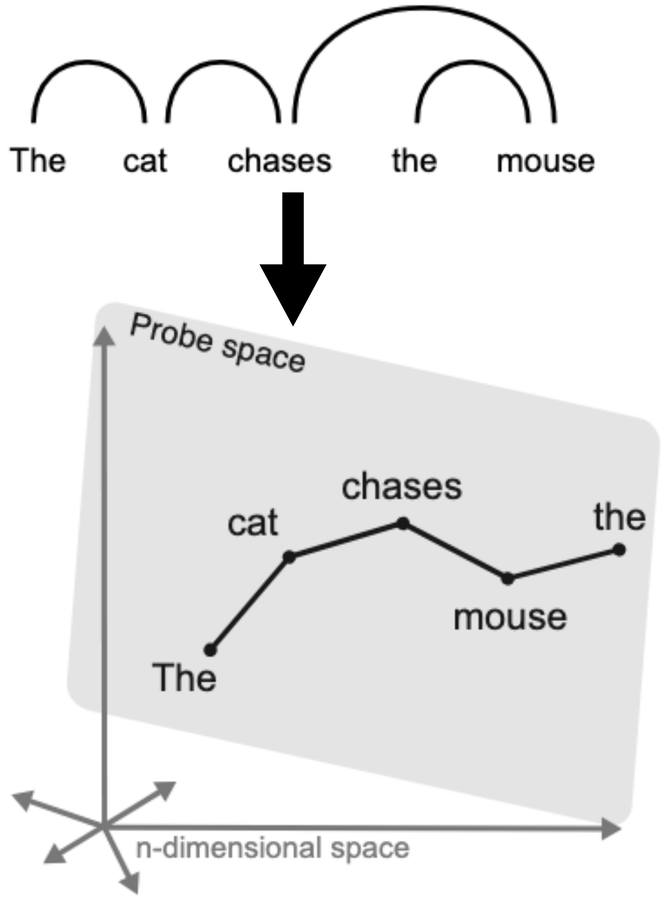
- The structural probe asks whether there exists **a global linear geometry** in which all pairwise dependency distances are represetend at once;
  - This differs from merely probing for the existence of, e.g., parent / head identity from hidden states;



Recap

The structural probe goes beyond local probing classifiers

- The structural probe asks whether there exists **a global linear geometry** in which all pairwise dependency distances are represetend at once;
  - This differs from merely probing for the existence of, e.g., parent / head identity from hidden states;
  - i.e., it is hypothesizing that there is a linear subspace, the *geometric organization* of which represents distance information for all word pairs — **information is not randomly organized in the activation space**!

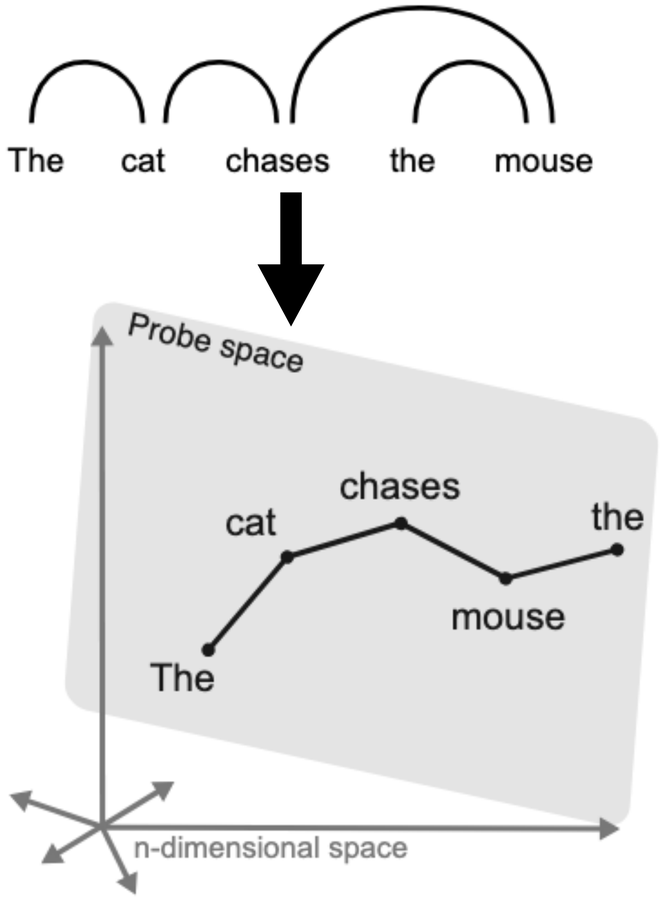


Recap

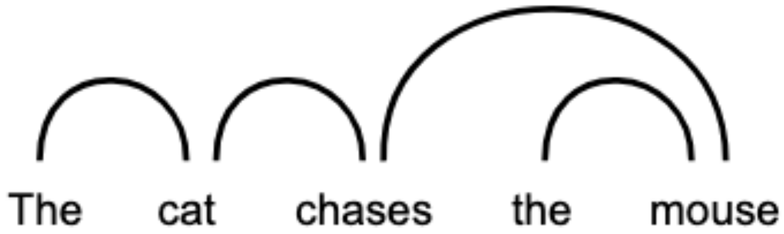
The structural probe goes beyond local probing classifiers

- The structural probe asks whether there exists **a global linear geometry** in which all pairwise dependency distances are represetend at once;
  - This differs from merely probing for the existence of, e.g., parent / head identity from hidden states;
  - i.e., it is hypothesizing that there is a linear subspace, the *geometric organization* of which represents distance information for all word pairs — **information is not randomly organized in the activation space**!

✳ This solves for distance d, but what about **direction** $u : w_i, w_j \rightarrow \{w_i, w_j\}$?

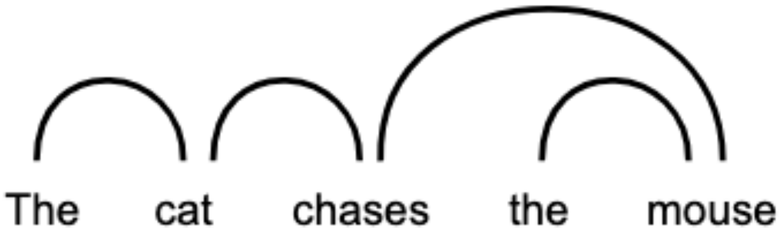


Tree Depth Structural Probe



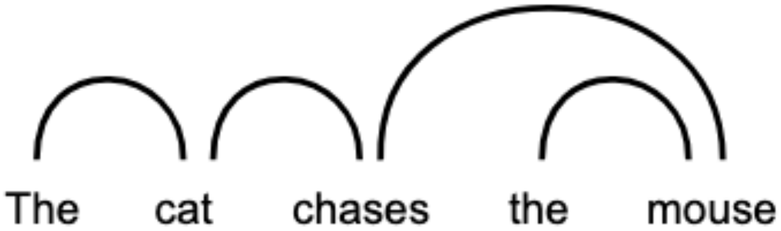
Tree Depth Structural Probe

- In addition to pairwise distances, they also probe for **the parse depth** $\|w_i\|$ **of a word** w_i, defined as the number of edges from w_i to the ROOT.
- $\|w_i\| := \|h_i\|_{\mathbf{B}}^2 = (\mathbf{B}h_i)^\top(\mathbf{B}h_i)$, train for optimizing $\mathbf{B}$ to recreate $\|w_i\|$



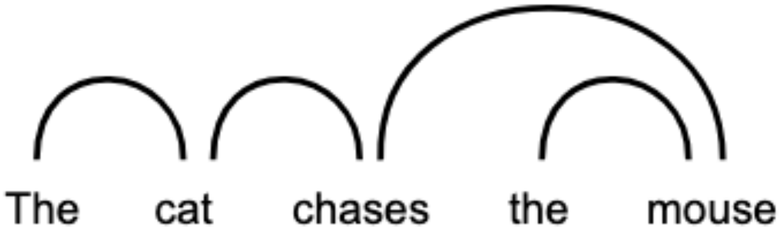
Tree Depth Structural Probe

- In addition to pairwise distances, they also probe for **the parse depth** $\|w_i\|$ **of a word** w_i, defined as the number of edges from w_i to the ROOT.
- $\|w_i\| := \|h_i\|_{\mathbf{B}}^2 = (\mathbf{B}h_i)^\top(\mathbf{B}h_i)$, train for optimizing $\mathbf{B}$ to recreate $\|w_i\|$
- Intuition:
  - Depth is a 1D, relative-to-root quantity;
  - Norm is “how far am I from the origin?”



Tree Depth Structural Probe

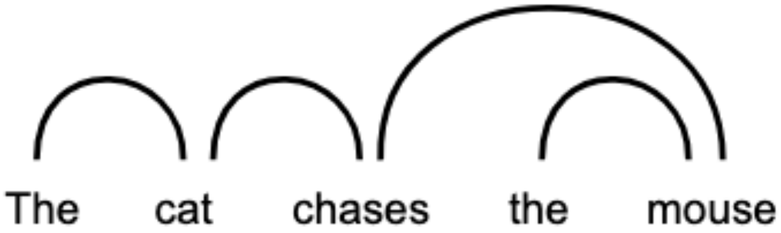
- In addition to pairwise distances, they also probe for **the parse depth** $\|w_i\|$ **of a word** w_i, defined as the number of edges from w_i to the ROOT.
- $\|w_i\| := \|h_i\|_{\mathbf{B}}^2 = (\mathbf{B}h_i)^\top(\mathbf{B}h_i)$, train for optimizing $\mathbf{B}$ to recreate $\|w_i\|$
- Intuition:
  - Depth is a 1D, relative-to-root quantity;
  - Norm is “how far am I from the origin?”



- Hewitt & Manning are *additionally* hypothesizing that:
  - In this syntax subspace, the root sits near the origin — deeper dependents are farther from origin.

Tree Depth Structural Probe

- In addition to pairwise distances, they also probe for **the parse depth** $\|w_i\|$ **of a word** w_i, defined as the number of edges from w_i to the ROOT.
- $\|w_i\| := \|h_i\|_{\mathbf{B}}^2 = (\mathbf{B}h_i)^\top(\mathbf{B}h_i)$, train for optimizing $\mathbf{B}$ to recreate $\|w_i\|$
- Intuition:
  - Depth is a 1D, relative-to-root quantity;
  - Norm is “how far am I from the origin?”
- Hewitt & Manning are *additionally* hypothesizing that:
  - In this syntax subspace, the root sits near the origin — deeper dependents are farther from origin.

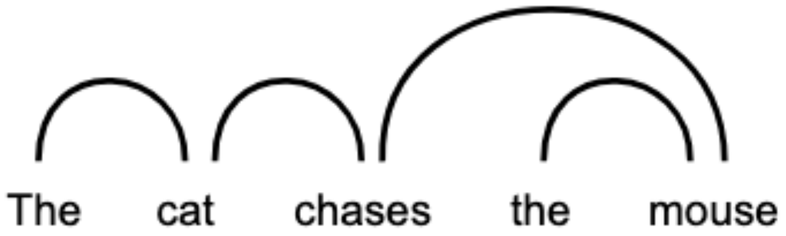


- $\|chases\| = 0$
- $\|cat\| = \|mouse\| = 1$
- $\|The\| = \|the\| = 2$

Depth Probe Identifies Direction

- $\|w_i\| := \|h_i\|_{\mathbf{B}}^2 = (\mathbf{B}h_i)^\top(\mathbf{B}h_i)$
- Use both distance and depth information to determine direction:
  - $d(mouse, the) = 1$
  - $\|\text{mouse}\| < \|\text{the}\|$

➡ Direction: mouse -> the



Experiments

Experiments

Experiments

- Models: `ELMo` ($k = 1024$), `BERT-base` ($k = 768$), `BERT-large` ($k = 1024$);
  - On tokenization: for multi-token words, they took the mean of all activations;

Experiments

- Models: `ELMo` ($k = 1024$), `BERT-base` ($k = 768$), `BERT-large` ($k = 1024$);
  - On tokenization: for multi-token words, they took the mean of all activations;
- To show that the probe doesn't learn from something vaguely useful, consider the following baselines (training $\mathbf{B}$ on different activations):
  - Linear: positional encoding
  - ELMo0: token-level word embedding without contextual information;
  - Decay0: a simple contextualization by linear-distance-weighted averaging of ELMo0 embeddings (decay by $1/2^d$);
  - Proj0: a randomly initialized BiLSTM contextualizer over ELMo0.

Experiments

- Models: `ELMo` ($k = 1024$), `BERT-base` ($k = 768$), `BERT-large` ($k = 1024$);
  - On tokenization: for multi-token words, they took the mean of all activations;
- To show that the probe doesn't learn from something vaguely useful, consider the following baselines (training $\mathbf{B}$ on different activations):
  - Linear: positional encoding
  - ELMo0: token-level word embedding without contextual information;
  - Decay0: a simple contextualization by linear-distance-weighted averaging of ELMo0 embeddings (decay by $1/2^d$);
  - Proj0: a randomly initialized BiLSTM contextualizer over ELMo0.
- Logic: if the probe only performs better on pretrained activations than on baselines, then the activations must contain syntactic geometry.

Results: best layers' performance

Metrics

- **UUAS** (undirected unlabeled attachment score): percentage of undirected edges placed correctly, compared against gold trees.
- **DSpr/NSpr**: Spearman correlation between true and predicted pairwise distances / depth orderings;
- **Root%**: percetage of root correctly identified (i.e., the least deep word);

	Distance		Depth	
Method	UUAS	DSpr.	Root%	NSpr.
LINEAR	48.9	0.58	2.9	0.27
ELMO0	26.8	0.44	54.3	0.56
DECAY0	51.7	0.61	54.3	0.56
PROJ0	59.8	0.73	64.4	0.75
ELMO1	77.0	0.83	86.5	0.87
BERTBASE7	79.8	0.85	88.0	0.87
BERTLARGE15	**82.5**	0.86	89.4	0.88
BERTLARGE16	81.7	**0.87**	**90.1**	**0.89**

BERT-base has 12 layers;
BERT-large has 24 layers.

Results: best layers' performance

Metrics

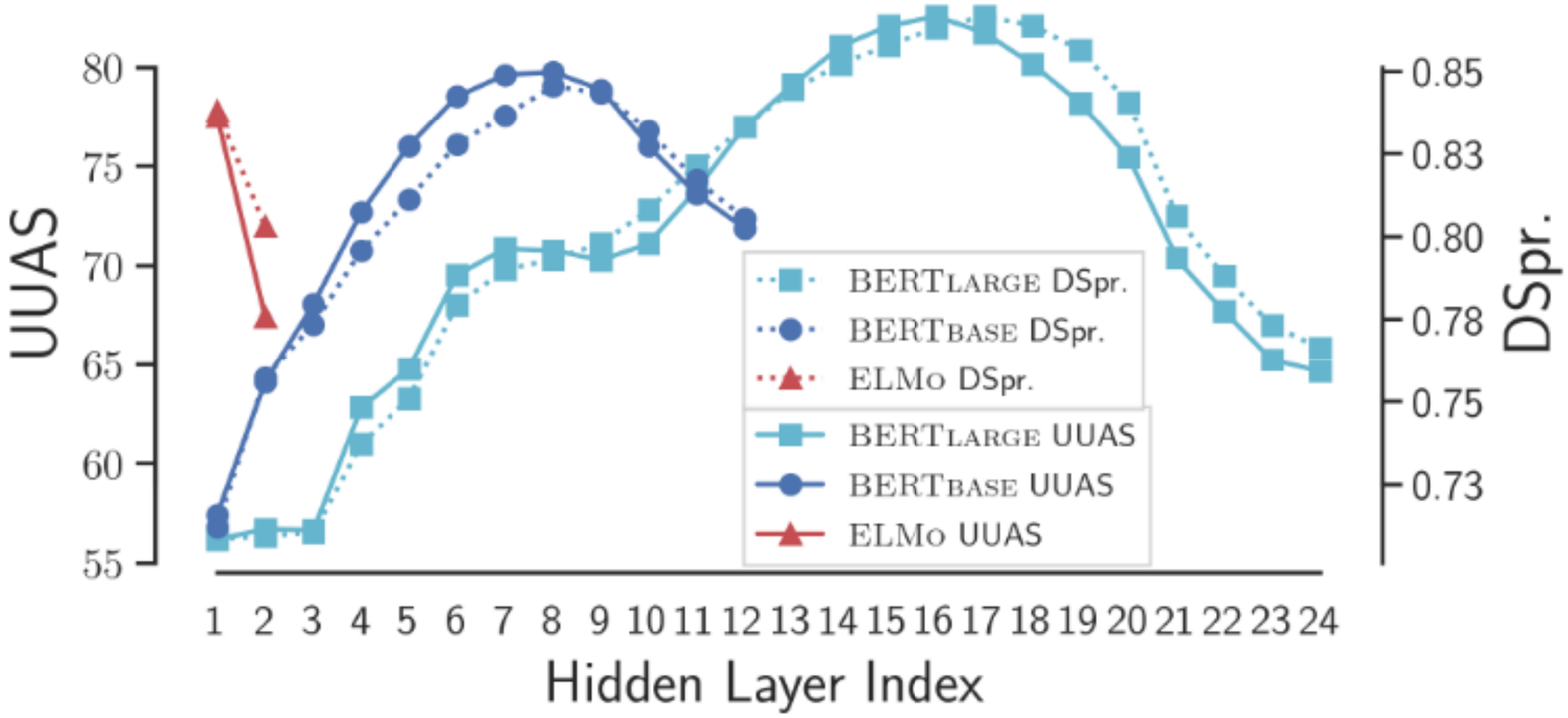
- **UUAS** (undirected unlabeled attachment score): percentage of undirected edges placed correctly, compared against gold trees.
- **DSpr/NSpr**: Spearman correlation between true and predicted pairwise distances / depth orderings;
- **Root%**: percetage of root correctly identified (i.e., the least deep word);

Method	Distance		Depth	
	UUAS	DSpr.	Root%	NSpr.
LINEAR	48.9	0.58	2.9	0.27
ELMO0	26.8	0.44	54.3	0.56
DECAY0	51.7	0.61	54.3	0.56
PROJ0	59.8	0.73	64.4	0.75
ELMO1	77.0	0.83	86.5	0.87
BERTBASE7	79.8	0.85	88.0	0.87
BERTLARGE15	**82.5**	0.86	89.4	0.88
BERTLARGE16	81.7	**0.87**	**90.1**	**0.89**

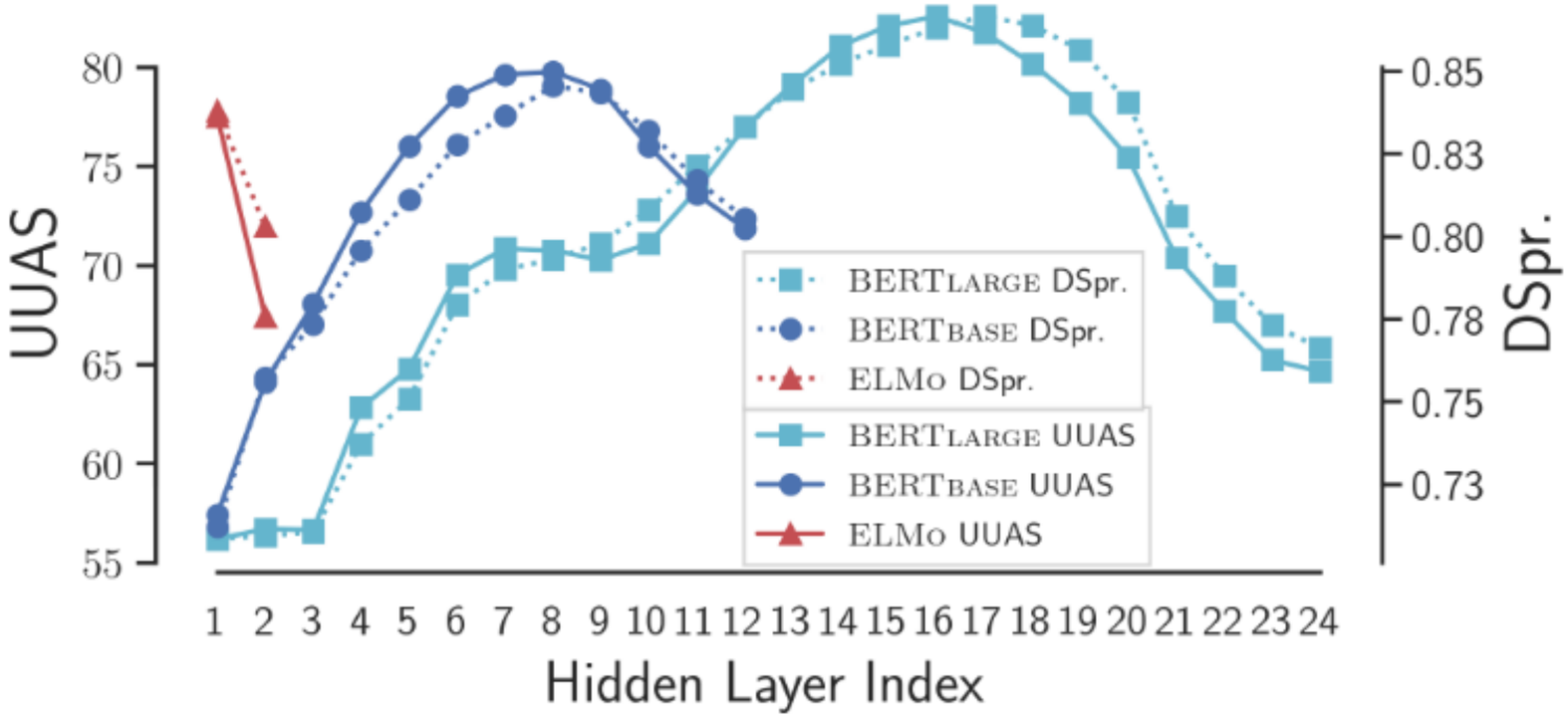
BERT-base has 12 layers;
BERT-large has 24 layers.

Takeaway: non-contextual or weakly contextual baselines do not suffice; pretrained contextual models do encode a lot of dependency structure;

Results: performance across layers

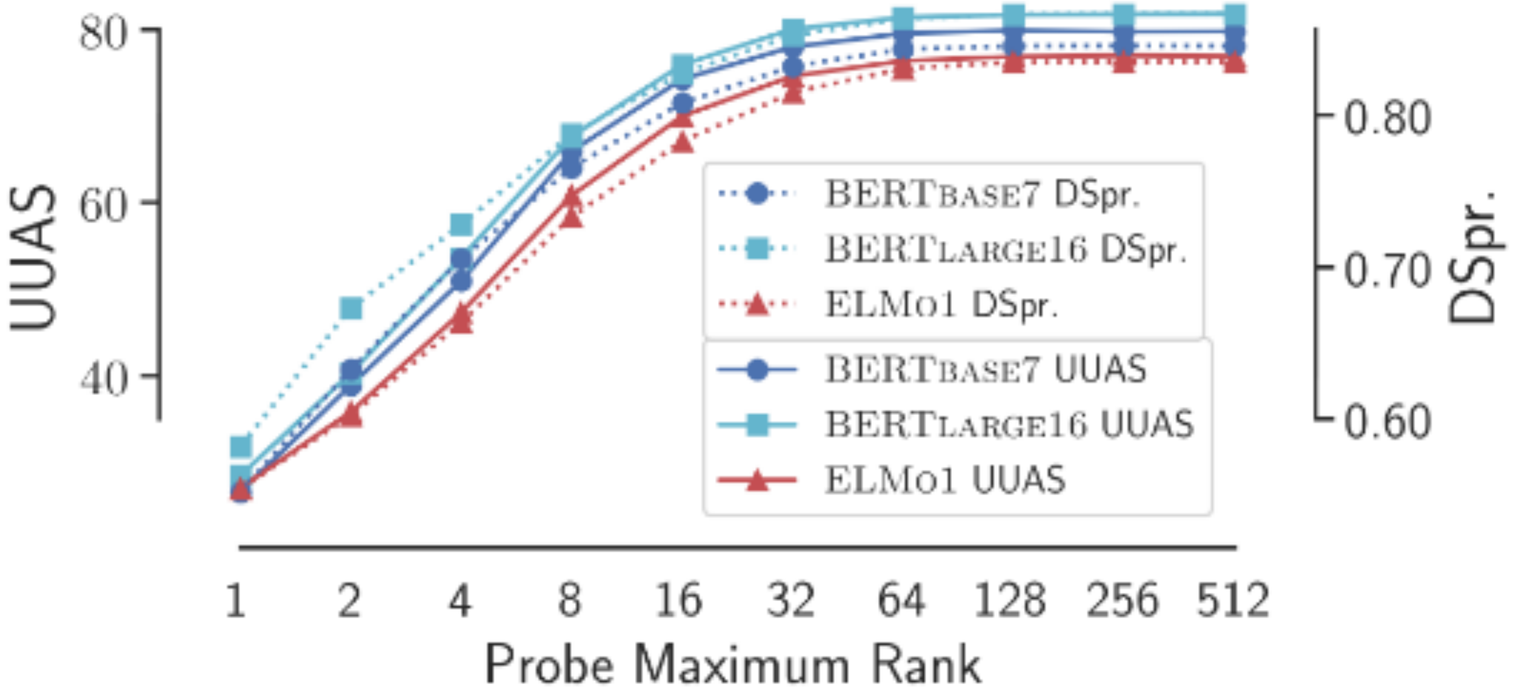


Results: performance across layers

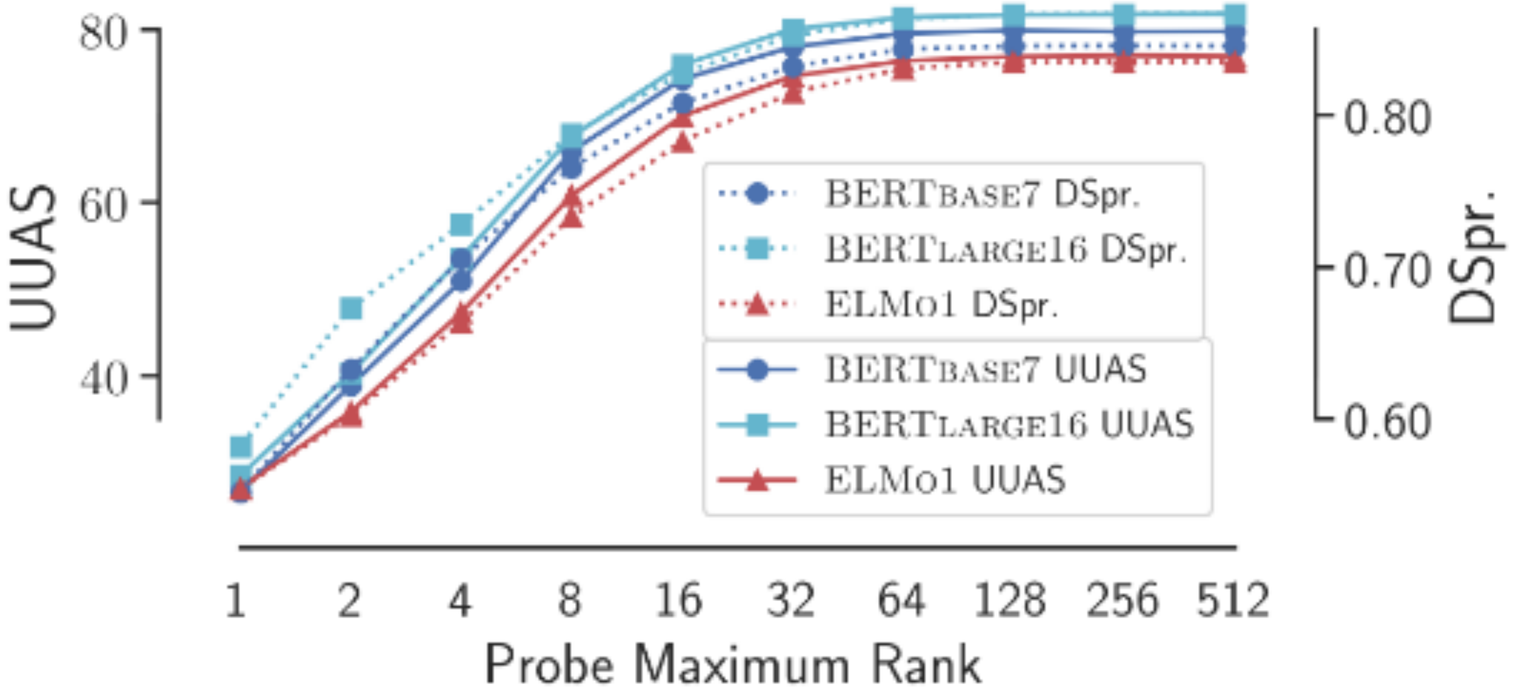


Takeaway: syntax is not uniform across layers — middle-to-upper layers outperform the others

On the dimensionality of the syntax subspace



On the dimensionality of the syntax subspace

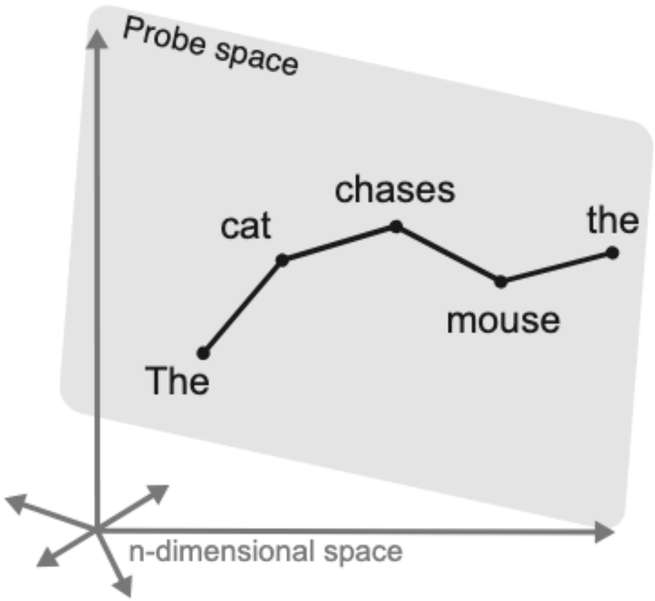


Takeaway: increasing the probe dimension beyond 64 or 128 gives little gain — syntax lives in a low-rank subspace!

Summary: Two Quantities

We need two quantities to reconstruct unlabeled, directed structures:

- **Distance**: hierarchical distance between pairs of words;
- **Norm**: a word's depth / distance from the origin / ROOT;



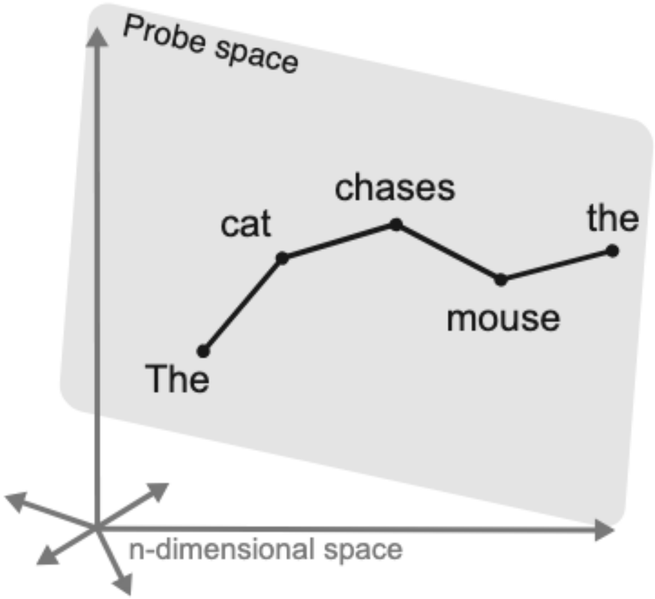
Summary: Two Quantities

We need two quantities to reconstruct unlabeled, directed structures:

- **Distance**: hierarchical distance between pairs of words;
- **Norm**: a word's depth / distance from the origin / ROOT;

Issues?

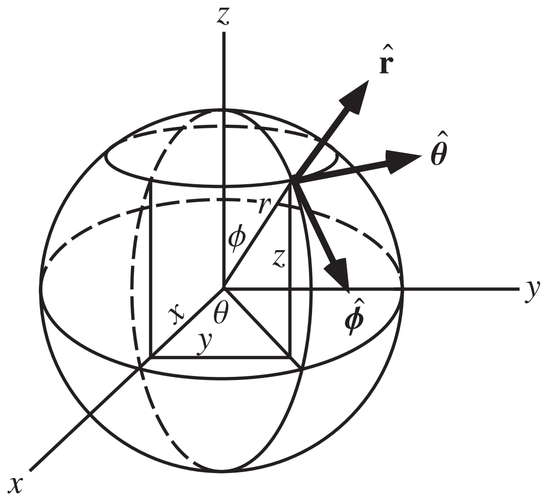
- No dependency type information:
  - cannot distinguish "The cat chases the mouse" vs. "The mouse chases the cat".
- Need both information to derive direction.



Diego-Simón et al. 2024

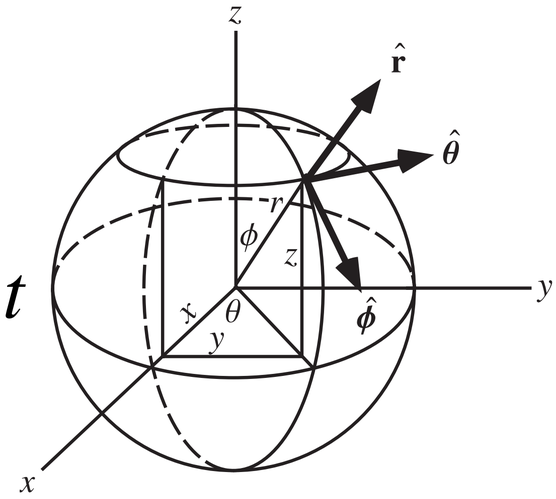
Polar Probe

Angular Probe: Core Idea

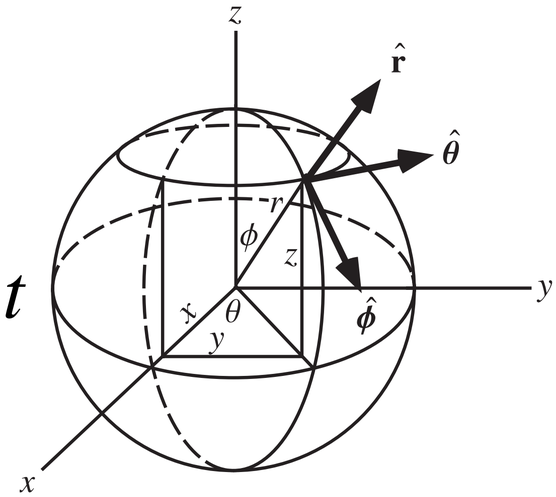


Angular Probe: Core Idea

- **Goal** = in addition to distance, also probe for dependency type t ($t : w_i, w_j \rightarrow C$) and head/direction u ($u : w_i, w_j \rightarrow \{w_i, w_j\}$);

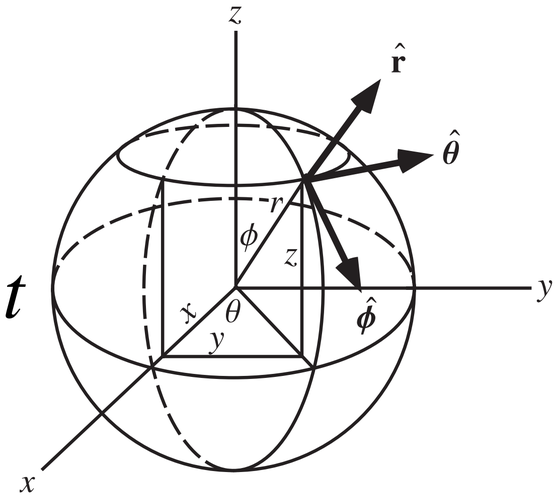


Angular Probe: Core Idea



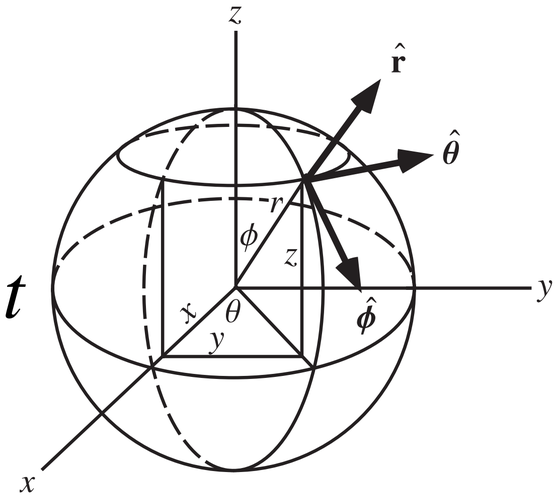
- **Goal** = in addition to distance, also probe for dependency type t ($t : w_i, w_j \rightarrow C$) and head/direction u ($u : w_i, w_j \rightarrow \{w_i, w_j\}$);
- Now consider only the edge set Ω_A: the set of all pairs of words with edges;

Angular Probe: Core Idea



- **Goal** = in addition to distance, also probe for dependency type t ($t : w_i, w_j \rightarrow C$) and head/direction u ($u : w_i, w_j \rightarrow \{w_i, w_j\}$);
- Now consider only the edge set Ω_A: the set of all pairs of words with edges;
- For each pair, let the difference in activation $s_{i,j} = h_i - h_j$ represent the edge;

Angular Probe: Core Idea



- **Goal** = in addition to distance, also probe for dependency type t ($t : w_i, w_j \rightarrow C$) and head/direction u ($u : w_i, w_j \rightarrow \{w_i, w_j\}$);
- Now consider only the edge set Ω_A: the set of all pairs of words with edges;
- For each pair, let the difference in activation $s_{i,j} = h_i - h_j$ represent the edge;
- **Intuition**:
  - If two eges have the same dependency type, they should be aligned (colinear) in the transformed space; different type → orthogonal;
  - Given a reference direction for a type, if current edge vector points in the same direction (cos similarity > 0), then w_i is the head; opposite direction → w_j is head.

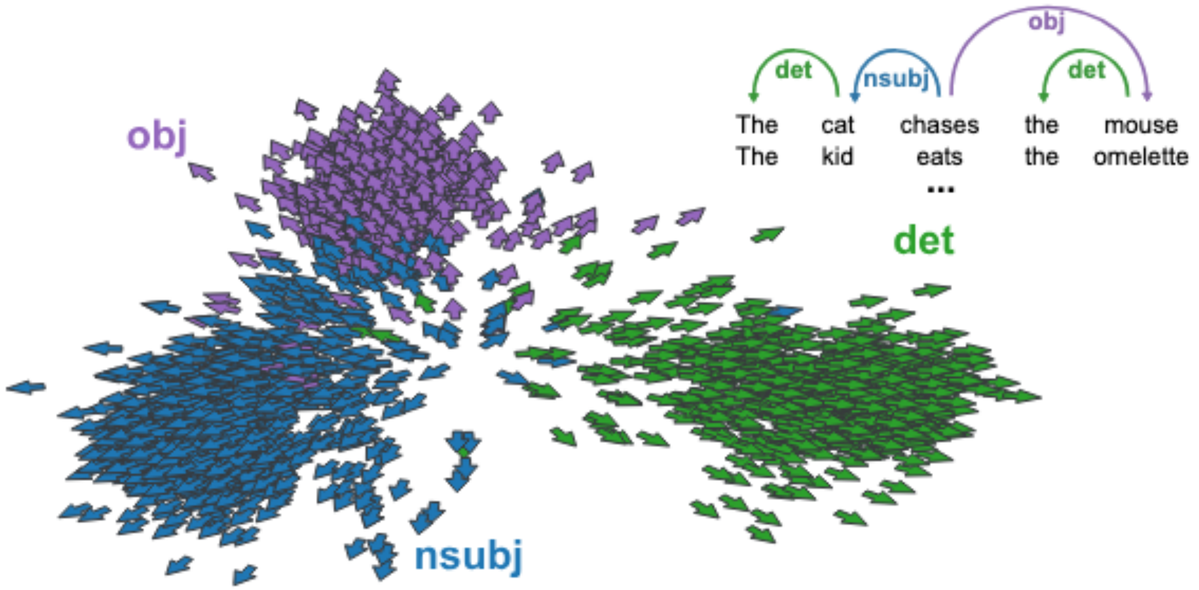
Angular Probe: Formalization

$$\mathcal{L}_A(\mathbf{B}_A) = \frac{1}{\Omega_A} \sum_{s,s' \in \Omega_A} \left(\measuredangle(\mathbf{B}_A\mathbf{s}, \mathbf{B}_A\mathbf{s}') - \mathbb{1}\left[t(\mathbf{s}) = t(\mathbf{s}')\right]\right)^2,$$

- Ω_A is the set of edge embeddings of syntactically connected words,
- $\measuredangle : \mathbf{x}, \mathbf{y} \mapsto \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|}$ is the cosine similarity,
- $\mathbb{1} : \mathcal{X} \mapsto \begin{cases} 1 \text{ if } \mathcal{X} \text{ is true} \\ 0 \text{ otherwise} \end{cases}$ is the indicator function.

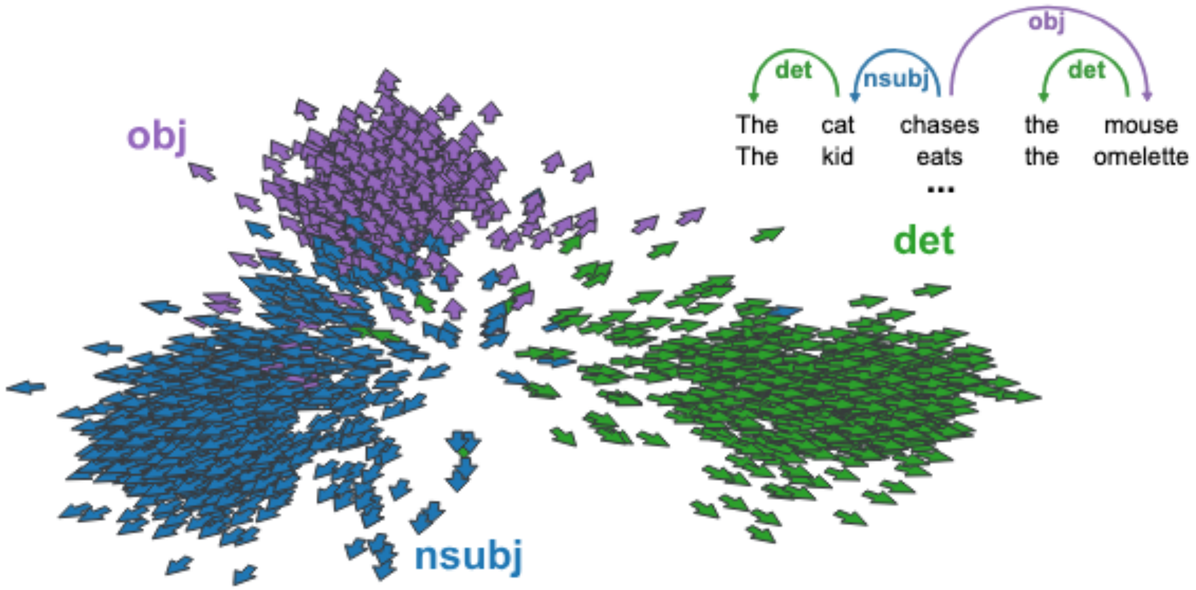
- $\mathbf{B}_A$ for angular probe; now comparing two edge embeddings s, s';
- **Intuition = contrastive learning**: find the linear transformation such that edges with the same type should align in the same direction, while edges with different types should be orthogonal;

Angular Probe: Decoding Type and Head



Angular Probe: Decoding Type and Head

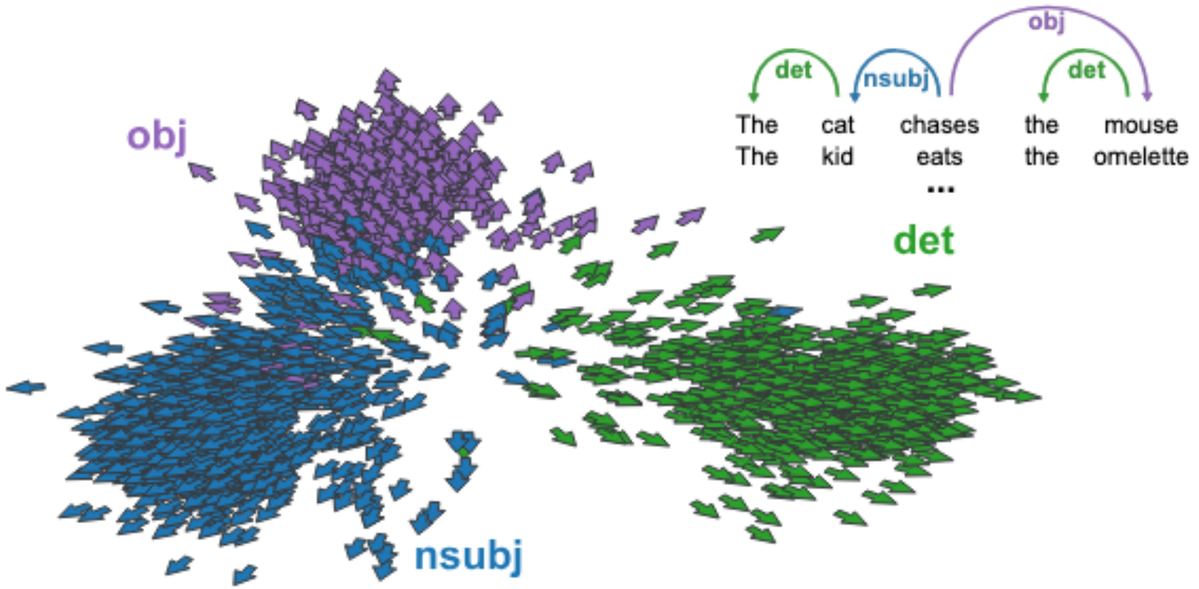
- Once learning is done and we obtain $\mathbf{B}_A$:



Angular Probe: Decoding Type and Head

- Once learning is done and we obtain $\mathbf{B}_A$:
- For each dependency type c, construct a prototypical vector by averaging all edges of this type:

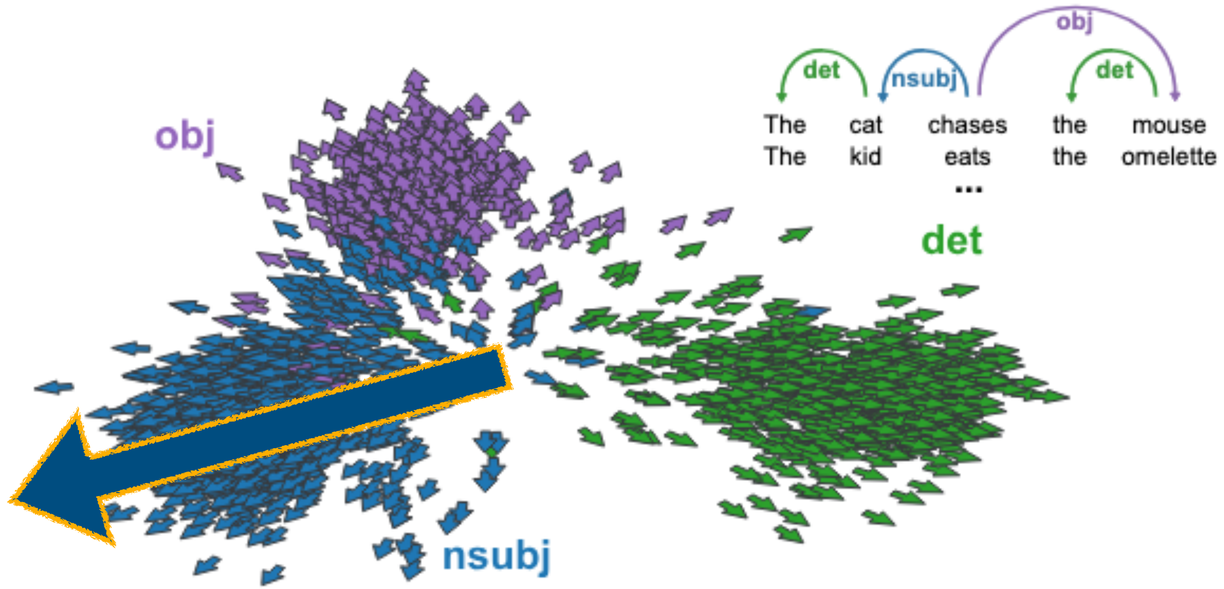
$$\mathbf{V}_c = \sum_{s \in \Omega_A^{(c)}} \mathbf{B}_A s, \text{ where } \Omega_A^{(c)} = \{ s \in \Omega_A \mid t(s) = c \}$$



Angular Probe: Decoding Type and Head

- Once learning is done and we obtain $\mathbf{B}_A$:
- For each dependency type c, construct a prototypical vector by averaging all edges of this type:

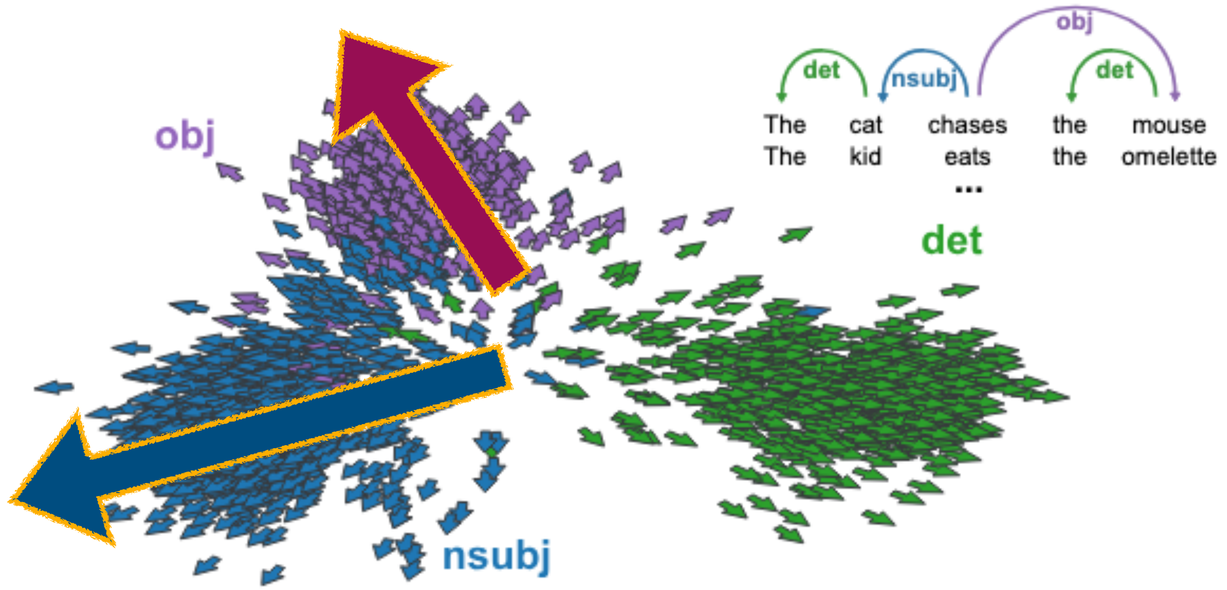
$$\mathbf{V}_c = \sum_{s \in \Omega_A^{(c)}} \mathbf{B}_A s, \text{ where } \Omega_A^{(c)} = \{ s \in \Omega_A \mid t(s) = c \}$$



Angular Probe: Decoding Type and Head

- Once learning is done and we obtain $\mathbf{B}_A$:
- For each dependency type c, construct a prototypical vector by averaging all edges of this type:

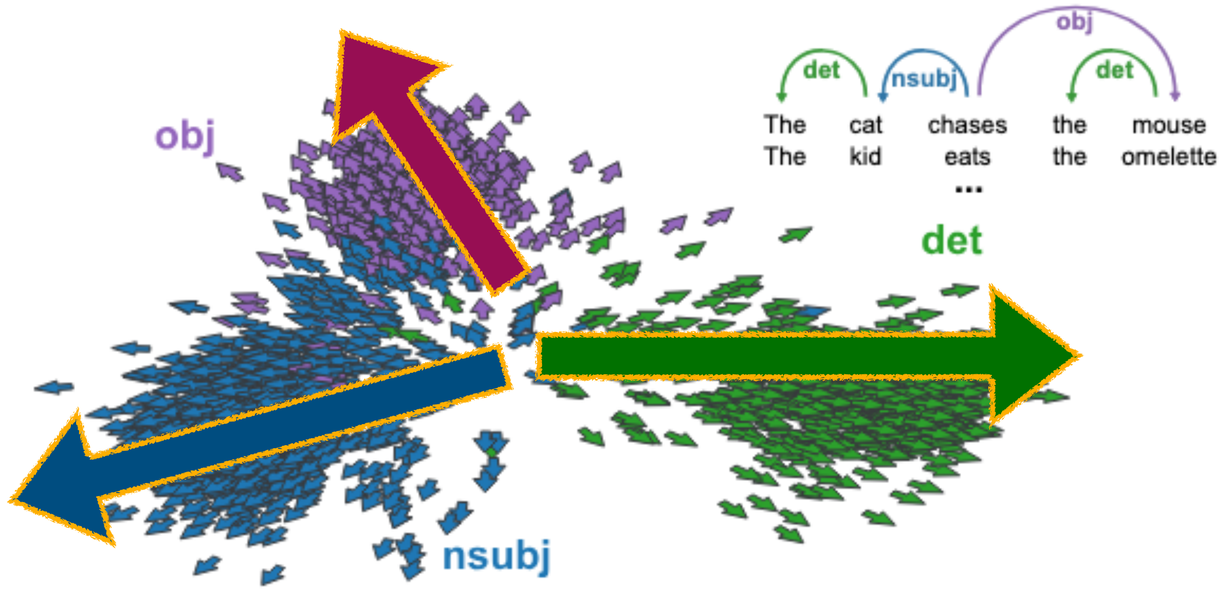
$$\mathbf{V}_c = \sum_{s \in \Omega_A^{(c)}} \mathbf{B}_A s, \text{ where } \Omega_A^{(c)} = \{ s \in \Omega_A \mid t(s) = c \}$$



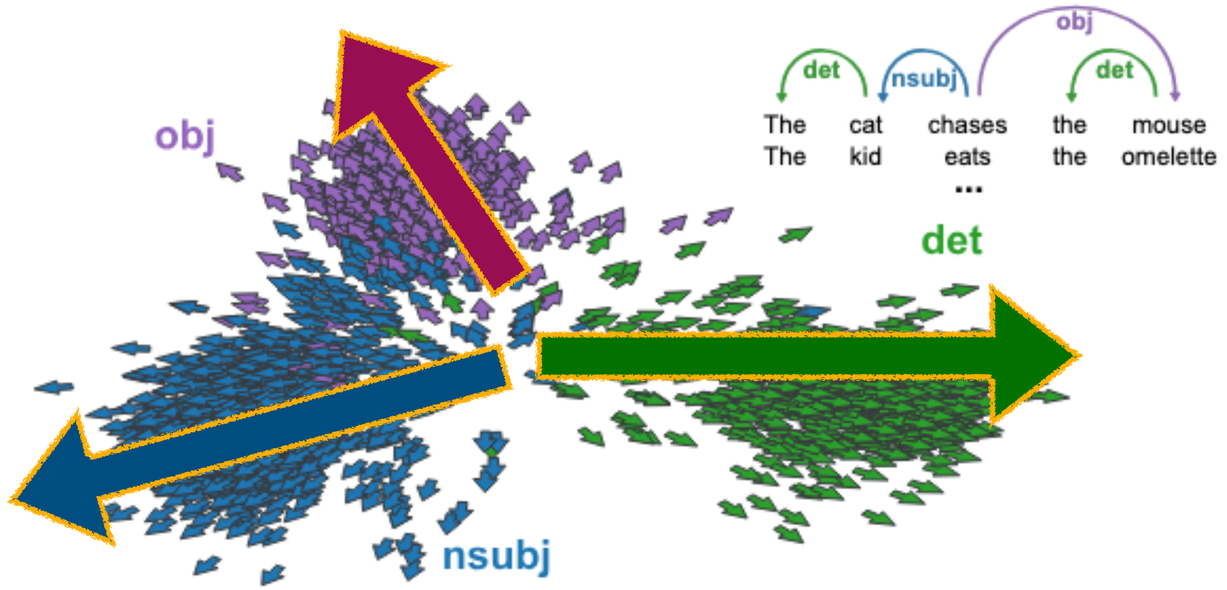
Angular Probe: Decoding Type and Head

- Once learning is done and we obtain $\mathbf{B}_A$:
- For each dependency type c, construct a prototypical vector by averaging all edges of this type:

$$\mathbf{V}_c = \sum_{s \in \Omega_A^{(c)}} \mathbf{B}_A s, \text{ where } \Omega_A^{(c)} = \{ s \in \Omega_A \mid t(s) = c \}$$

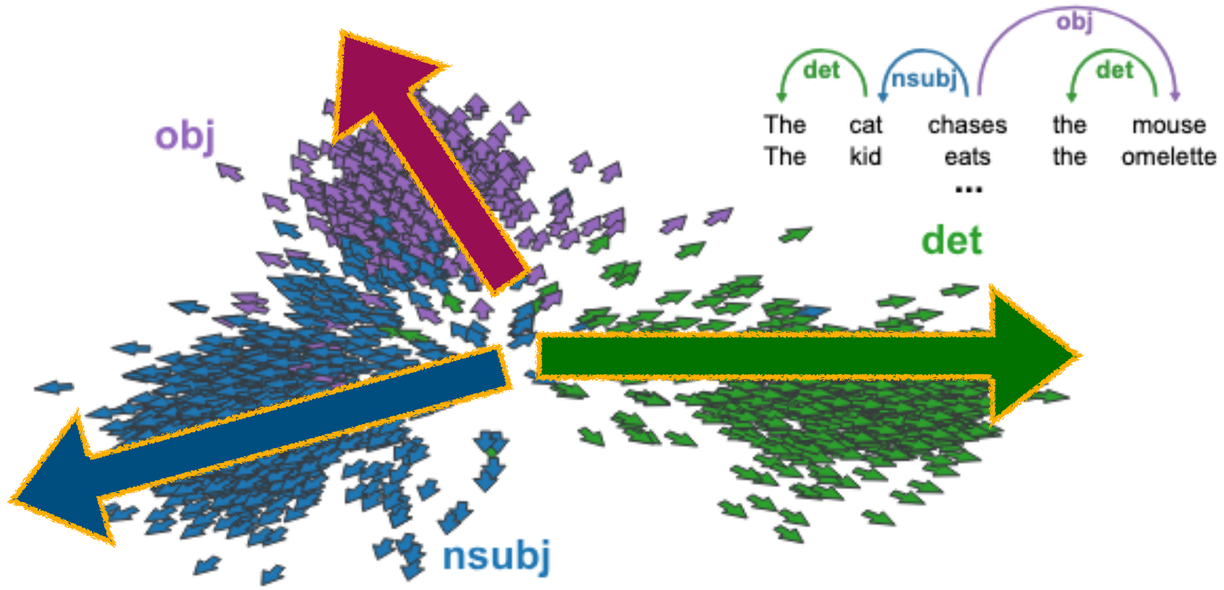


Angular Probe: Decoding Type and Head



Angular Probe: Decoding Type and Head

- Then, for arbitrary edge, $\hat{t}(h_i, h_j) = \underset{c}{\text{argmax}} \left| cos\_sim(\mathbf{B}_A s_{ij}, \mathbf{V}_c) \right|$

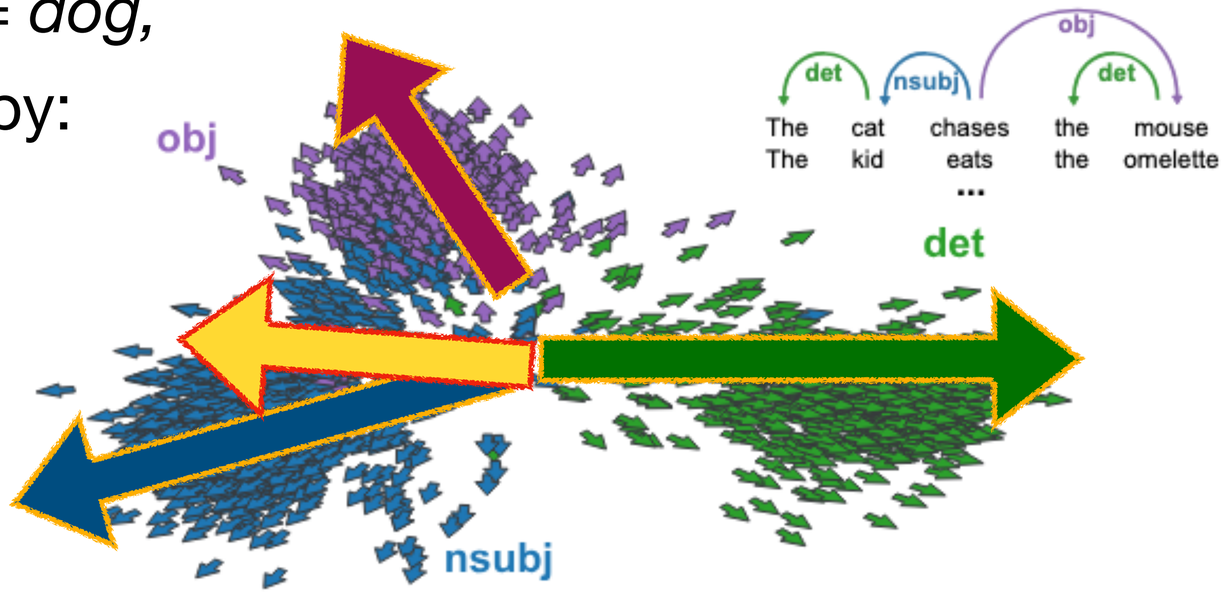


Angular Probe: Decoding Type and Head

- Then, for arbitrary edge, $\hat{t}(h_i, h_j) = \underset{c}{\text{argmax}} \, |\, cos\_sim(\mathbf{B}_A s_{ij}, \mathbf{V}_c)\,|$

Suppose $w_i = A$, $w_j = dog$,

$\mathbf{B}_A s_{ij}$ is represented by:

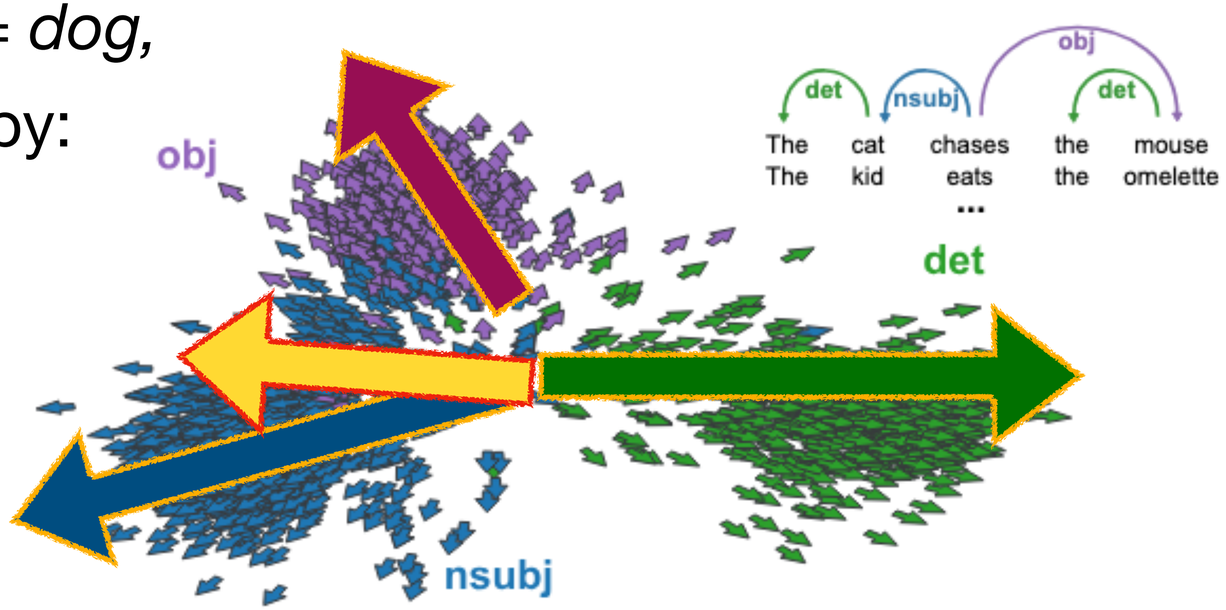


Angular Probe: Decoding Type and Head

- Then, for arbitrary edge, $\hat{t}(h_i, h_j) = \underset{c}{\text{argmax}} \, |\, cos\_sim(\mathbf{B}_A s_{ij}, \mathbf{V}_c)\,|$

Suppose $w_i = A$, $w_j = dog$,

$\mathbf{B}_A s_{ij}$ is represented by:



Then, $\hat{t}(h_i, h_j)$ = **det**

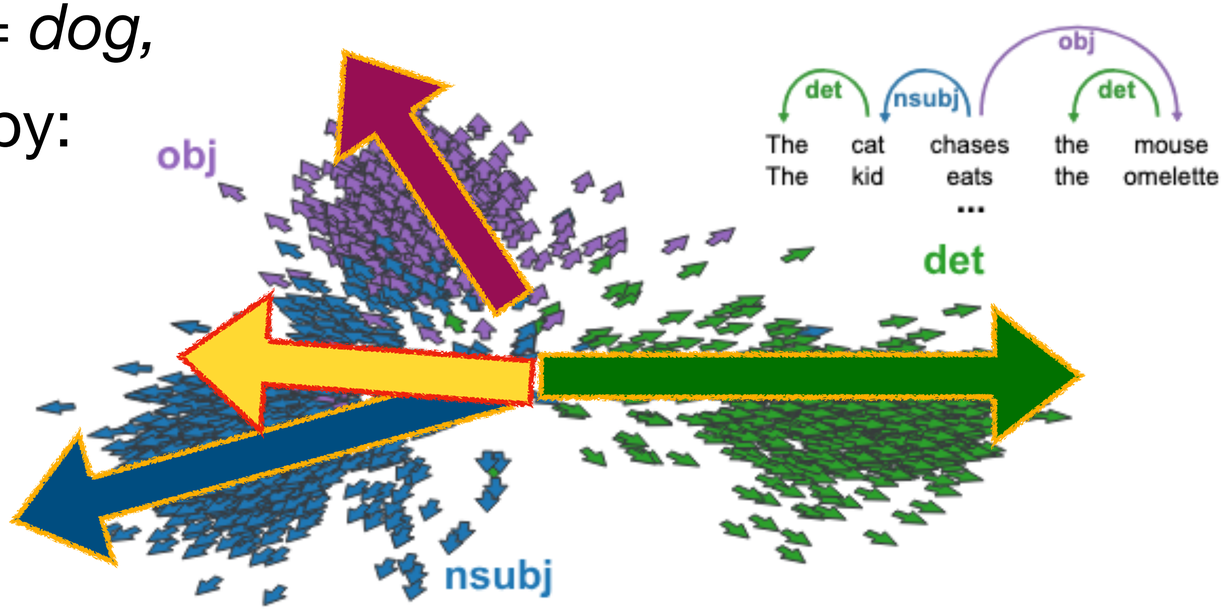
Angular Probe: Decoding Type and Head

- Then, for arbitrary edge, $\hat{t}(h_i, h_j) = \underset{c}{\text{argmax}} \, |\, cos\_sim(\mathbf{B}_A s_{ij}, \mathbf{V}_c)\,|$
- Predict head word (i.e. direction) given predicted type $\hat{t}$ with:

$$\hat{u}(\mathbf{h}_i, \mathbf{h}_j) := \begin{cases} \mathbf{h}_i & \text{if } \measuredangle(\mathbf{B}_A \mathbf{s}_{ij}, \mathbf{V}_{\hat{t}(\mathbf{h}_i, \mathbf{h}_j)}) \geq 0 \\ \mathbf{h}_j & \text{if } \measuredangle(\mathbf{B}_A \mathbf{s}_{ij}, \mathbf{V}_{\hat{t}(\mathbf{h}_i, \mathbf{h}_j)}) < 0 \end{cases}$$

Suppose w_i = *A*, w_j = *dog*,

$\mathbf{B}_A s_{ij}$ is represented by:



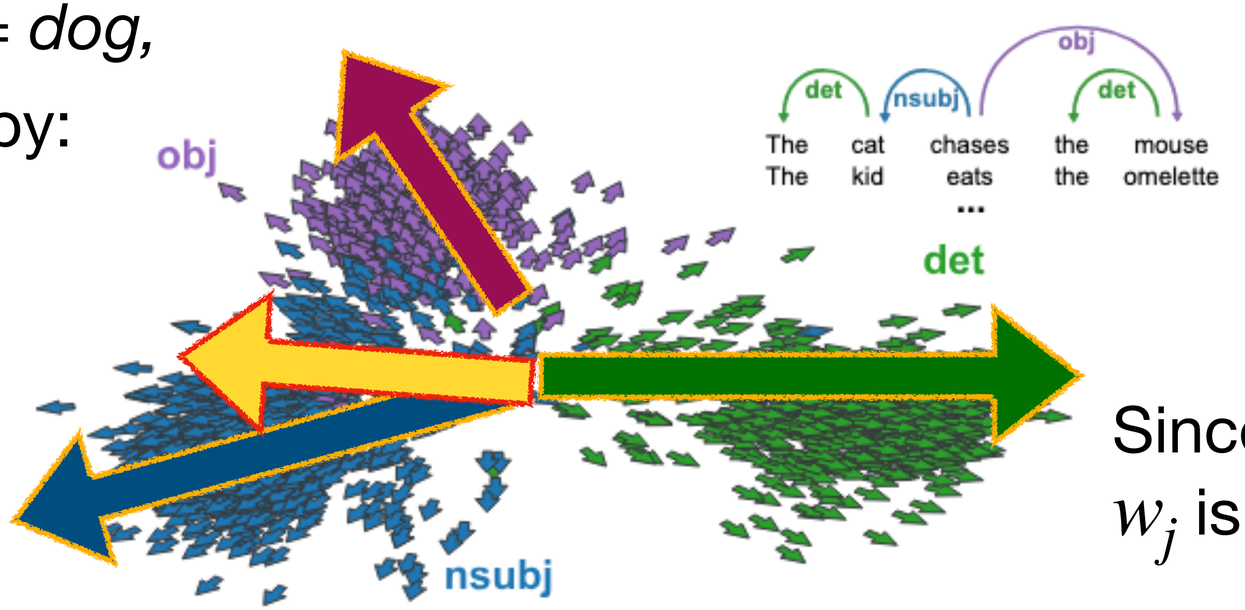
Then, $\hat{t}(h_i, h_j)$ = **`det`**

Angular Probe: Decoding Type and Head

- Then, for arbitrary edge, $\hat{t}(h_i, h_j) = \underset{c}{\text{argmax}} \, |cos\_sim(\mathbf{B}_A s_{ij}, \mathbf{V}_c)|$
- Predict head word (i.e. direction) given predicted type $\hat{t}$ with:

$$\hat{u}(\mathbf{h}_i, \mathbf{h}_j) := \begin{cases} \mathbf{h}_i & \text{if } \measuredangle(\mathbf{B}_A \mathbf{s}_{ij}, \mathbf{V}_{\hat{t}(\mathbf{h}_i, \mathbf{h}_j)}) \geq 0 \\ \mathbf{h}_j & \text{if } \measuredangle(\mathbf{B}_A \mathbf{s}_{ij}, \mathbf{V}_{\hat{t}(\mathbf{h}_i, \mathbf{h}_j)}) < 0 \end{cases}$$

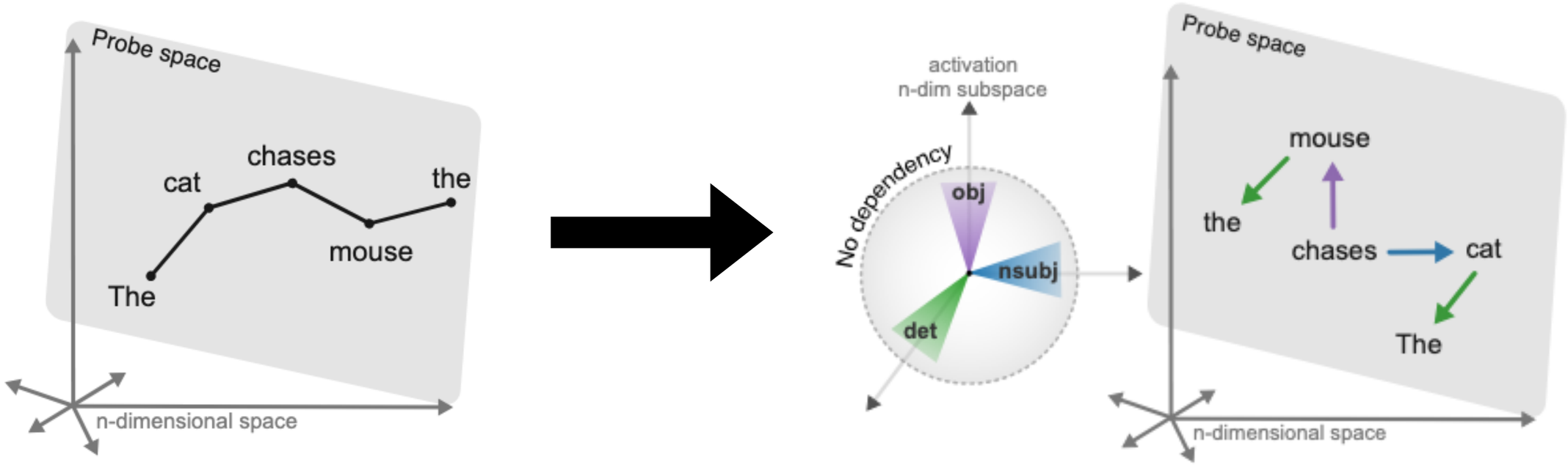
Suppose w_i = *A*, w_j = *dog*,
$\mathbf{B}_A s_{ij}$ is represented by:



Then, $\hat{t}(h_i, h_j)$ = **`det`**

Since cosine similarity < 0,
w_j is the head!

Angular Probe: Illustration



Polar Probe: Structural Probe + Angular Probe

Polar Probe: Structural Probe + Angular Probe

- Finally, polar probe is simply combining structural probe and angular probe.
  - Intuition: in polar coordinate, we need two quantities to determine a point — an angle and a distance.

Polar Probe: Structural Probe + Angular Probe

- Finally, polar probe is simply combining structural probe and angular probe.
  - Intuition: in polar coordinate, we need two quantities to determine a point — an angle and a distance.
- Remember $\mathscr{L}_S$ stands for structural loss and $\mathscr{L}_A$ stands for angular loss;

Polar Probe: Structural Probe + Angular Probe

- Finally, polar probe is simply combining structural probe and angular probe.
  - Intuition: in polar coordinate, we need two quantities to determine a point — an angle and a distance.
- Remember $\mathscr{L}_S$ stands for structural loss and $\mathscr{L}_A$ stands for angular loss;
- Here is the final solution, this time with a single $\mathbf{B}_P$, minimizing the polar loss:

$$\underset{\mathbf{B}_P}{\operatorname{argmax}}\, \mathscr{L}_S(\mathbf{B}_P) + \lambda \mathscr{L}_A(\mathbf{B}_P)$$

$$\hat{d}(\mathbf{h}_i, \mathbf{h}_j) := \|\mathbf{B}_P \mathbf{s}_{ij}\|^2$$

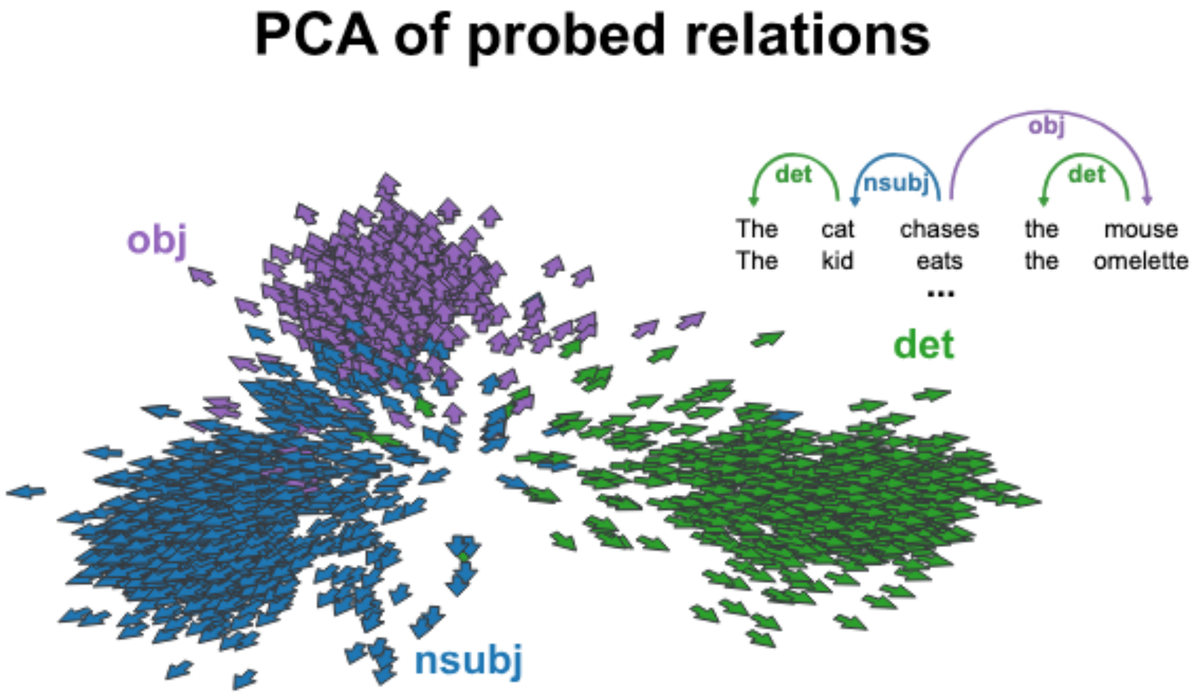
$$\hat{t}(\mathbf{h}_i, \mathbf{h}_j) := \underset{c}{\operatorname{argmax}} |\measuredangle(\mathbf{B}_P \mathbf{s}_{ij}, \mathbf{V}_c)|$$

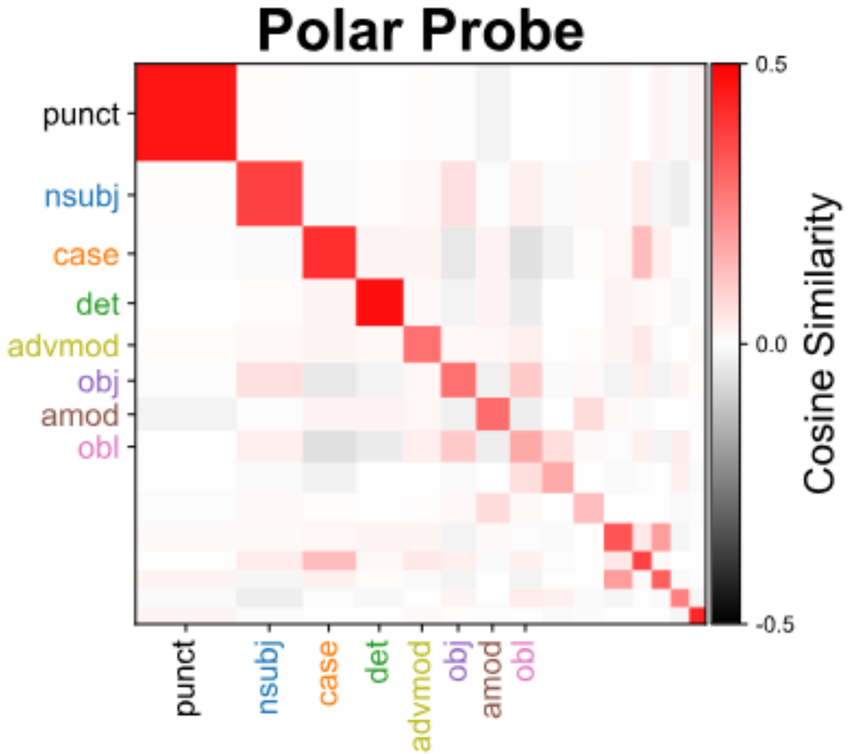
$$\hat{u}(\mathbf{h}_i, \mathbf{h}_j) := \begin{cases} \mathbf{h}_i & \text{if } \measuredangle(\mathbf{B}_P \mathbf{s}_{ij}, \mathbf{V}_{\hat{t}(\mathbf{h}_i, \mathbf{h}_j)}) \geq 0 \\ \mathbf{h}_j & \text{if } \measuredangle(\mathbf{B}_P \mathbf{s}_{ij}, \mathbf{V}_{\hat{t}(\mathbf{h}_i, \mathbf{h}_j)}) < 0 \end{cases}$$

Experiments

Result: can polar probe recover types?

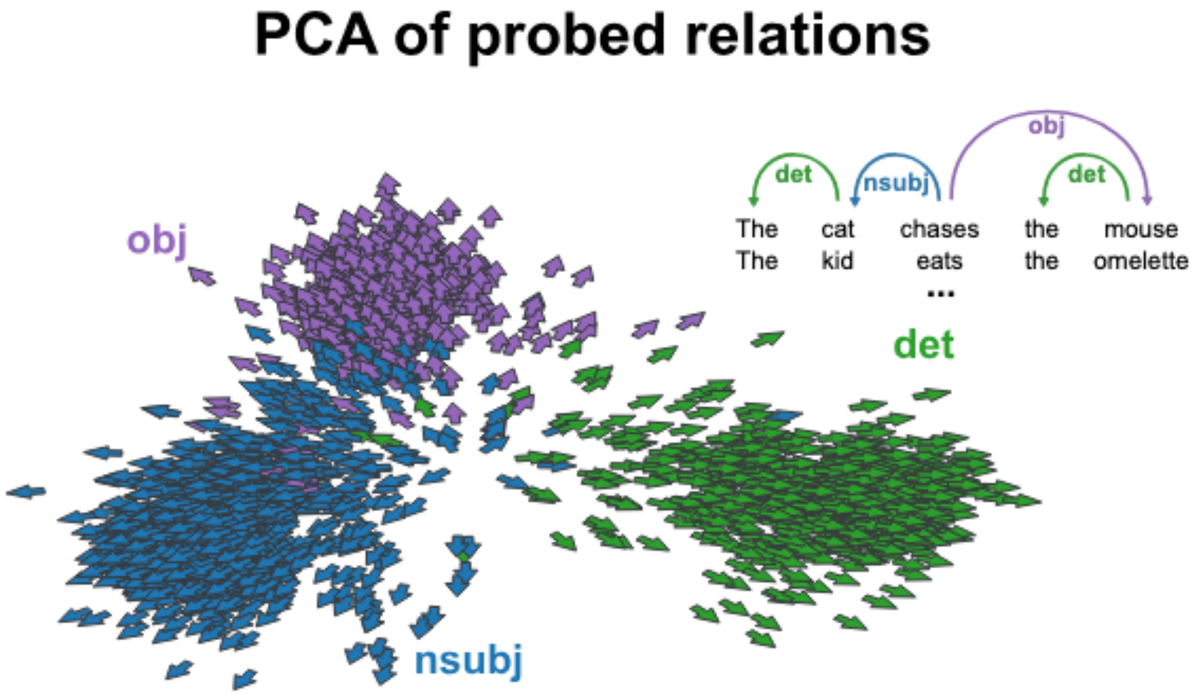
- **Models**: Mistral-7B ($k = 4096$), Llama-2-7b ($k = 4096$); BERT-large ($k = 1024$);

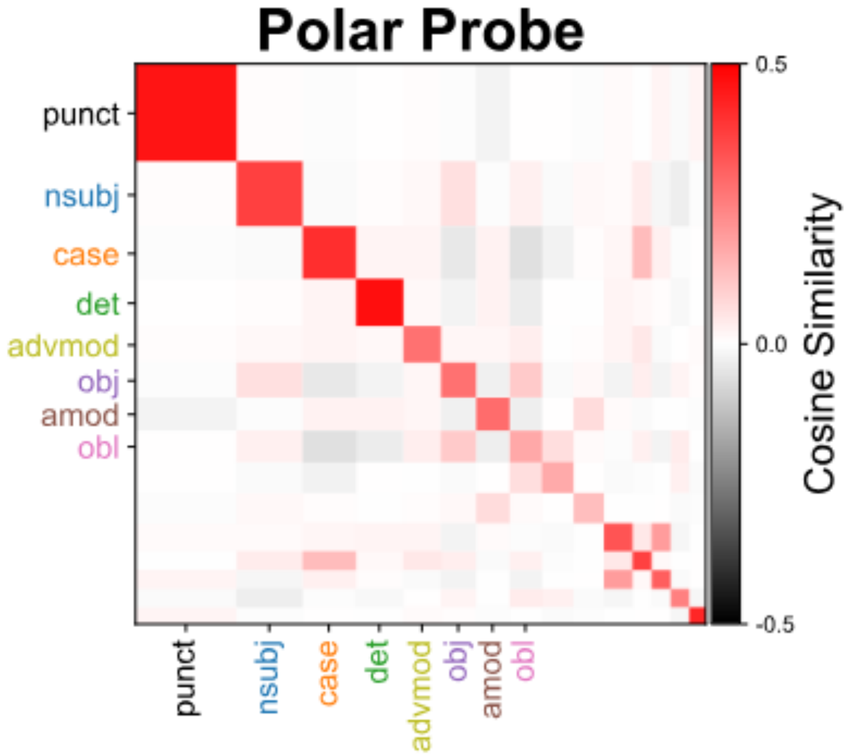




Result: can polar probe recover types?

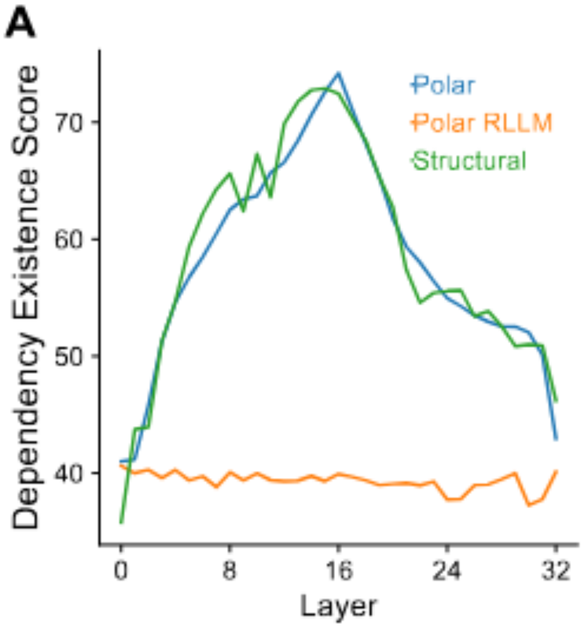
- **Models**: Mistral-7B ($k = 4096$), Llama-2-7b ($k = 4096$); BERT-large ($k = 1024$);



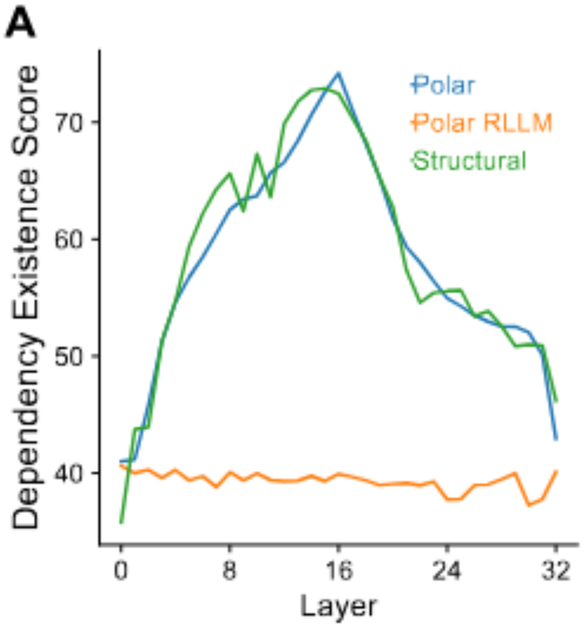


Takeaway: dependency type information is encoded geometrically as direction in the transformed space, not just recoverable by the probe.

Result: how Angular and Structural interact?



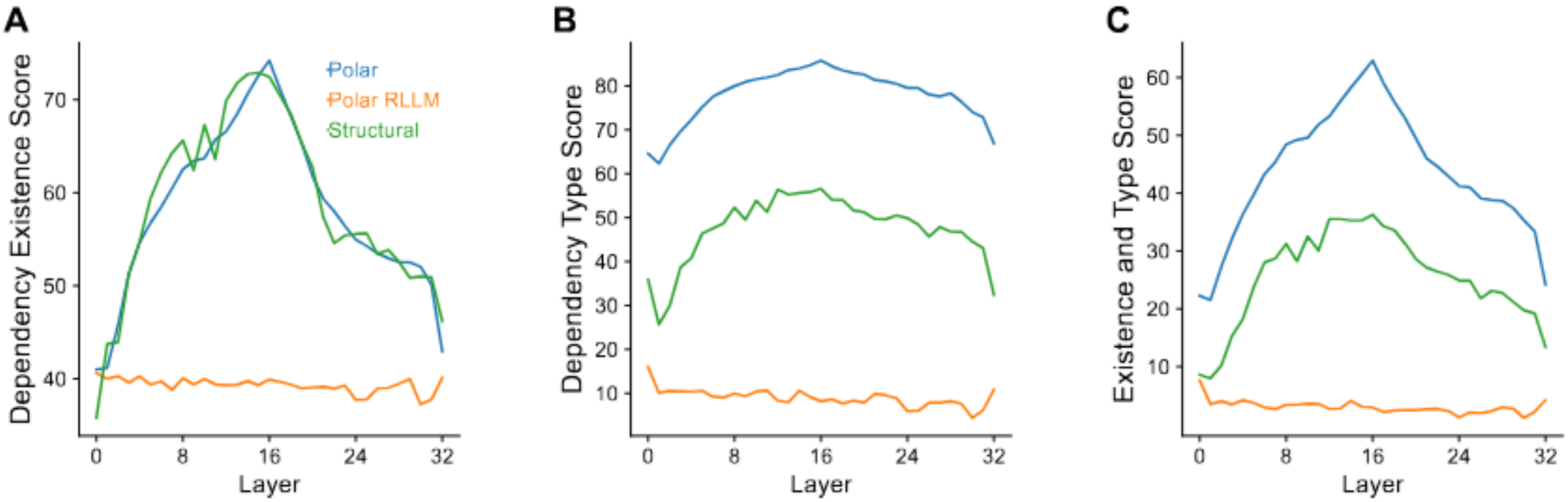
Result: how Angular and Structural interact?



Takeaway:

- the best layer is at layer 16 (middle), aligning with structural result.

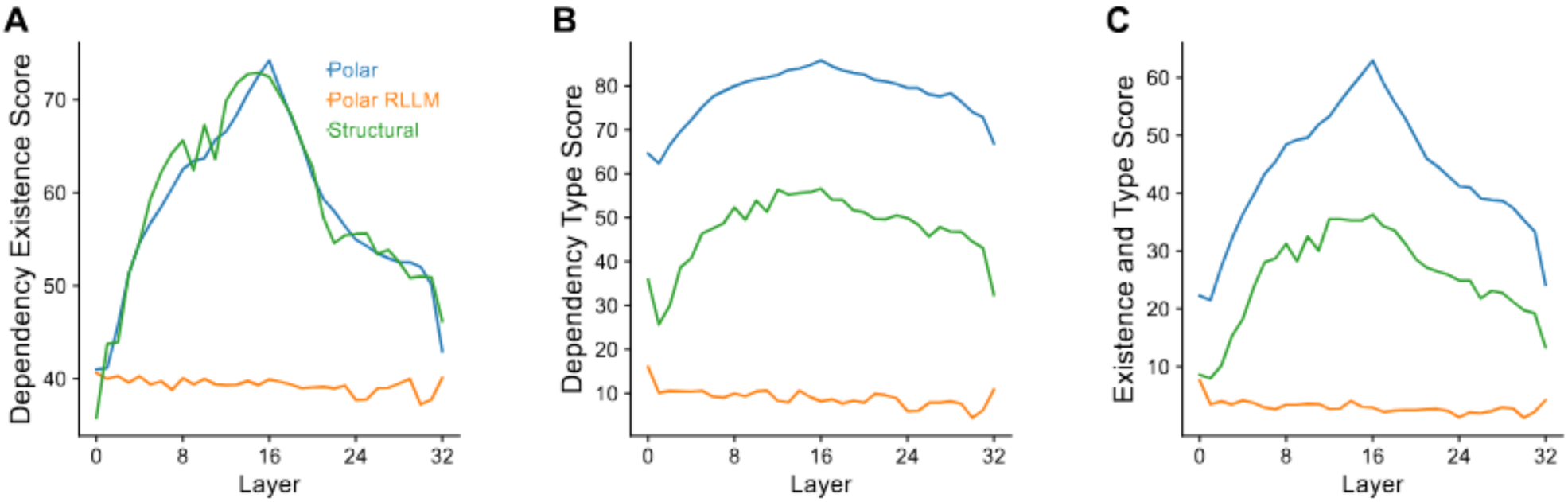
Result: how Angular and Structural interact?



Takeaway:

- the best layer is at layer 16 (middle), aligning with structural result.

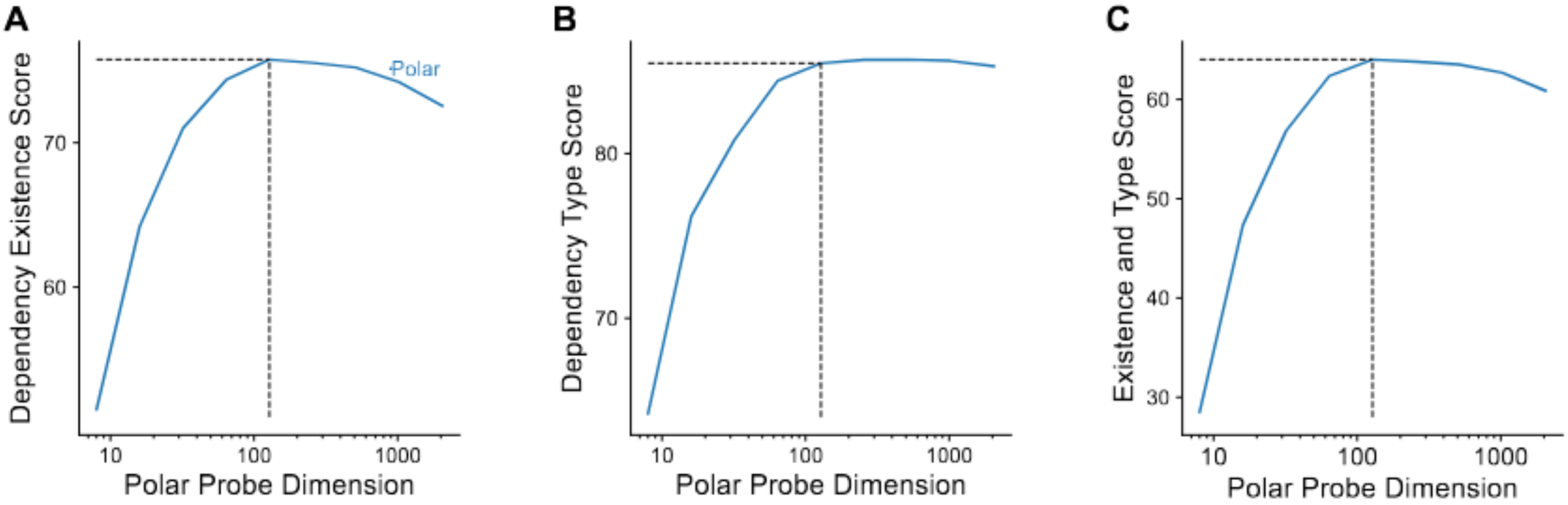
Result: how Angular and Structural interact?



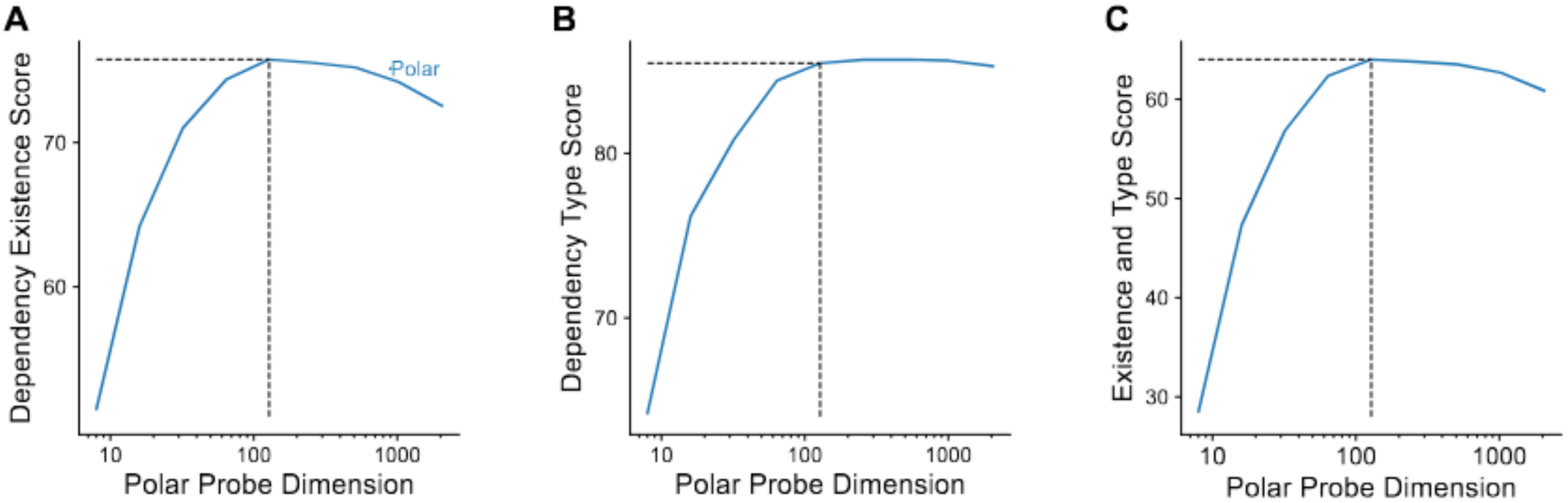
Takeaway:

- the best layer is at layer 16 (middle), aligning with structural result.
- BERT-large has the best performance than the two autoregressive models — masked LMs often encode syntax more cleanly in the current setup.
- Polar Probe keeps the old distance-based syntax signal and adds a large amount of labeled, directed information on top of it.

Result: how big is the syntax subspace?

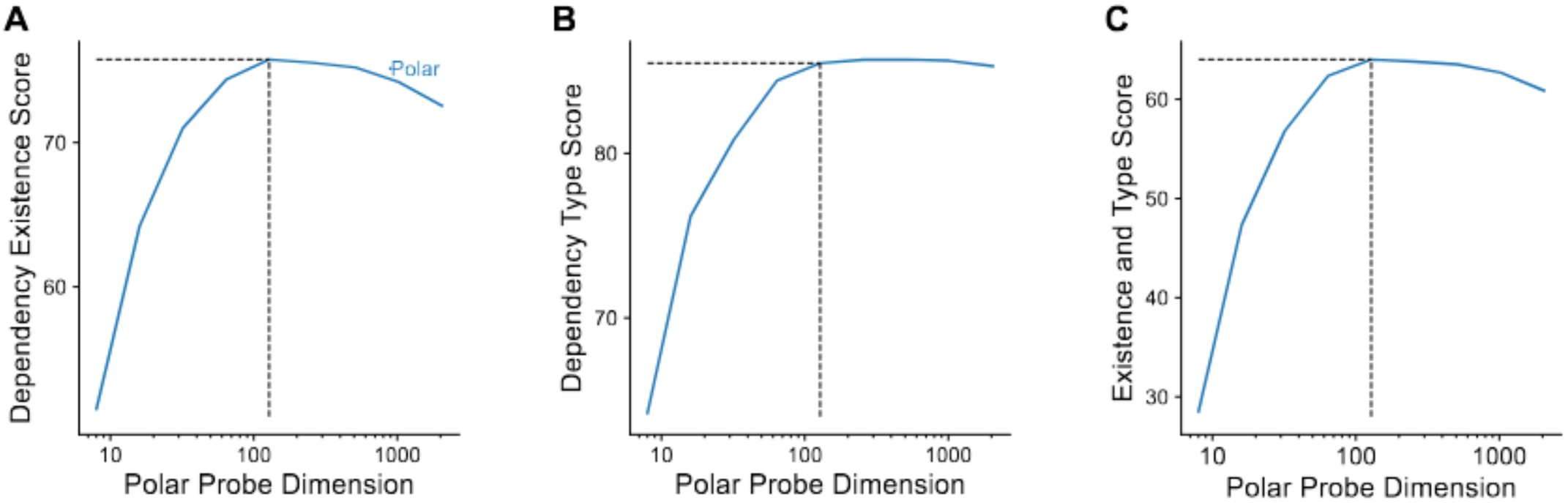


Result: how big is the syntax subspace?



A = UUAS; B = Dependency type accuracy; C = LAS

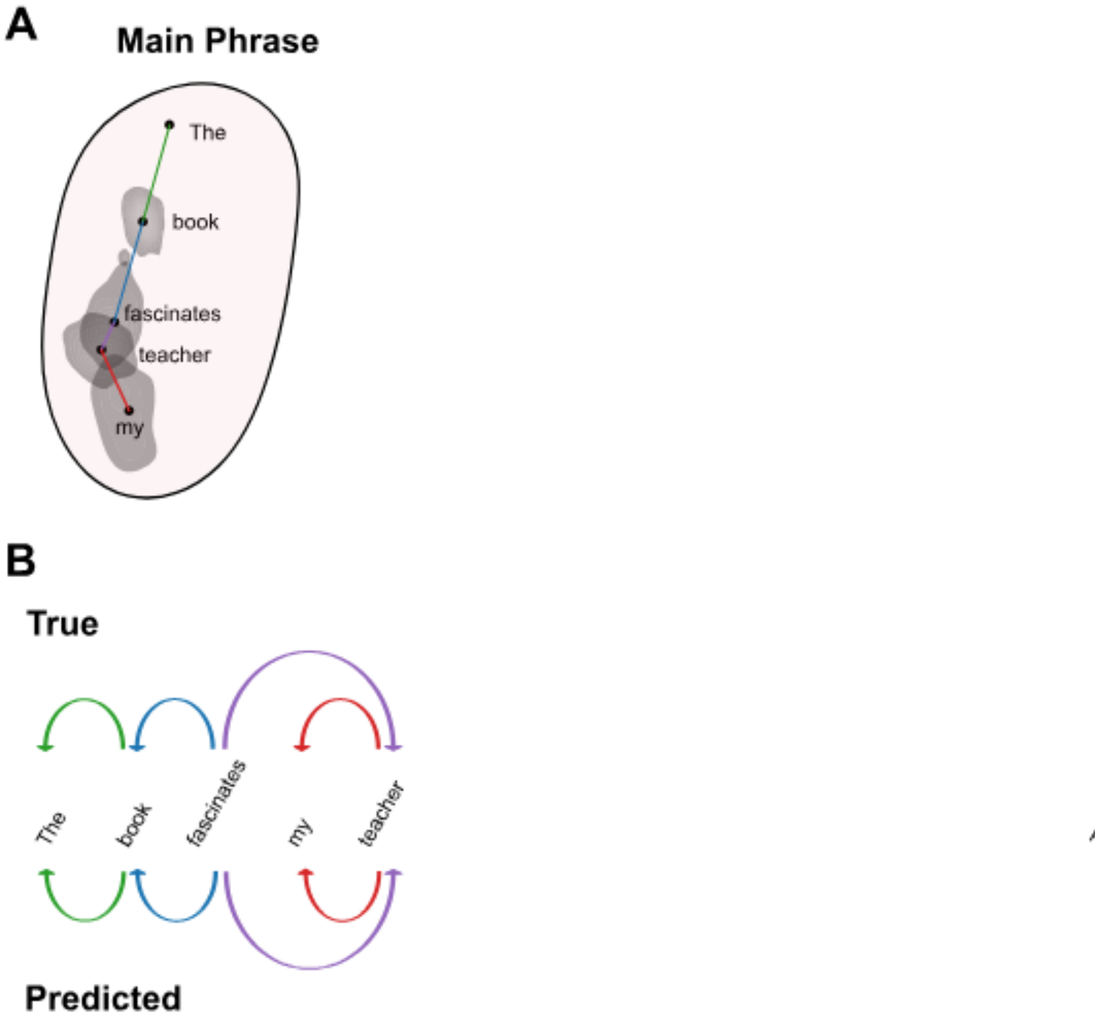
Result: how big is the syntax subspace?



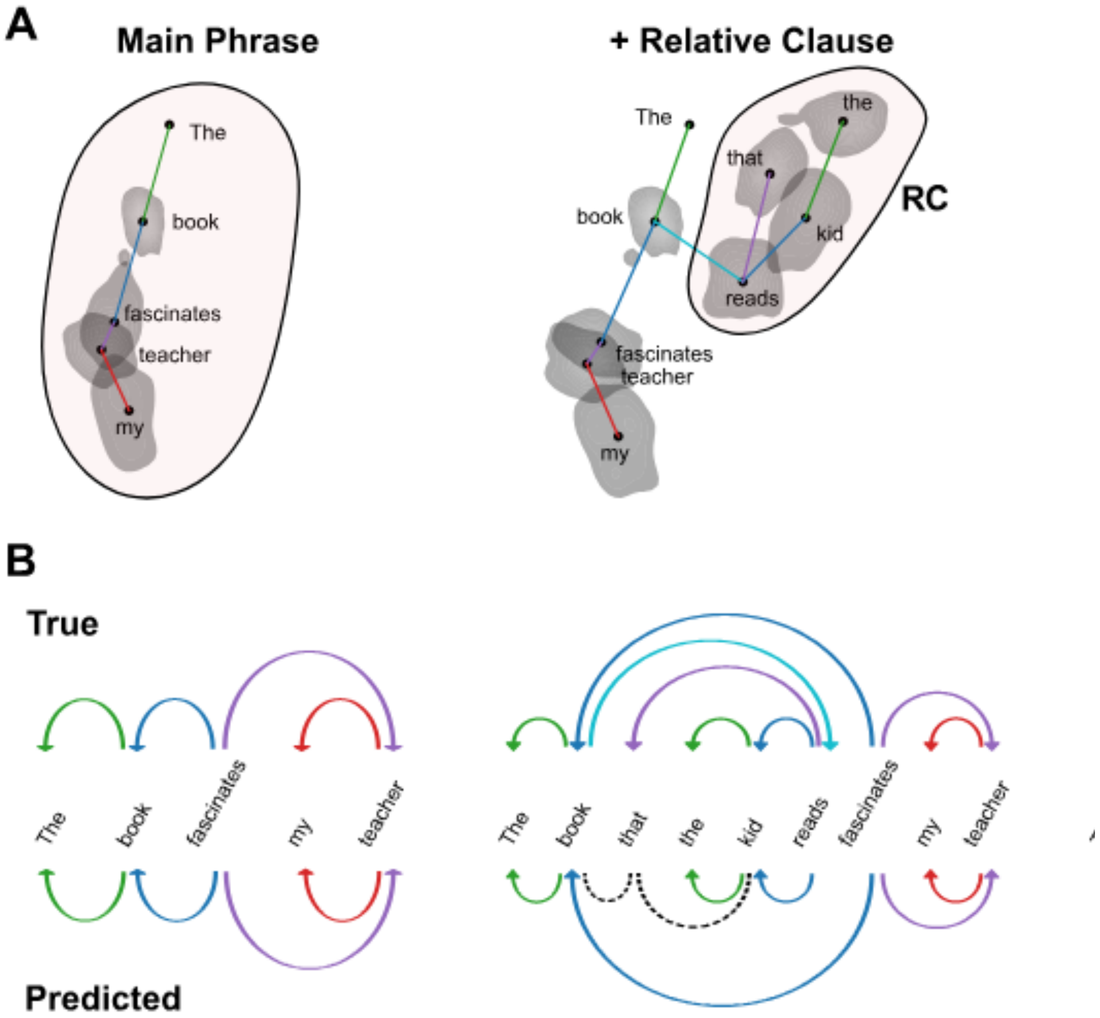
A = UUAS; B = Dependency type accuracy; C = LAS

Takeaway: same as before, the required probe space is still much smaller than the model's full hidden dimension — even the richer polar code is still fairly compact.

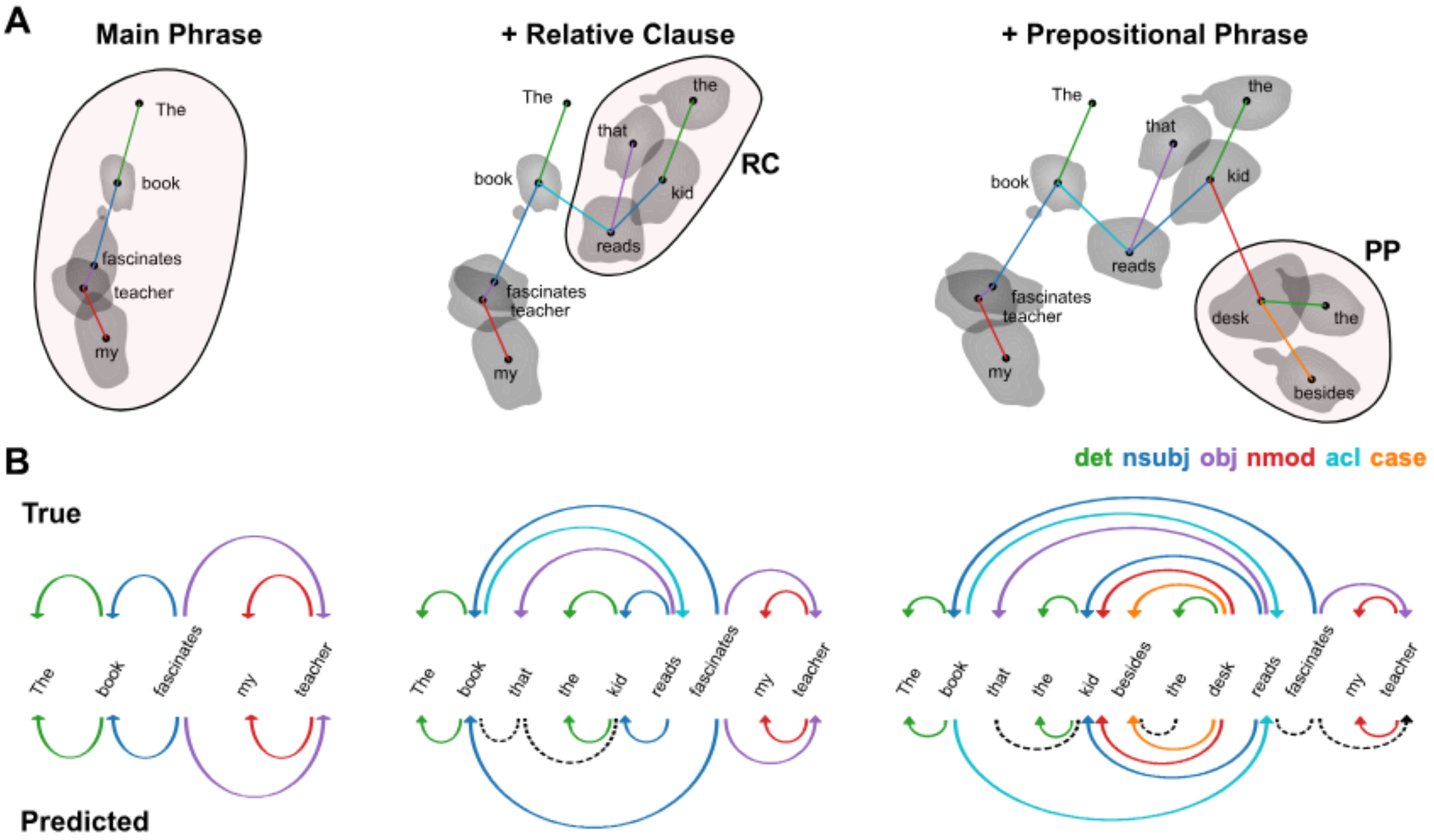
Result: on more complex sentences?



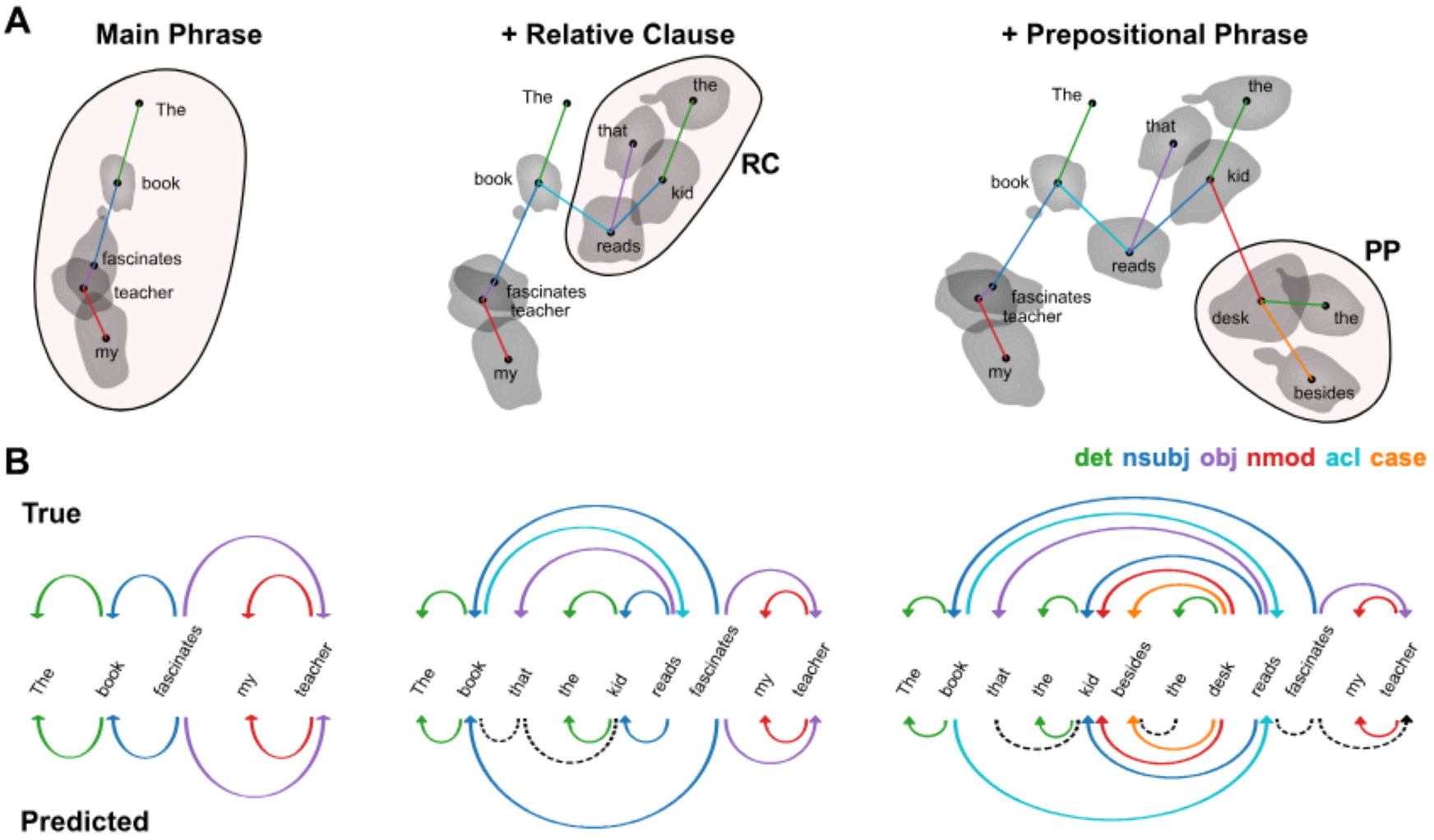
Result: on more complex sentences?



Result: on more complex sentences?



Result: on more complex sentences?



Takeaway: syntax is represented in a **systematic coordinate system** that stays stable across recursive embedding.

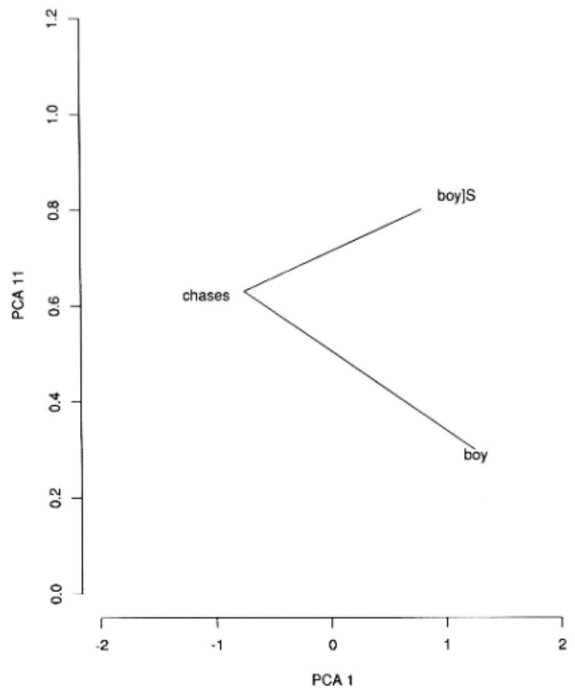
See also Elman 1991

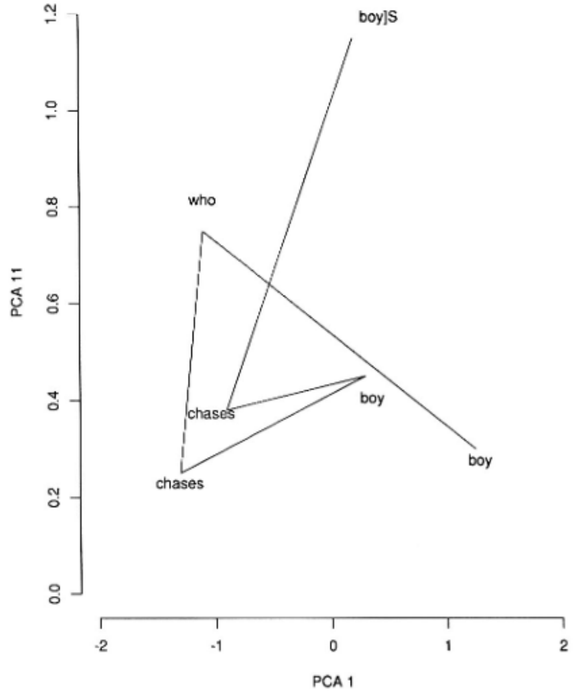
Distributed Representations, Simple Recurrent Networks, and Grammatical Structure

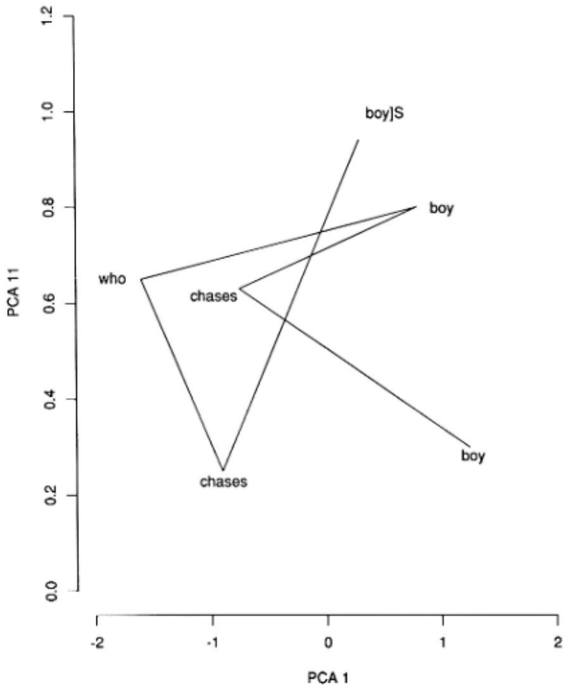
JEFFREY L. ELMAN (ELMAN@CRL.UCSD.EDU)
Departments of Cognitive Science and Linguistics, University of California, San Diego

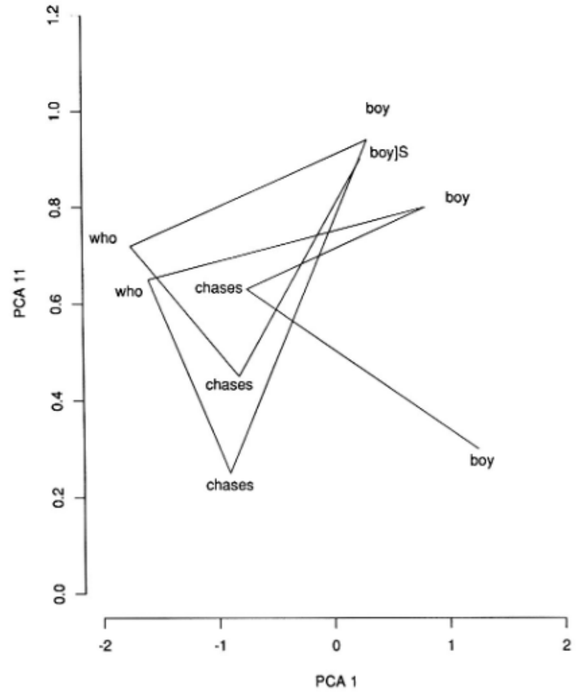
(10a) boy chases boy.
(10b) boy chases boy who chases boy.
(10c) boy who chases boy chases boy.
(10d) boy chases boy who chases boy who chases boy.

Relative clauses in small neural networks!

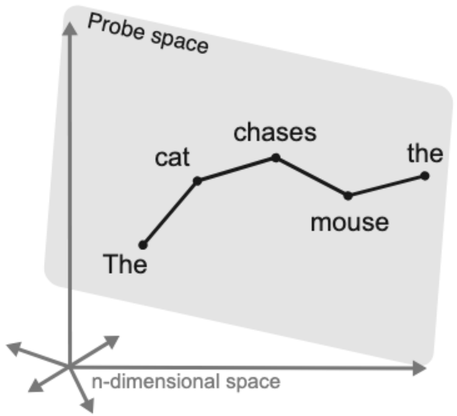






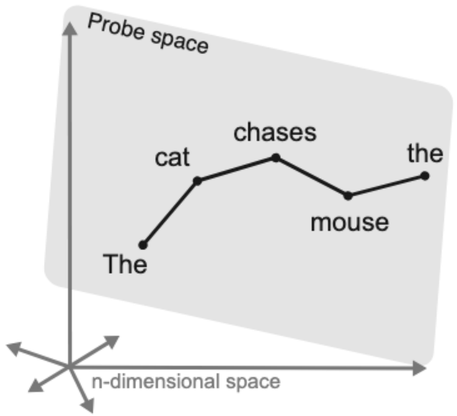


Summary



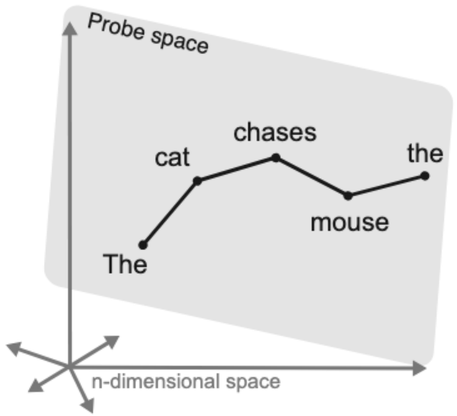
Summary

- **Hewitt & Manning:** syntax behaves like a *metric geometry*;



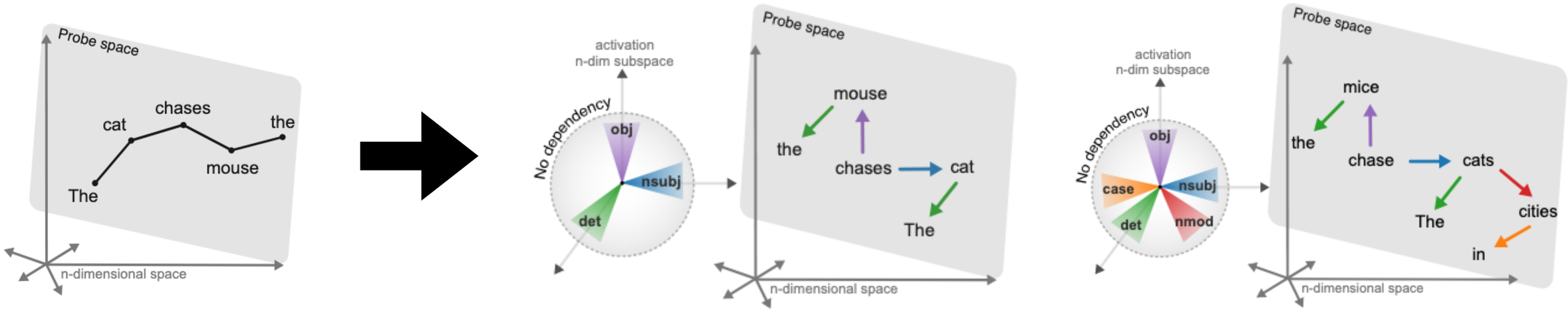
Summary

- **Hewitt & Manning:** syntax behaves like a *metric geometry*;
- **Diego-Simón et al**: the full dependency syntax behaves more like a *polar geometry*, where magnitude represents hierarchical depth/distance and angle represents type and head/direction.



Summary

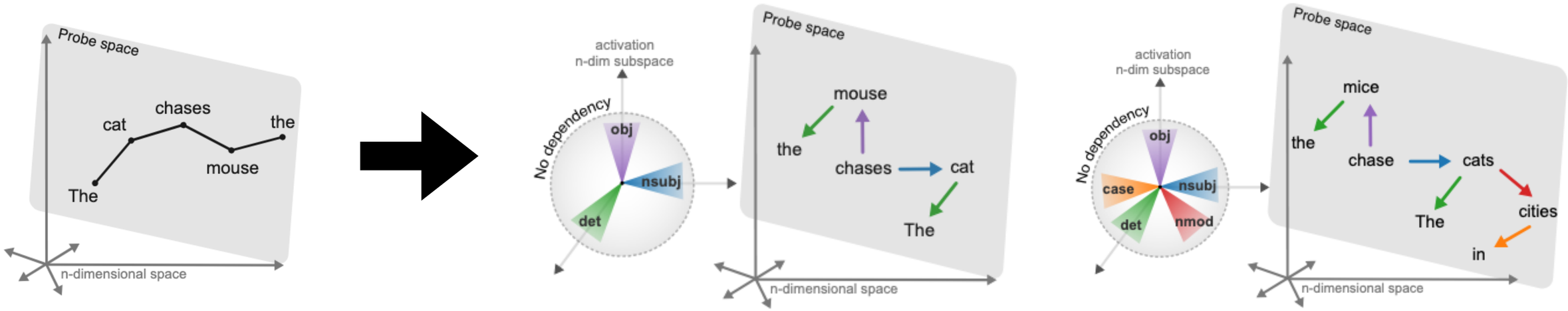
- **Hewitt & Manning:** syntax behaves like a *metric geometry*;
- **Diego-Simón et al**: the full dependency syntax behaves more like a *polar geometry*, where magnitude represents hierarchical depth/distance and angle represents type and head/direction.



Summary

- **Hewitt & Manning:** syntax behaves like a *metric geometry*;
- **Diego-Simón et al**: the full dependency syntax behaves more like a *polar geometry*, where magnitude represents hierarchical depth/distance and angle represents type and head/direction.

The role of probing = to find a compact subspace that organizes the syntactic information into this polar, human interpretable way!

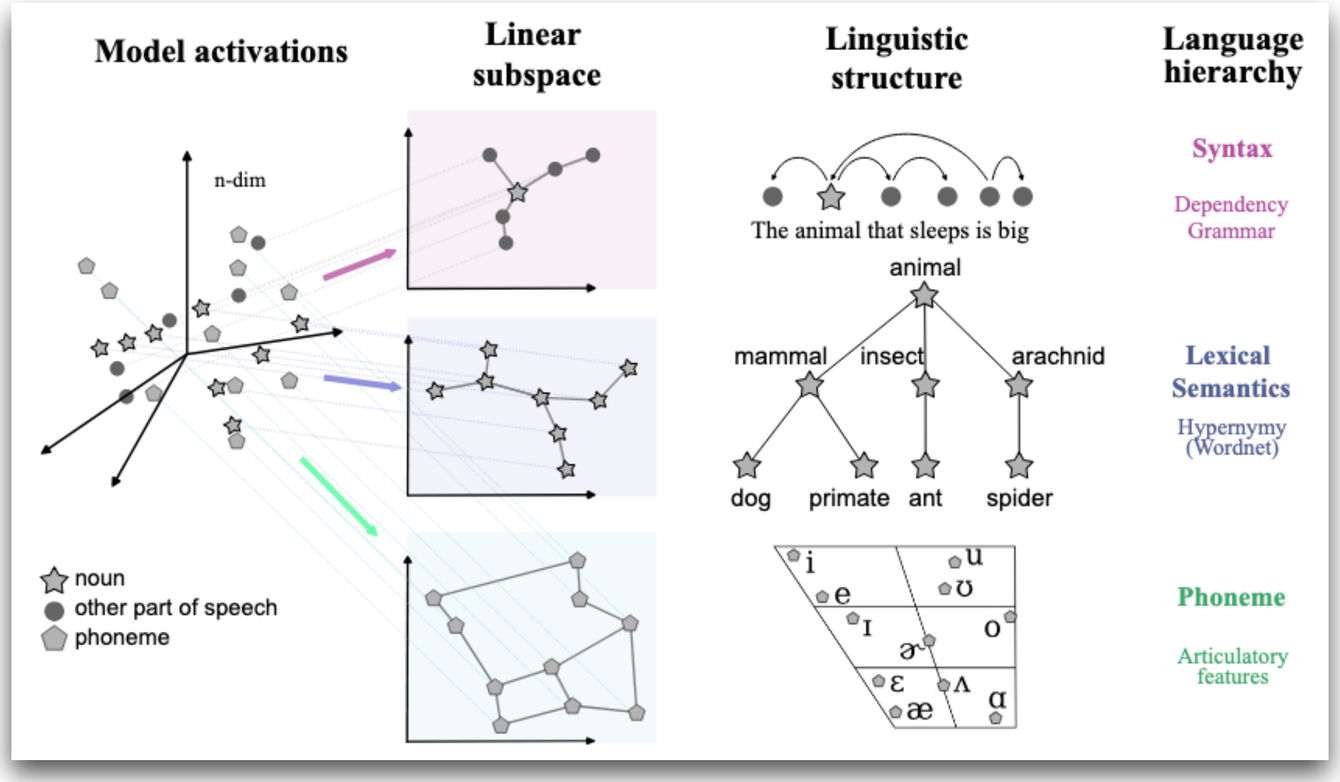


Orhan et al. 2026

Generalizing structural probe to phonetics and lexical semantics, on acquisition as well

Motivating Questions

- Can we also find linear subspaces for different kinds of linguistic structures — e.g., phonemic, lexical-semantics, in addition to dependency relations?
- How do those structures emerge during training? — to what extent is it analogous to human language acquisition?



Methods

Methods

- **One unified framework**: structural probe (same procedure as Hewitt & Manning);

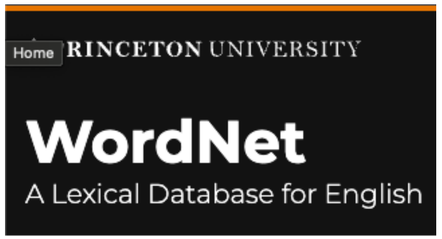
Methods

- **One unified framework**: structural probe (same procedure as Hewitt & Manning);
- **Models**
  - Text: Mistral-7B ($k = 4096$), Llama-2-7b ($k = 4096$);
  - Speech: Wav2Vec 2.0 (English, $k = 4096$), Controls (trained on French + music + environmental sounds);

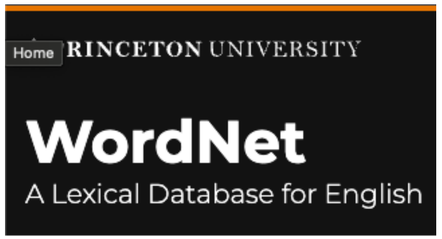
Methods

- **One unified framework**: structural probe (same procedure as Hewitt & Manning);
- **Models**
  - Text: Mistral-7B ($k = 4096$), Llama-2-7b ($k = 4096$);
  - Speech: Wav2Vec 2.0 (English, $k = 4096$), Controls (trained on French + music + environmental sounds);
- **Why speech models?**
  - Speech models' input is continuous audio, and the models are trained to reconstruct masked latent speech segments (differ from next-token pred);
  - It makes sense to test whether the model can discover useful information about phonemic structure, and whether syntax & semantics can emerge from raw speech rather than discrete tokenized text.
  - Can control to rule out the confound of detecting simpler acoustic regularities.

Methods, cont.

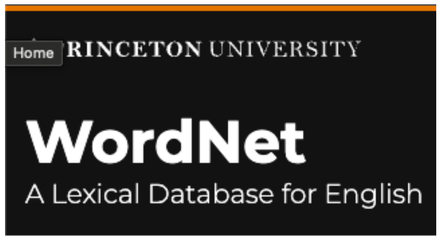


Methods, cont.



Text Input

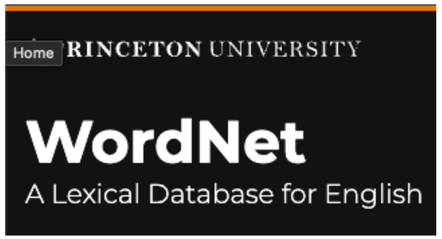
Methods, cont.



Text Input

Acoustic

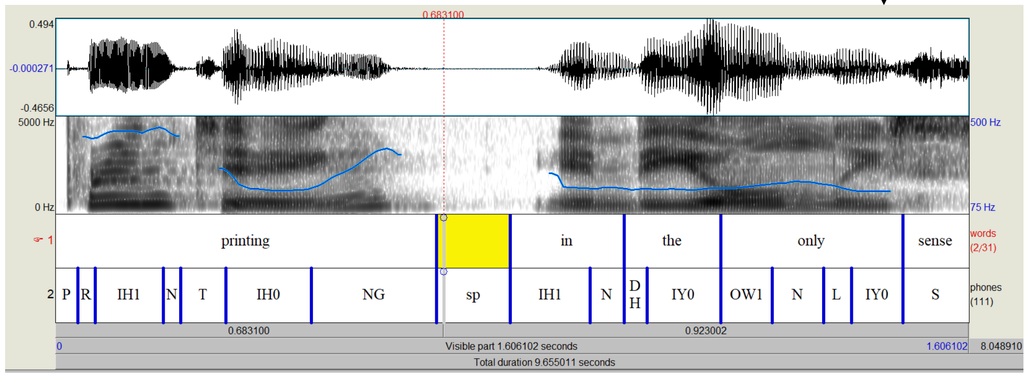
Methods, cont.



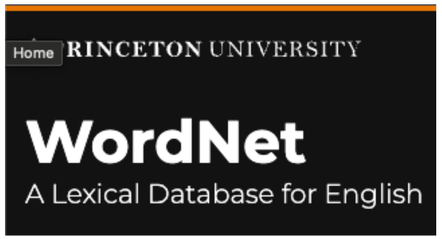
Text Input

Acoustic

Time information for word/phoneme segmentation



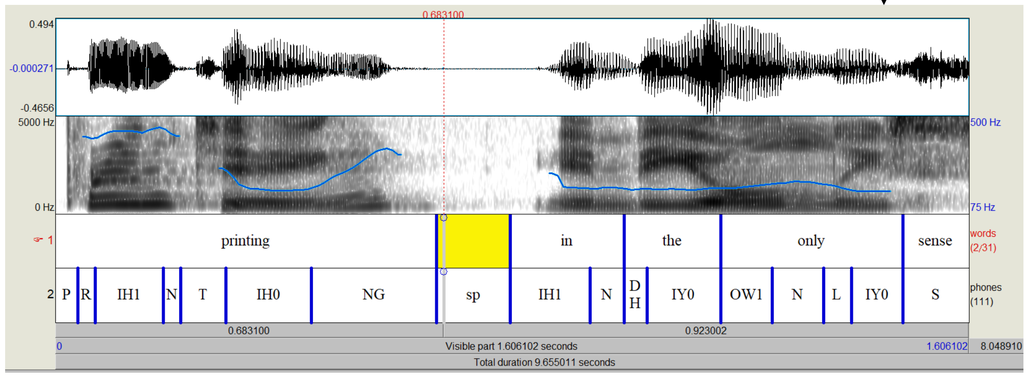
Methods, cont.



Text Input

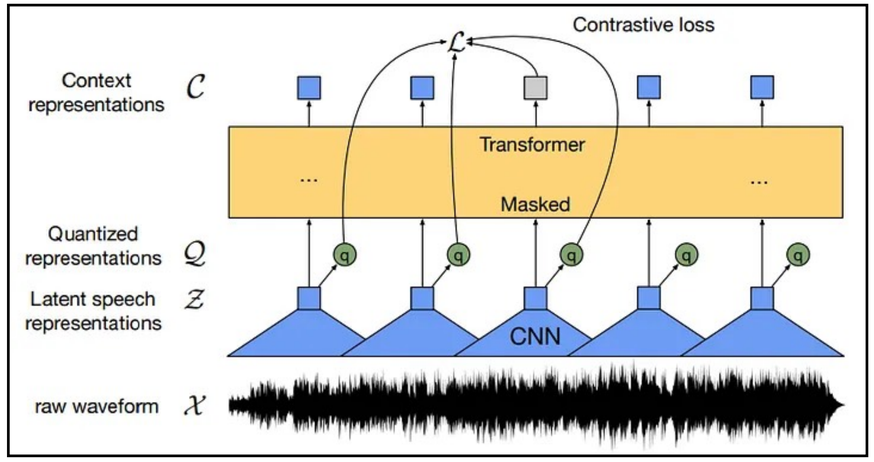
Acoustic

Time information for word/phoneme segmentation



Waveform & time

Wav2Vec 2.0



Methods, cont.

Home RINCETON UNIVERSITY

WordNet

A Lexical Database for English

UniversalDependencies/
UD_English-EWT

English data

23 Contributors · 144 Issues · 222 Stars · 45 Forks

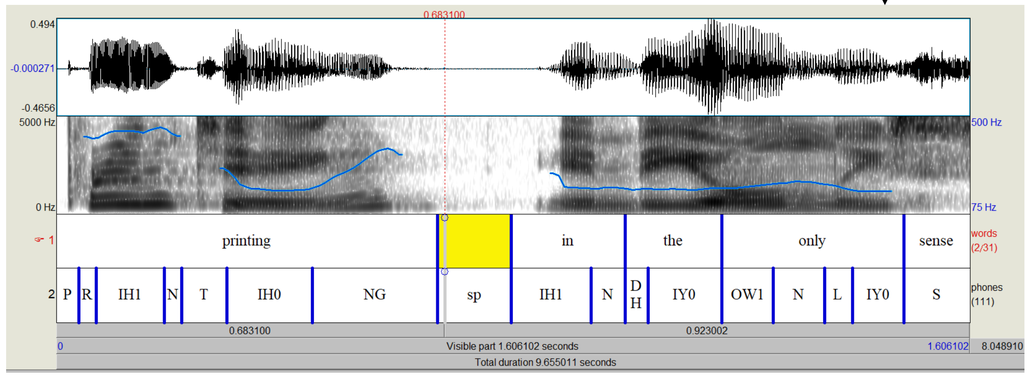
Text Input

Seamless M4T v2

Acoustic

Montreal Forced Aligner

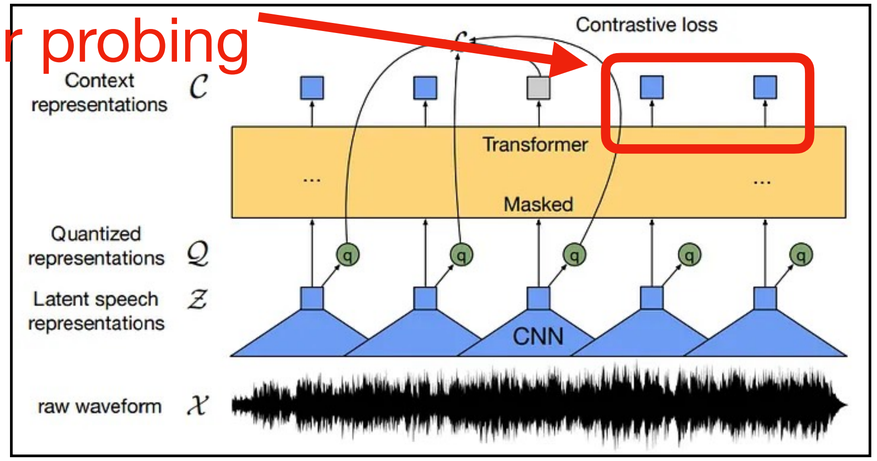
Time information for word/phoneme segmentation



Waveform & time

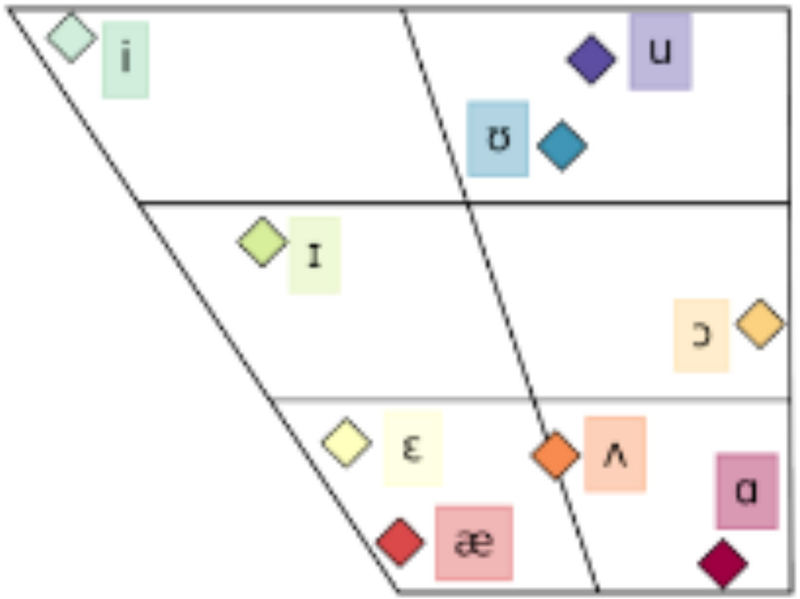
Activations for probing

Wav2Vec 2.0



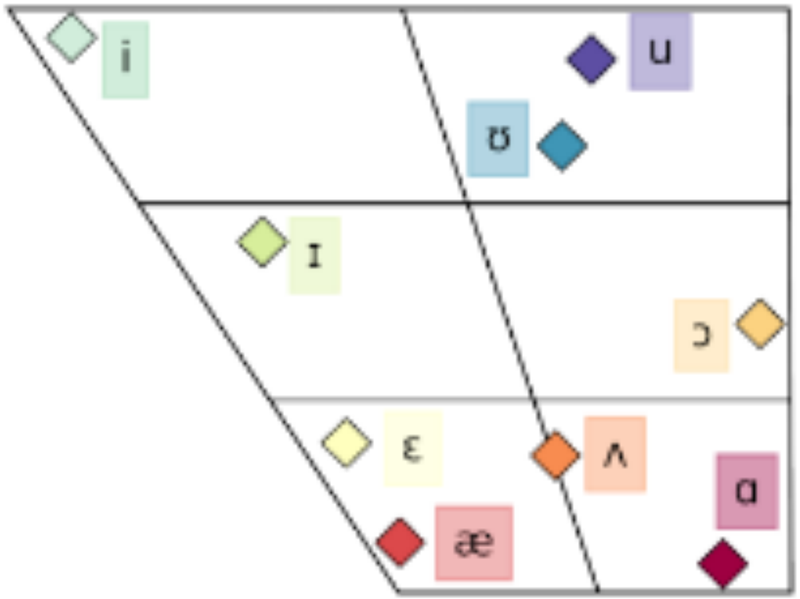
Phonemic Articulation

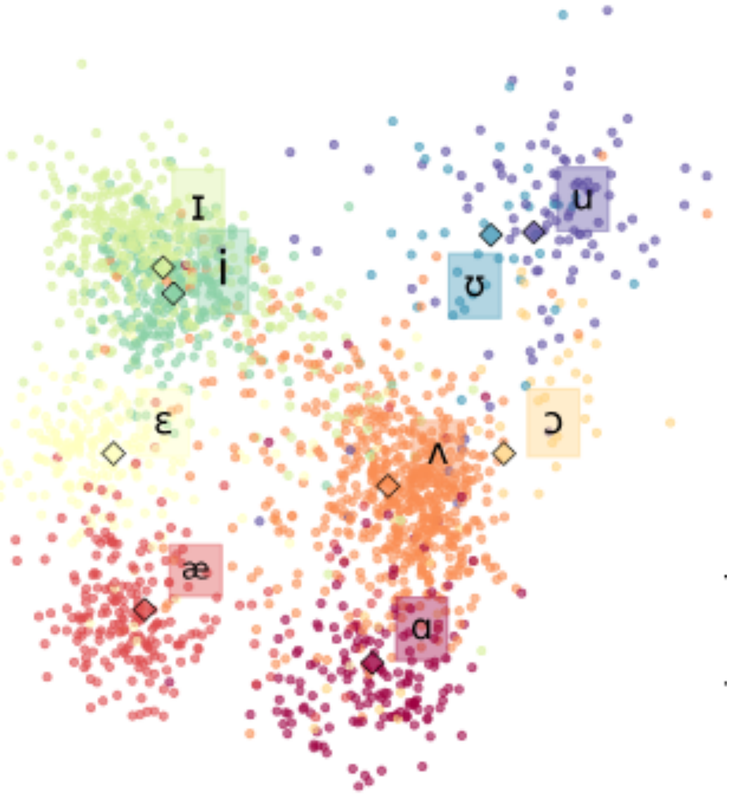
- **Hypothesis**: speech models organize phonemes into structures that reflect articulatory features, rather than just treating them as separate categories.
- **Structural metric**: gold distance is articulatory-feature dissimilarity (Mortensen et al. 2016) between two phoneme activations.



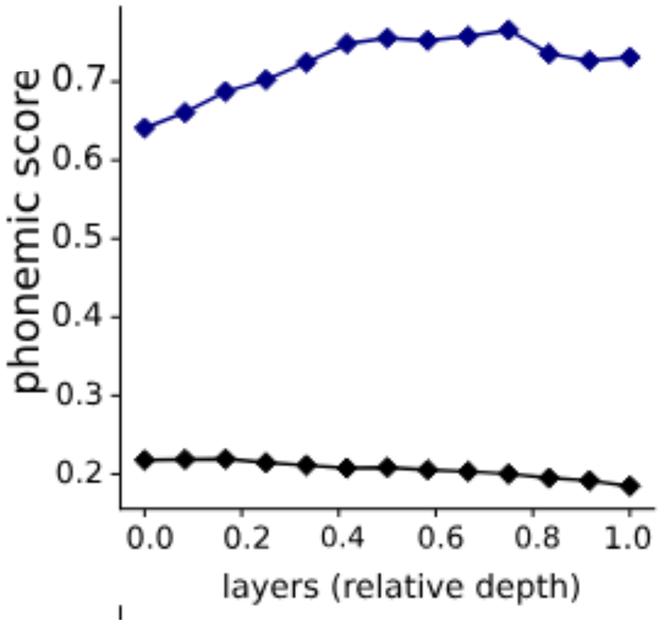
Phonemic Articulation

- **Hypothesis**: speech models organize phonemes into structures that reflect articulatory features, rather than just treating them as separate categories.
- **Structural metric**: gold distance is articulatory-feature dissimilarity (Mortensen et al. 2016) between two phoneme activations.

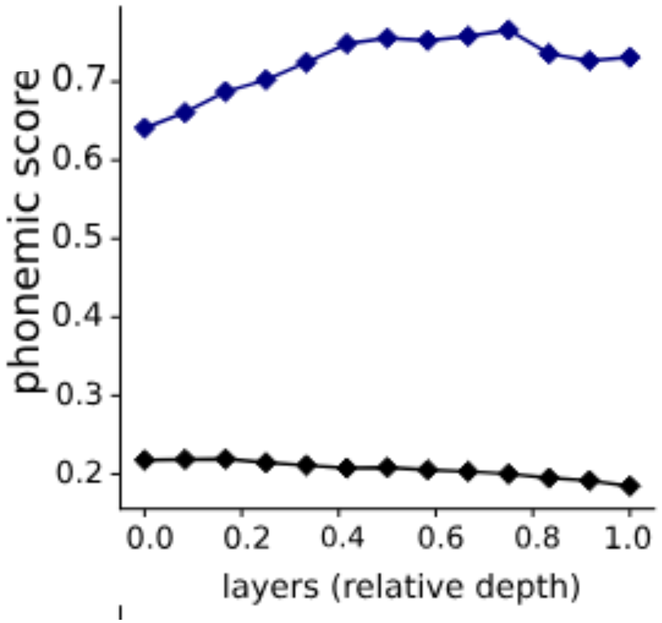




Phonemic Articulation

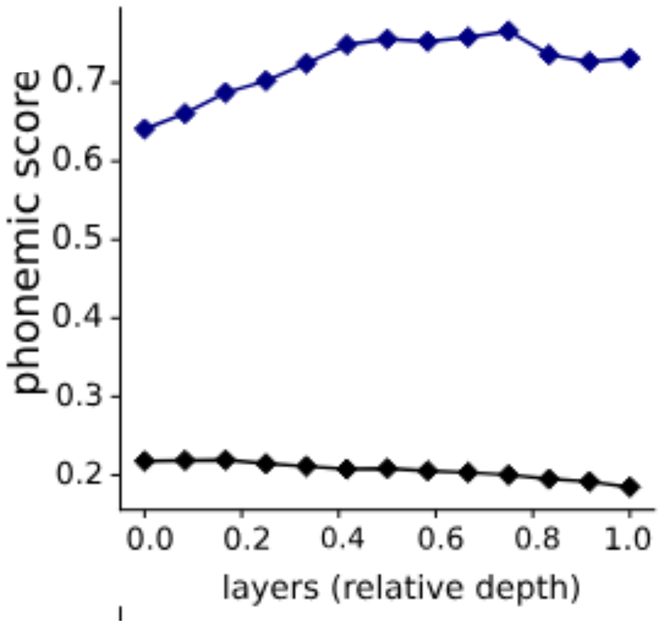


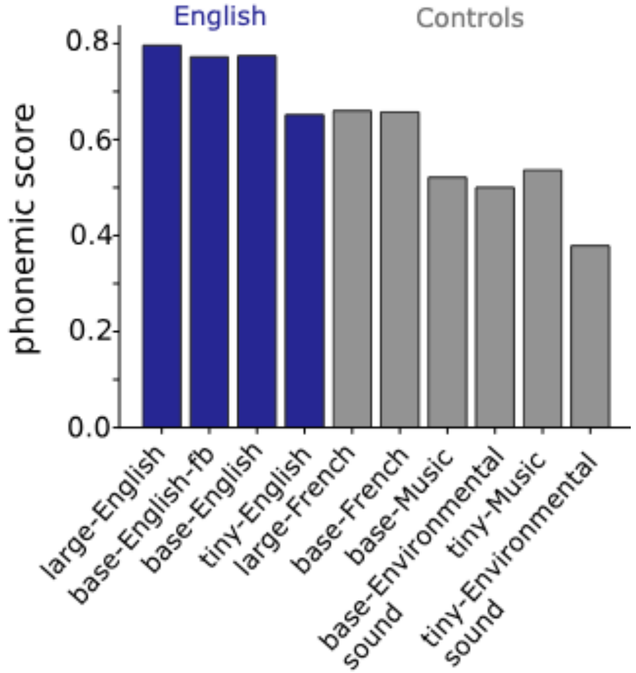
Phonemic Articulation



- **Left**: pretrained models (top) consistently outperform random initializations (bottom);

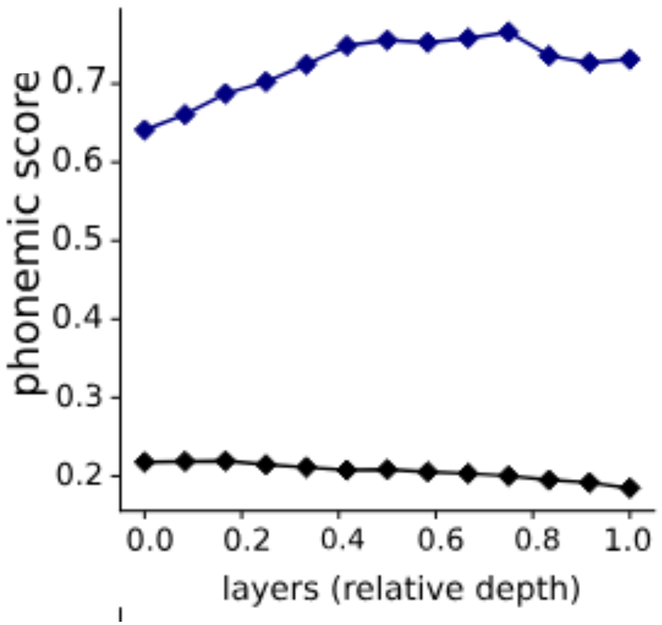
Phonemic Articulation

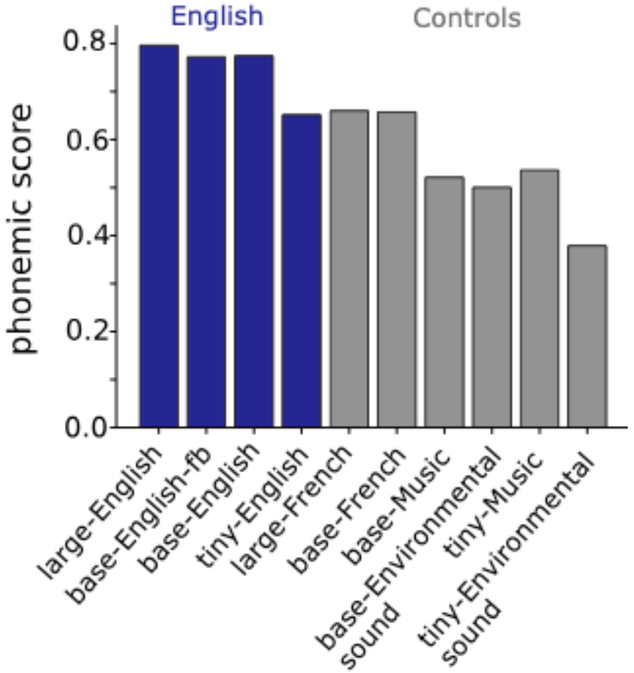


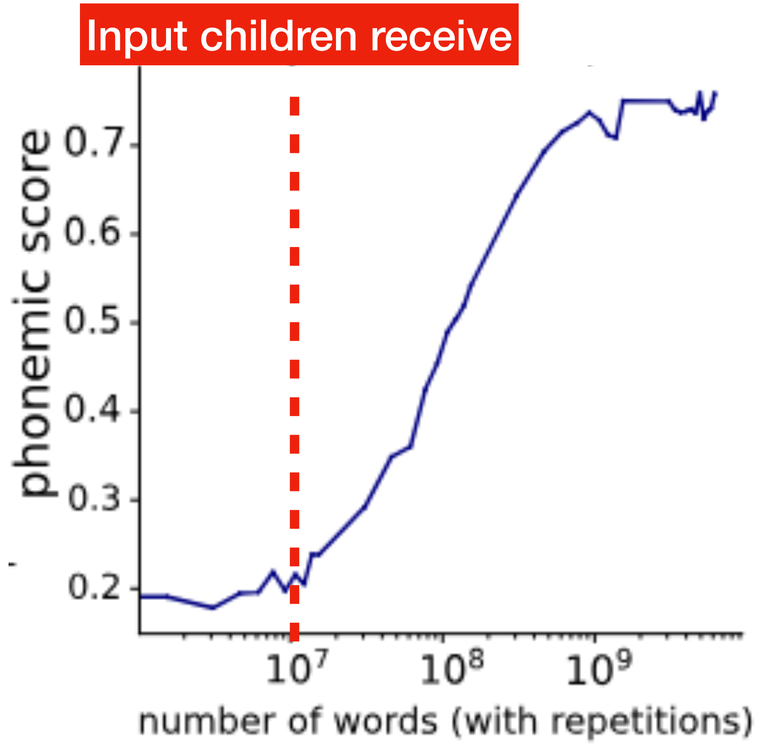


- **Left**: pretrained models (top) consistently outperform random initializations (bottom);
- **Mid**: English pretraining are better than control models (?); larger models are better;

Phonemic Articulation



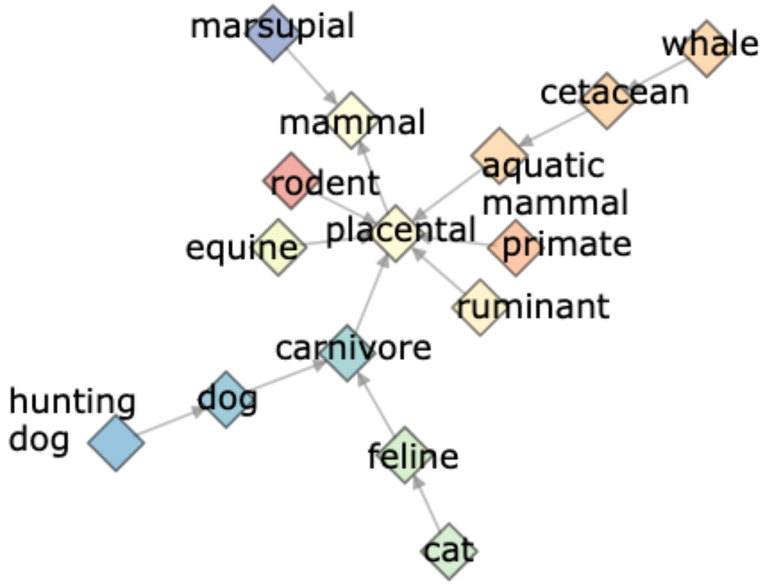




- **Left**: pretrained models (top) consistently outperform random initializations (bottom);
- **Mid**: English pretraining are better than control models (?); larger models are better;
- **Right**: the model gradually acquired this phonemic representation with 10^9 data, 100 times more than children input.

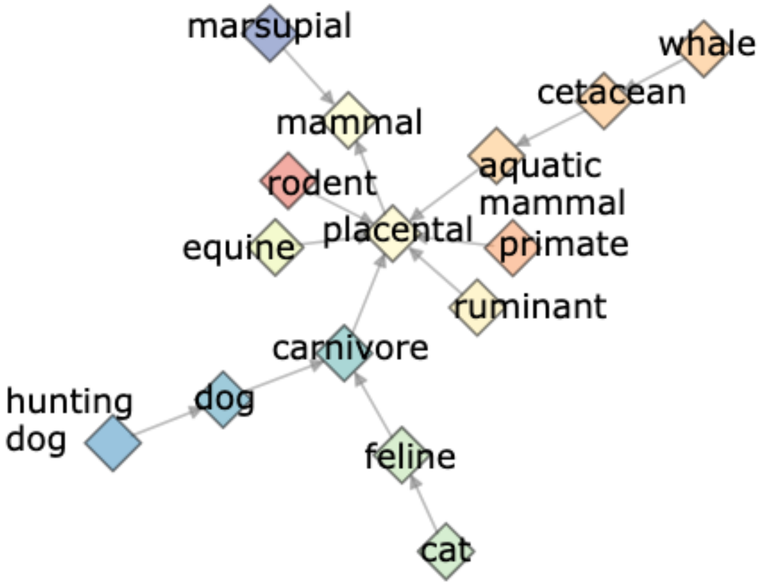
Lexical Semantics

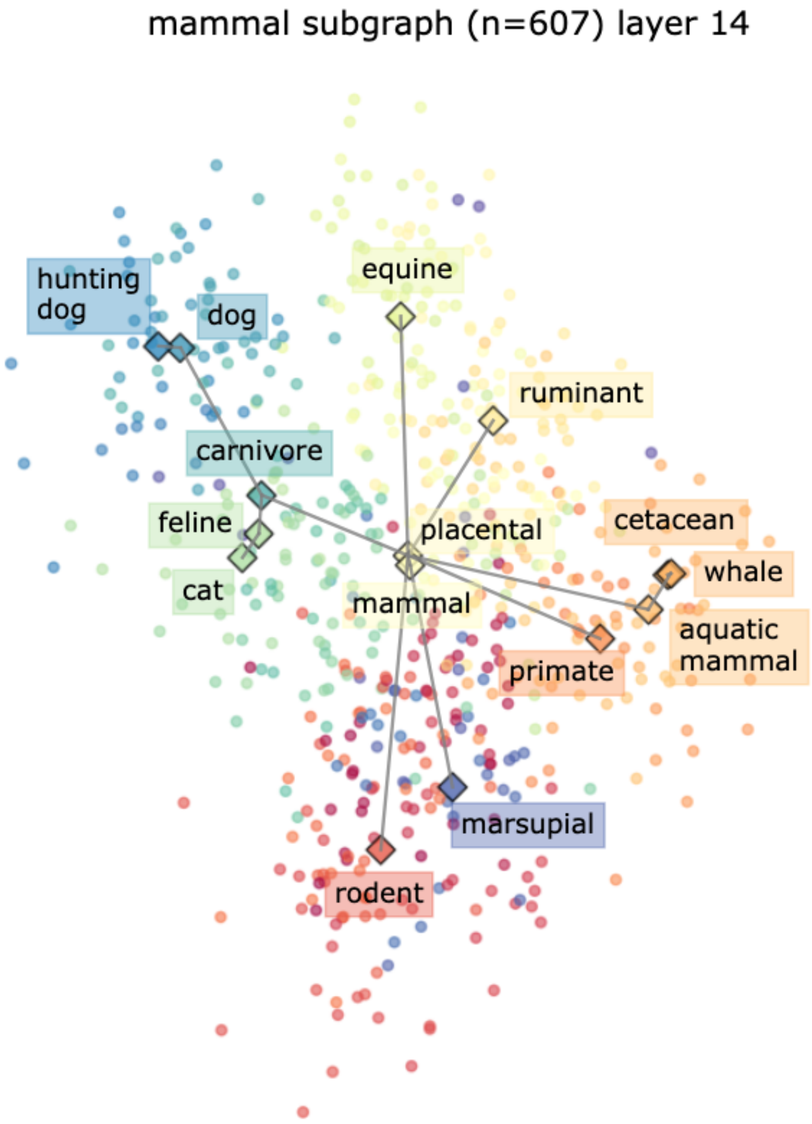
- **Hypothesis**: text and speech models organize lexical informations into semantic structures, specifically, hypernym-hyponym relations.
- **Structural metric**: gold distance is path length in the WordNet hypernym graph.



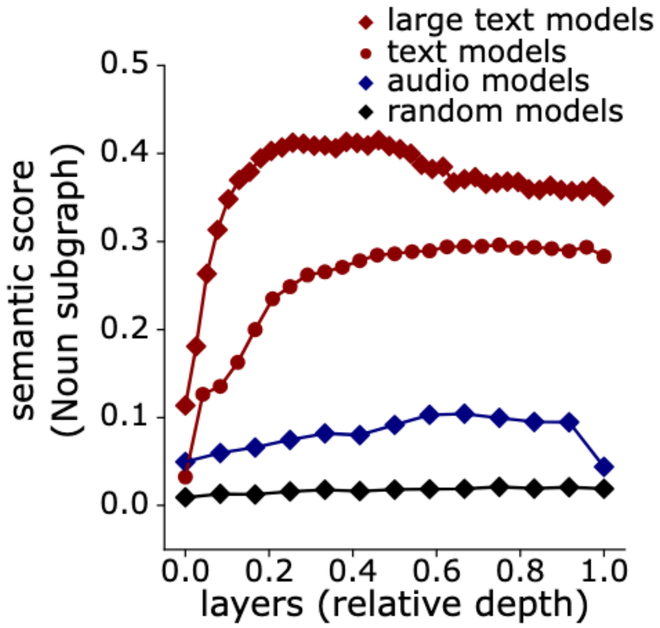
Lexical Semantics

- **Hypothesis**: text and speech models organize lexical informations into semantic structures, specifically, hypernym-hyponym relations.
- **Structural metric**: gold distance is path length in the WordNet hypernym graph.

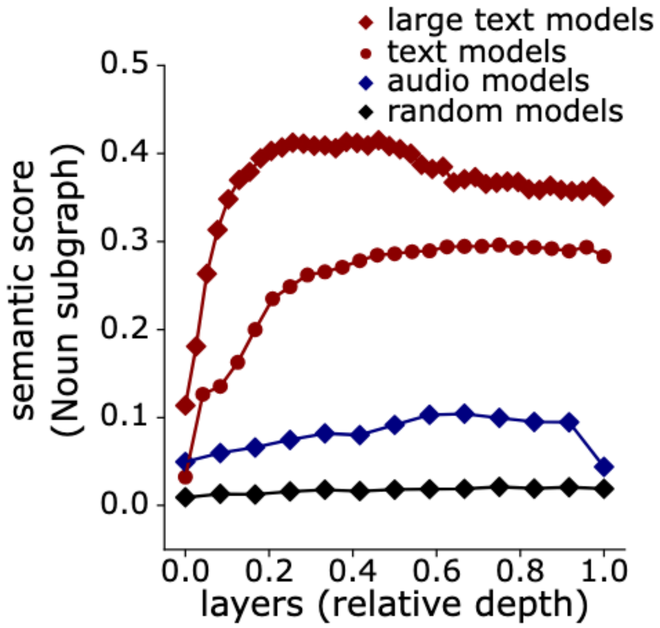




Lexical Semantics

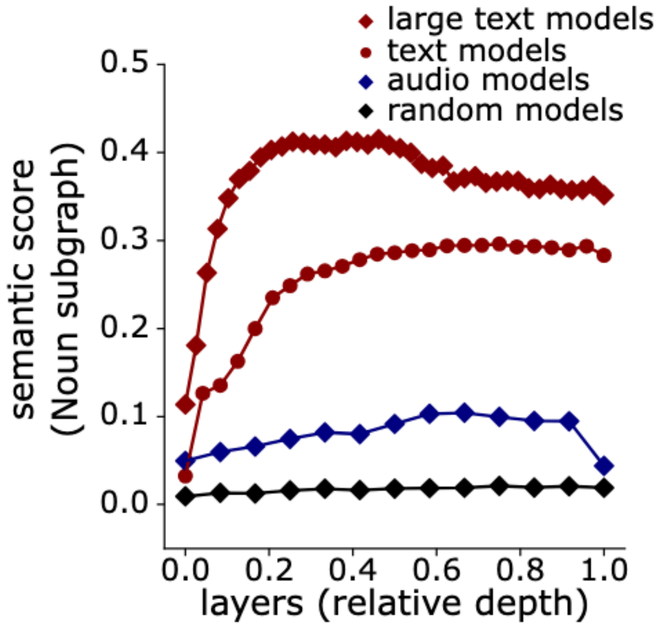


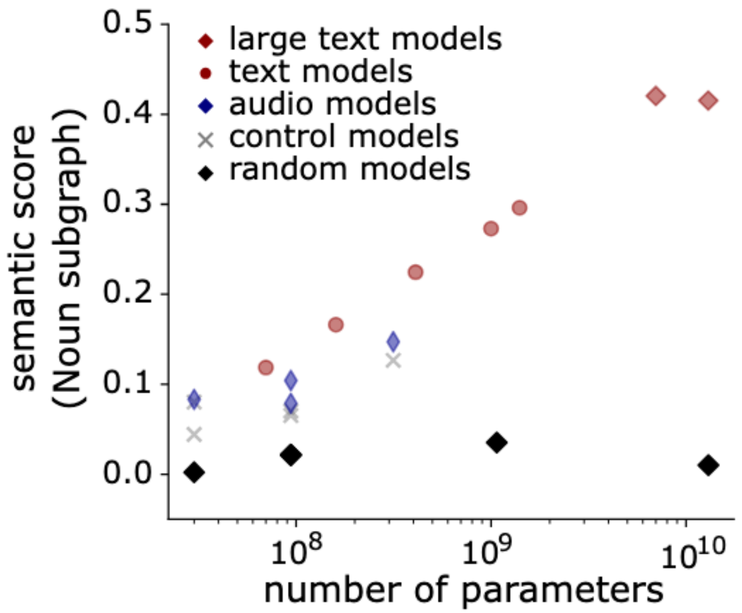
Lexical Semantics



- **Left**: intermediate layers are better; scores are lower than phonemic;

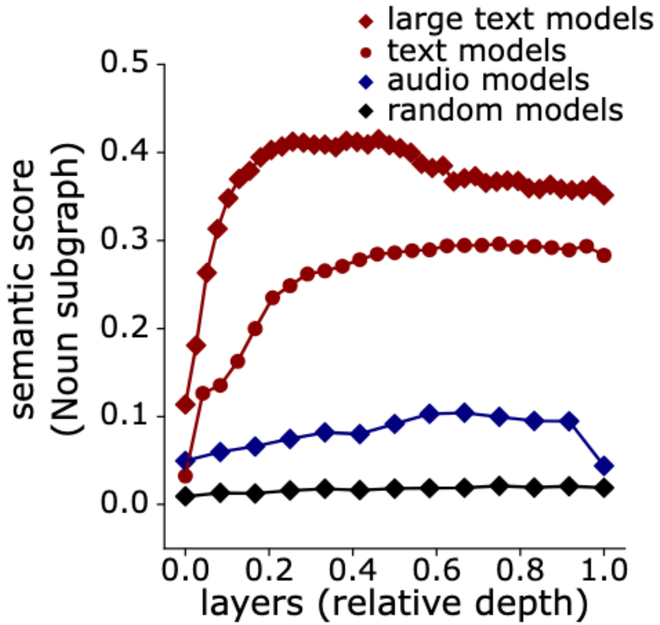
Lexical Semantics

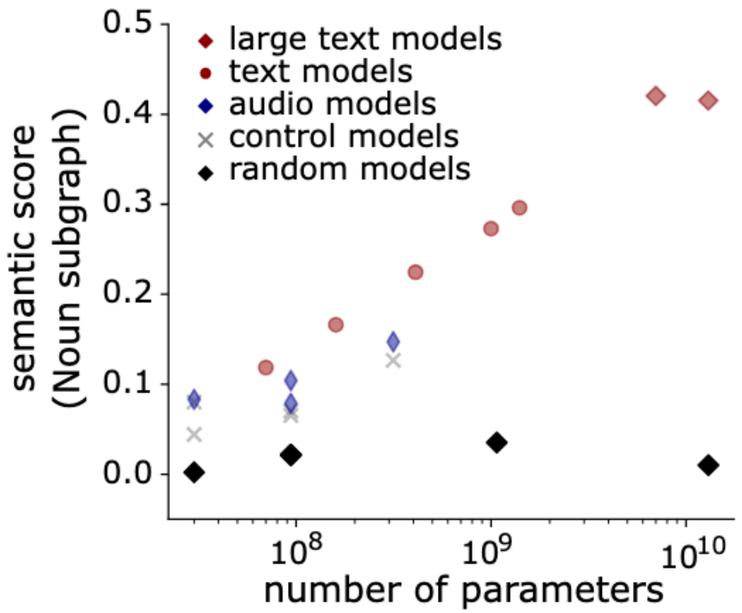


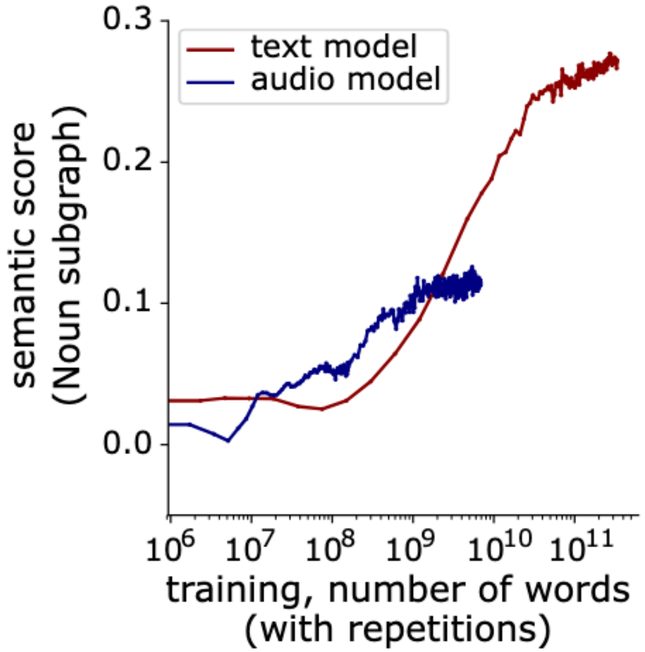


- **Left**: intermediate layers are better; scores are lower than phonemic;
- **Mid**: text models outperforms audio models; larger models are better; control achieves similar performance as audio models… confounding with acoustic properties…?!

Lexical Semantics







- **Left**: intermediate layers are better; scores are lower than phonemic;
- **Mid**: text models outperforms audio models; larger models are better; control achieves similar performance as audio models… confounding with acoustic properties…?!
- **Right**: audio semantic scores rise faster initially, but then saturate, whereas text-model scores continue to improve more substantially;

Taking all together: developmental stages?

- **Goal**: to see how the three properties emerge in relative order.

Phonemes

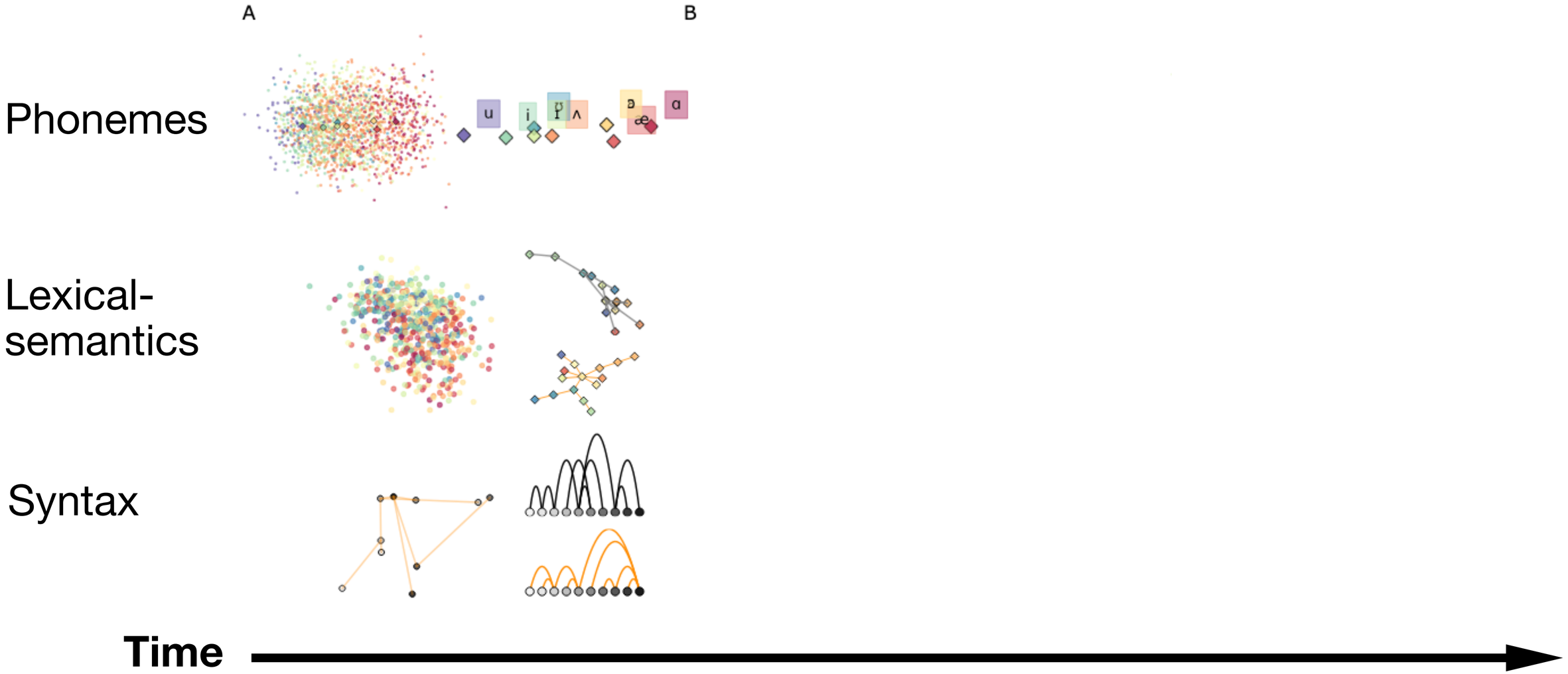
Lexical-
semantics

Syntax

Time 

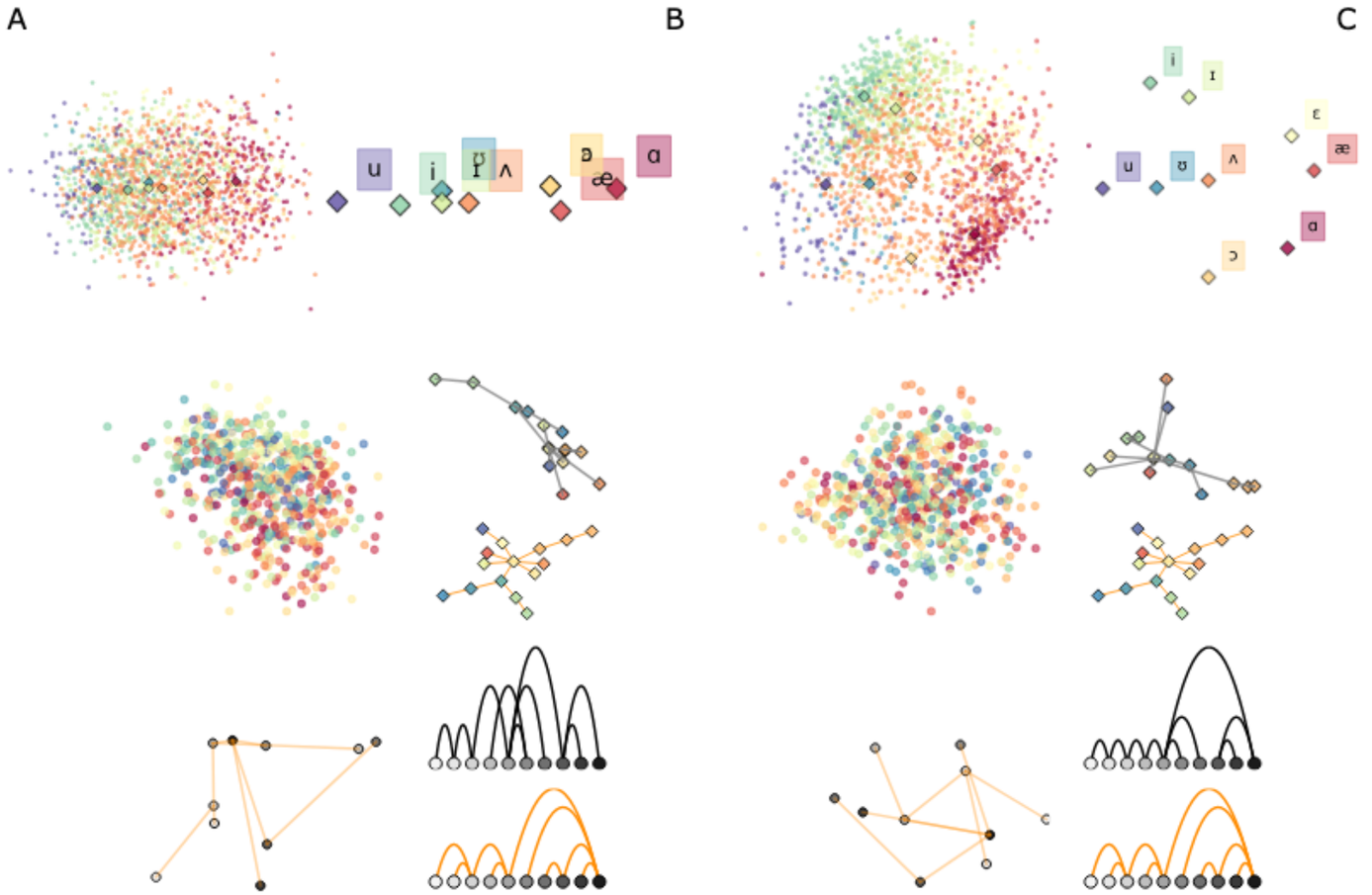
Taking all together: developmental stages?

- **Goal**: to see how the three properties emerge in relative order.



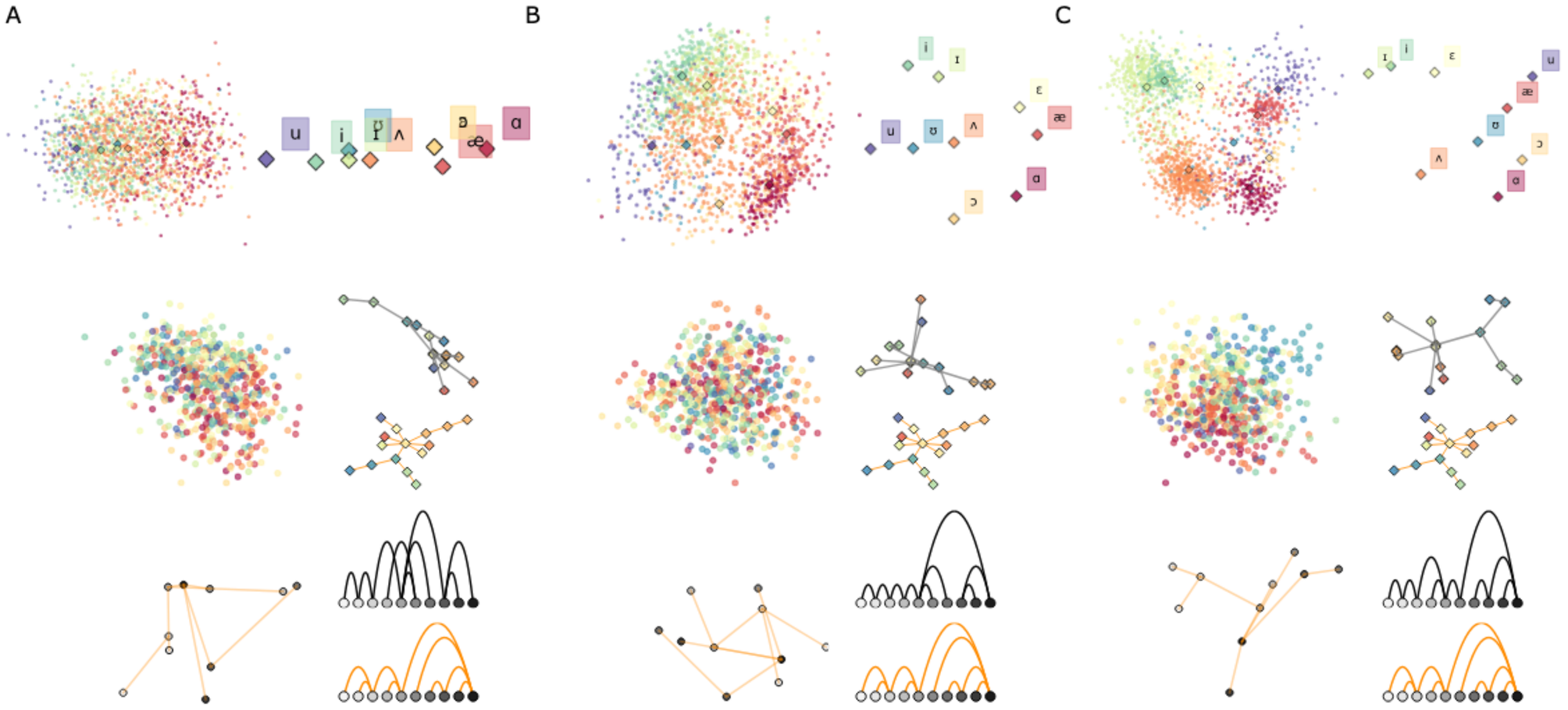
Taking all together: developmental stages?

- **Goal**: to see how the three properties emerge in relative order.



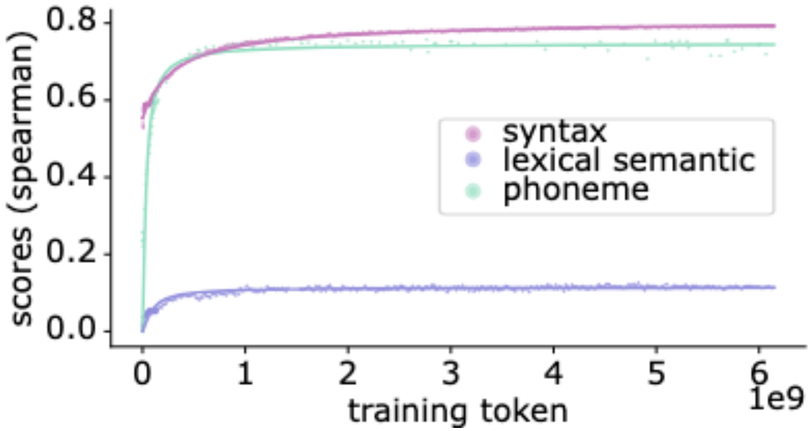
Taking all together: developmental stages?

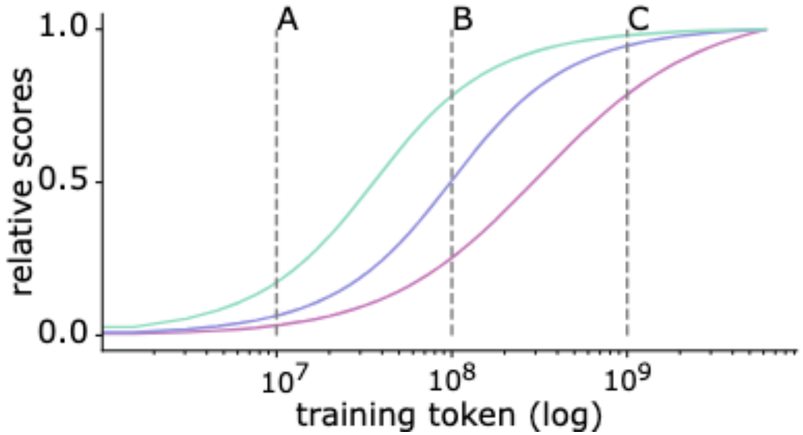
- **Goal**: to see how the three properties emerge in relative order.

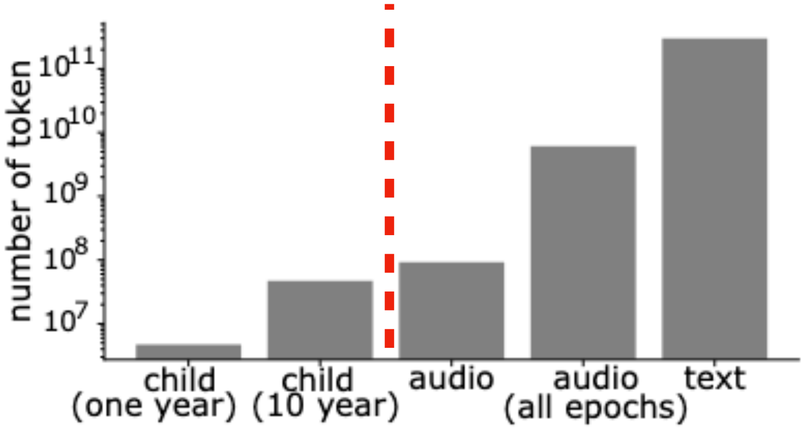


Taking all together: developmental stages?

Wav2Vec 2.0

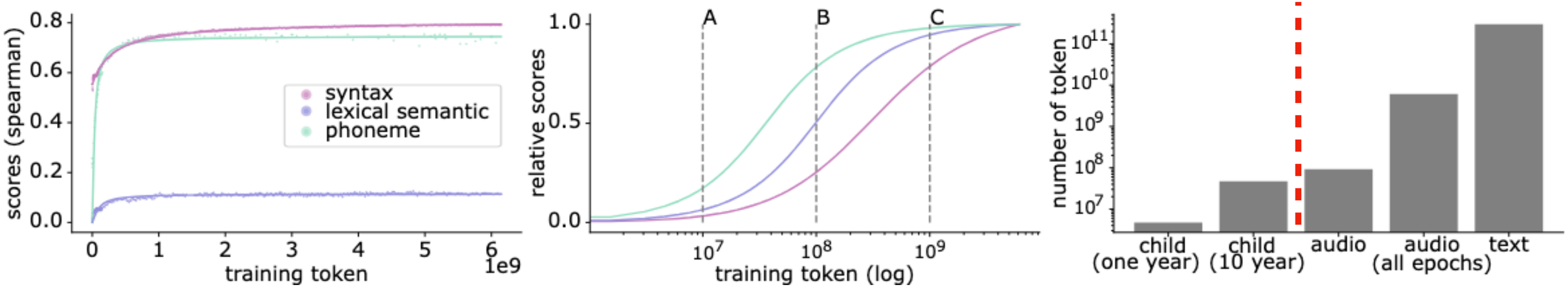






Taking all together: developmental stages?

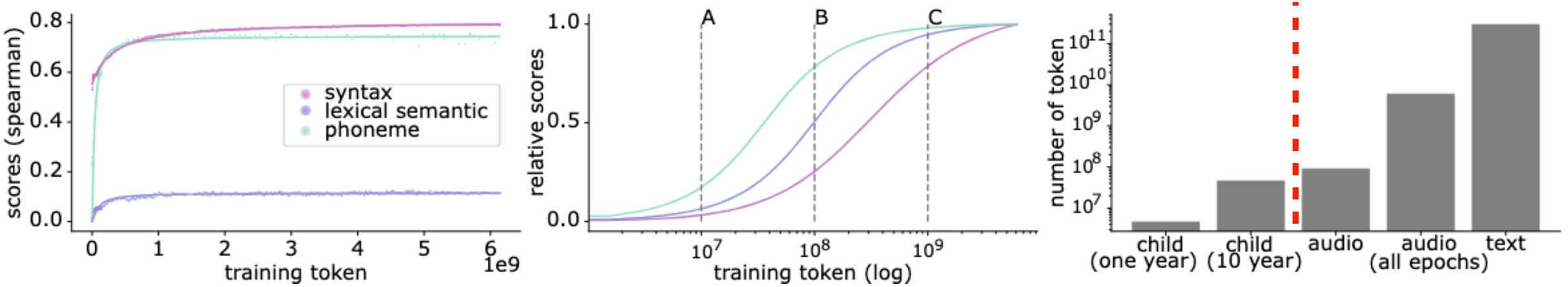
Wav2Vec 2.0



- **Left and Mid**: demonstrating successive order of acquisition (left = raw score, mid = cumulative score), all showing phoneme → lexical semantics → syntax.

Taking all together: developmental stages?

Wav2Vec 2.0



- **Left and Mid**: demonstrating successive order of acquisition (left = raw score, mid = cumulative score), all showing phoneme → lexical semantics → syntax.
- **Right**: compare to children input — although the developmental *order* is qualitatively reminiscent of child language acquisition, the *data efficiency* is radically different: the models require **2 to 4 orders of magnitude more data** for these representational structures to emerge.

Discussions

Critiques?

Critiques?

- Can we reject the claim that: **the probe itself is too powerful to lead to such multi-level decodable linguistic structures?**

Critiques?

- Can we reject the claim that: **the probe itself is too powerful to lead to such multi-level decodable linguistic structures?**

  ✦ This critique is common in the probing classifier literature

Critiques?

- Can we reject the claim that: **the probe itself is too powerful to lead to such multi-level decodable linguistic structures?**

  ✦ This critique is common in the probing classifier literature

  ✦ But it is weaker here than when criticizing nonlinear (e.g. MLP) probes, since all the learned parameter here is a linear projector $\mathbf{B}$. In this sense, it is not a high-capacity classifier that maximizes accuracy.

Critiques?

- Can we reject the claim that: **the probe itself is too powerful to lead to such multi-level decodable linguistic structures?**

  ✦ This critique is common in the probing classifier literature

  ✦ But it is weaker here than when criticizing nonlinear (e.g. MLP) probes, since all the learned parameter here is a linear projector $\mathbf{B}$. In this sense, it is not a high-capacity classifier that maximizes accuracy.

  ✦ However, it is, after all, a supervised learning task, so some *representation learning* that could reshape the space in the desired way is inevitable.

Critiques?

- Can we reject the claim that: **the probe itself is too powerful to lead to such multi-level decodable linguistic structures?**

  ✦ This critique is common in the probing classifier literature

  ✦ But it is weaker here than when criticizing nonlinear (e.g. MLP) probes, since all the learned parameter here is a linear projector $\mathbf{B}$. In this sense, it is not a high-capacity classifier that maximizes accuracy.

  ✦ However, it is, after all, a supervised learning task, so some *representation learning* that could reshape the space in the desired way is inevitable.

  ✦ Think about what a linear operator could do: rotate the space, rescale dimensions, collapse nuisance variation, separate directions that are less important; amplifying structures that are originally entangled / distributed.

  ✦ …

Does structural probe target syntax only?

Evidence from Jabberwocky

Do Syntactic Probes Probe Syntax?
Experiments with Jabberwocky Probing

Rowan Hall Maudslay[1,2] **Ryan Cotterell**[1,2]
[1]University of Cambridge [2]ETH Zürich
rh635@cam.ac.uk ryan.cotterell@inf.ethz.ch

Does structural probe target syntax only?

Evidence from Jabberwocky

Do Syntactic Probes Probe Syntax?
Experiments with Jabberwocky Probing

Rowan Hall Maudslay[1,2] **Ryan Cotterell**[1,2]
[1]University of Cambridge [2]ETH Zürich
rh635@cam.ac.uk ryan.cotterell@inf.ethz.ch

A The Jabberwocky

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.

"Beware the Jabberwock, my son!
The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
The frumious Bandersnatch!"

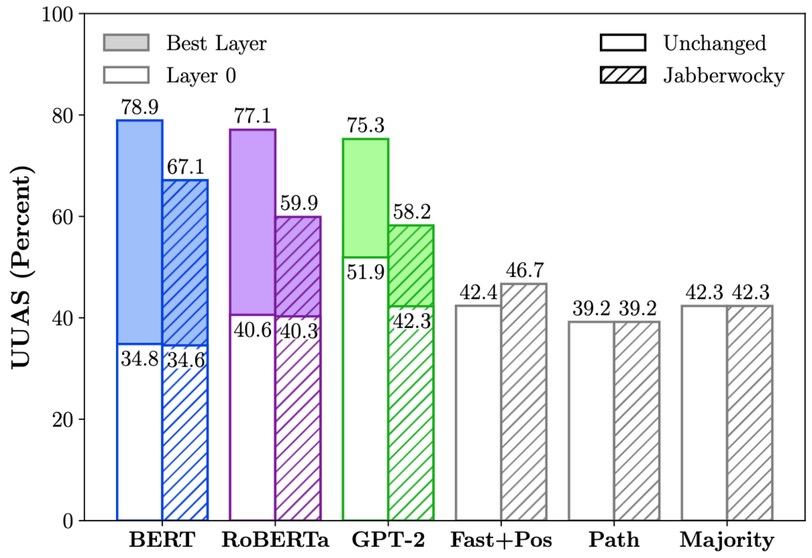
He took his vorpal sword in hand:
Long time the manxome foe he sought—
So rested he by the Tumtum tree,
And stood awhile in thought.

Does structural probe target syntax only?

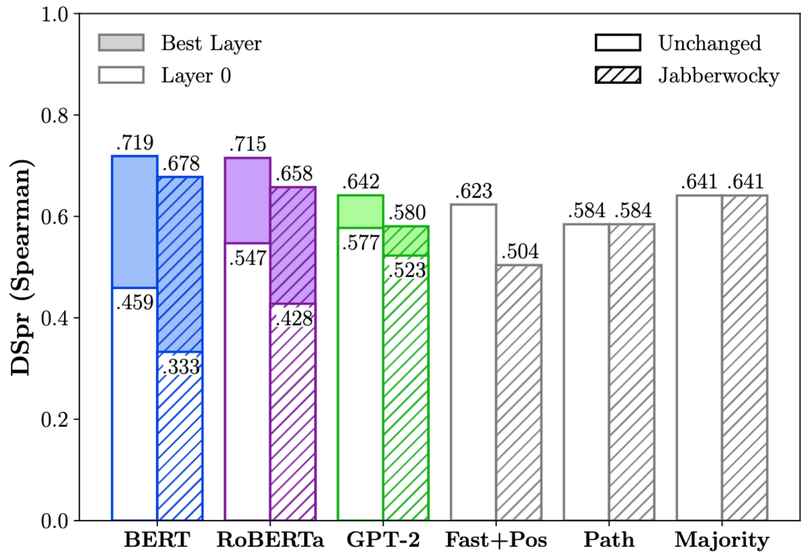
Evidence from Jabberwocky

Do Syntactic Probes Probe Syntax? Experiments with Jabberwocky Probing

ın Hall Maudslay[1,2] **Ryan Cotterell**[1,2]
niversity of Cambridge [2]ETH Zürich
.ac.uk ryan.cotterell@inf.ethz.ch



(a) UUAS results from the Perceptron Probe



(b) DSpr results from the Structural Probe

Figure 3: How the models fair when the probes are evaluated on unchanged sentences vs. the Jabberwocky.

A The Jabberwocky

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.

"Beware the Jabberwock, my son!
The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
The frumious Bandersnatch!"

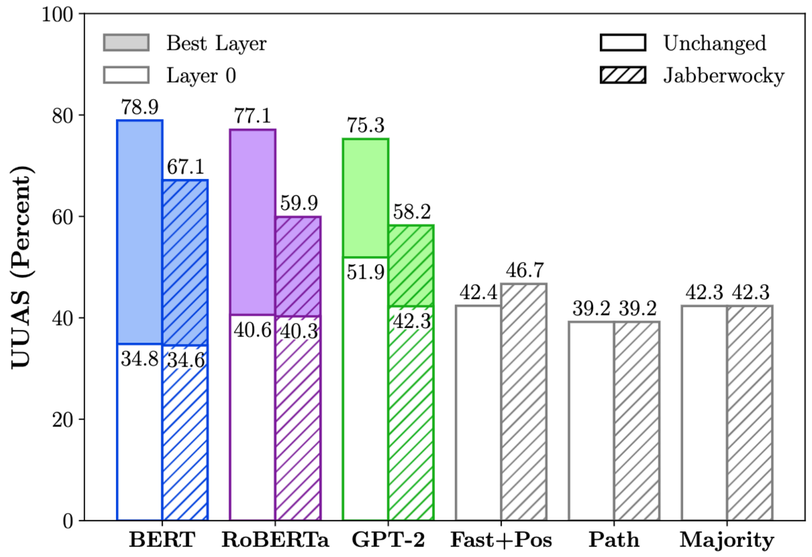
He took his vorpal sword in hand:
Long time the manxome foe he sought—
So rested he by the Tumtum tree,
And stood awhile in thought.

Does structural probe target syntax only?

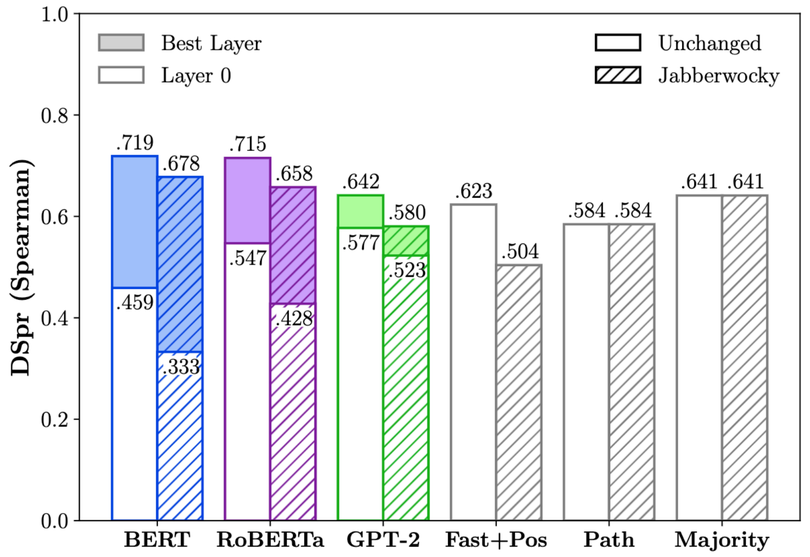
Evidence from Jabberwocky

Do Syntactic Probes Probe Syntax? Experiments with Jabberwocky Probing

ın Hall Maudslay[1,2] **Ryan Cotterell**[1,2]
niversity of Cambridge [2]ETH Zürich
.ac.uk ryan.cotterell@inf.ethz.ch



(a) UUAS results from the Perceptron Probe



(b) DSpr results from the Structural Probe

Figure 3: How the models fair when the probes are evaluated on unchanged sentences vs. the Jabberwocky.

Takeaway:
structural probe does not propertly isolate syntax…

A The Jabberwocky

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.

"Beware the Jabberwock, my son!
The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
The frumious Bandersnatch!"

He took his vorpal sword in hand:
Long time the manxome foe he sought—
So rested he by the Tumtum tree,
And stood awhile in thought.

CAUTION: how strong is the claim here?

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):
  - Decodability: whether a feature exists in the hidden states;

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):
  - Decodability: whether a feature exists in the hidden states;
  - **Linear Decodability: a linear map is enough to decode the information —i.e., in an non-basis-aligned subspace;**

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):
  - Decodability: whether a feature exists in the hidden states;
  - **Linear Decodability: a linear map is enough to decode the information —i.e., in an non-basis-aligned subspace;**
  - Native Representation: the model itself intrinsically uses that representation — i.e., in a basis-aligned subspace;

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):
  - Decodability: whether a feature exists in the hidden states;
  - **Linear Decodability: a linear map is enough to decode the information —i.e., in an non-basis-aligned subspace;**
  - Native Representation: the model itself intrinsically uses that representation — i.e., in a basis-aligned subspace;
- Is polar coordinate really special here?

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):
  - Decodability: whether a feature exists in the hidden states;
  - **Linear Decodability: a linear map is enough to decode the information —i.e., in an non-basis-aligned subspace;**
  - Native Representation: the model itself intrinsically uses that representation — i.e., in a basis-aligned subspace;
- Is polar coordinate really special here?
  - Strong Claim: LLMs use a polar coordinate system internally.

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):
  - Decodability: whether a feature exists in the hidden states;
  - **Linear Decodability: a linear map is enough to decode the information —i.e., in an non-basis-aligned subspace;**
  - Native Representation: the model itself intrinsically uses that representation — i.e., in a basis-aligned subspace;
- Is polar coordinate really special here?
  - Strong Claim: LLMs use a polar coordinate system internally.
  - **Weaker Claim: there *exists* a linear subspace in whcih syntax etc. is conveniently described by radial and angular quantities.**

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):
  - Decodability: whether a feature exists in the hidden states;
  - **Linear Decodability: a linear map is enough to decode the information —i.e., in an non-basis-aligned subspace;**
  - Native Representation: the model itself intrinsically uses that representation — i.e., in a basis-aligned subspace;
- Is polar coordinate really special here?
  - Strong Claim: LLMs use a polar coordinate system internally.
  - **Weaker Claim: there *exists* a linear subspace in whcih syntax etc. is conveniently described by radial and angular quantities.**
    - But it is still elegant that the human-interpretable quantities naturally align with geometrical properties (hierarchical depth ↔ magnitude, dependency type ↔ orientation, and head direction ↔ orientation sign);

CAUTION: how strong is the claim here?

- Levels of claims (weak to strong):
  - Decodability: whether a feature exists in the hidden states;
  - **Linear Decodability: a linear map is enough to decode the information —i.e., in an non-basis-aligned subspace;**
  - Native Representation: the model itself intrinsically uses that representation — i.e., in a basis-aligned subspace;
- Is polar coordinate really special here?
  - Strong Claim: LLMs use a polar coordinate system internally.
  - **Weaker Claim: there *exists* a linear subspace in whcih syntax etc. is conveniently described by radial and angular quantities.**
    - But it is still elegant that the human-interpretable quantities naturally align with geometrical properties (hierarchical depth ↔ magnitude, dependency type ↔ orientation, and head direction ↔ orientation sign);
    - In principle, Cartesian coordinate can represent exactly the same information, thus it is possible to find an alternative $\mathbf{B}'$ that represents it in Cartesian….

Zooming Even Out

Zooming Even Out

We gained intuitions on how to think about the activation space, enabling us to:

Zooming Even Out

We gained intuitions on how to think about the activation space, enabling us to:

- For people who are interested in **superposition / polysemanticity** of the residual stream: this linear probe is helpful for discovering more properites.

Zooming Even Out

We gained intuitions on how to think about the activation space, enabling us to:

- For people who are interested in **superposition / polysemanticity** of the residual stream: this linear probe is helpful for discovering more properites.
- For people who love **TPR**: more testable hypotheses here!

Zooming Even Out

We gained intuitions on how to think about the activation space, enabling us to:

- For people who are interested in **superposition / polysemanticity** of the residual stream: this linear probe is helpful for discovering more properites.
- For people who love **TPR**: more testable hypotheses here!
- For people who love **acquisition**: curious to see how those linear subspaces are shaped and interact (e.g., bootstrapping across multiple linguistic levels) develomentally;

Zooming Even Out

We gained intuitions on how to think about the activation space, enabling us to:

- For people who are interested in **superposition / polysemanticity** of the residual stream: this linear probe is helpful for discovering more properites.
- For people who love **TPR**: more testable hypotheses here!
- For people who love **acquisition**: curious to see how those linear subspaces are shaped and interact (e.g., bootstrapping across multiple linguistic levels) develomentally;
- For people who love **dynamical systems / processing**: think about how grammatical information are accessed in those more targeted subspaces;

Zooming Even Out

We gained intuitions on how to think about the activation space, enabling us to:

- For people who are interested in **superposition / polysemanticity** of the residual stream: this linear probe is helpful for discovering more properites.
- For people who love **TPR**: more testable hypotheses here!
- For people who love **acquisition**: curious to see how those linear subspaces are shaped and interact (e.g., bootstrapping across multiple linguistic levels) develomentally;
- For people who love **dynamical systems / processing**: think about how grammatical information are accessed in those more targeted subspaces;
- For **theoretical syntacticians**: there are work probing formalisms / more complicated syntactic representations (e.g., empty categories) beyong dependency!

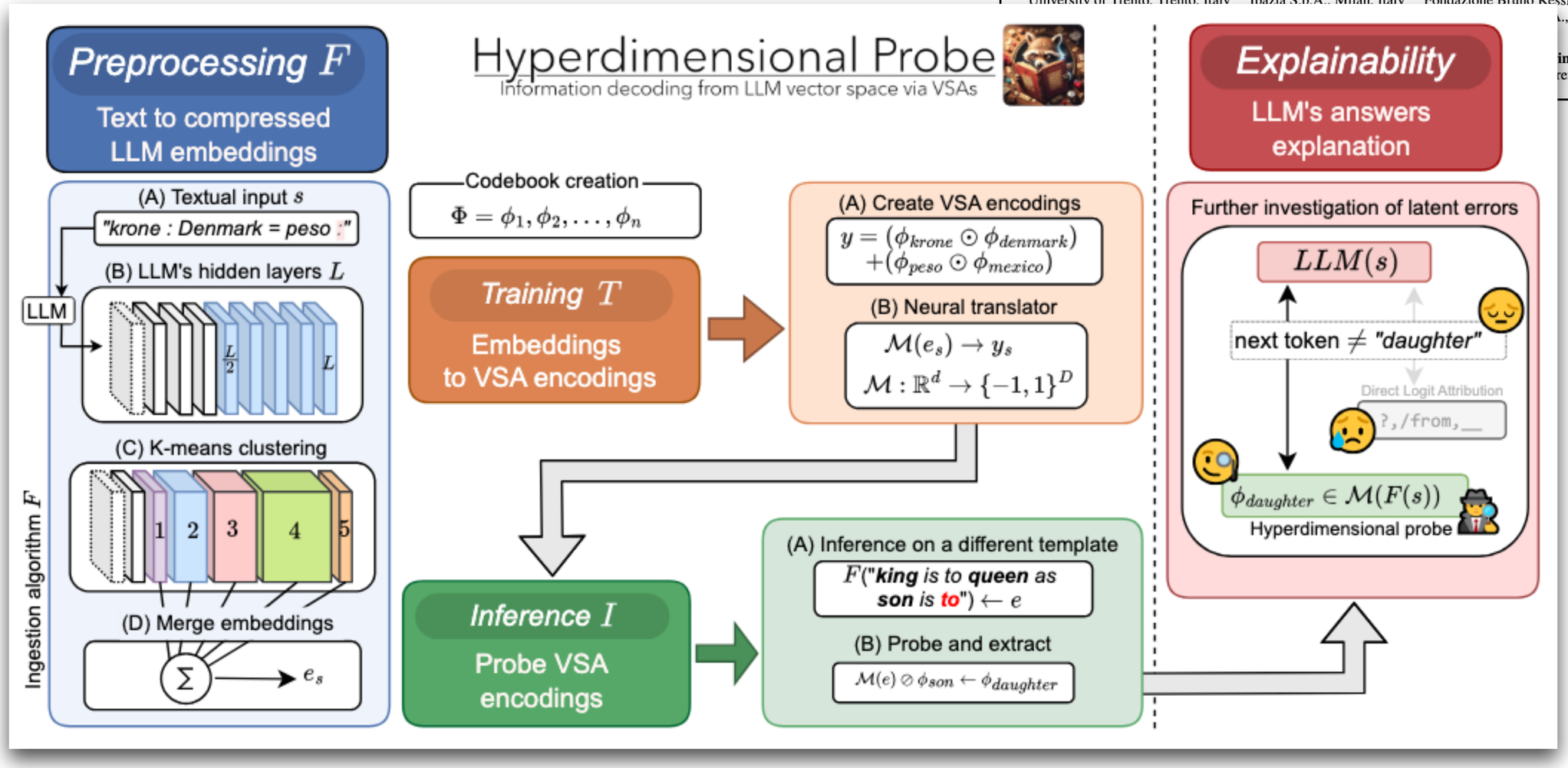
Further Recommended Reading

Hyperdimensional Probe

HYPERDIMENSIONAL PROBE: DECODING LLM REPRESENTATIONS VIA VECTOR SYMBOLIC ARCHITECTURES

A PREPRINT

Marco Bronzini* University of Trento, Trento, Italy
Carlo Nicolini Ipazia S.p.A., Milan, Italy
Bruno Lepri Fondazione Bruno Kessler (FBK), Trento, Italy
., Milan, Italy
ini
rento, Italy



Thank you!