# Correlating neural and symbolic representations of language

Grzegorz Chrupała
Afra Alishahi

Tilburg University

# Beyond diagnostic classifiers

- Diagnostic classifiers/regressors
  - non-trivial to apply to structures such as syntax trees
- Workarounds
  - Tree-depth, topmost constituent labels (Conneau & Kiela 2018)
  - Edge probing (Tenney et al. 2019)
- Our solution: **RSA + tree kernels**

# RSA

Kriegeskorte, N., Mur, M., & Bandettini, P. A. (2008). Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2, 4.

# Representational similarity: an example

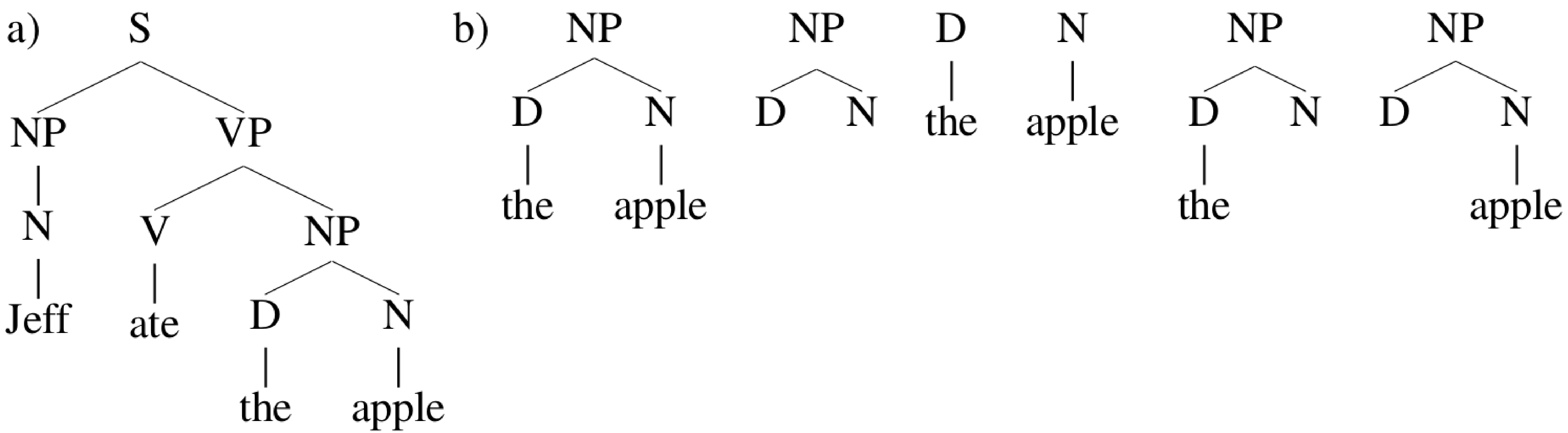
| Utt 1 | Utt 2 | Sim 1 | Sim 2 |
| --- | --- | --- | --- |
| A slice of pizza | A bowl of salad | 7.0 | 6.2 |
| Two dogs run | A kitty running | 8.0 | 9.0 |
| A yellow and white bird | A kitty running | 3.0 | 4.5 |

Correlation between similarity 1 and similarity 2

# Structured Spaces

- RSA is applicable whenever we can define a similarity/distance metric WITHIN spaces **A** and **B**.
- Crucially, no need for a metric BETWEEN **A** and **B**
- **A** can be a vector space, while **B** can be a space of **strings/trees/graphs**.

# Tree Kernels



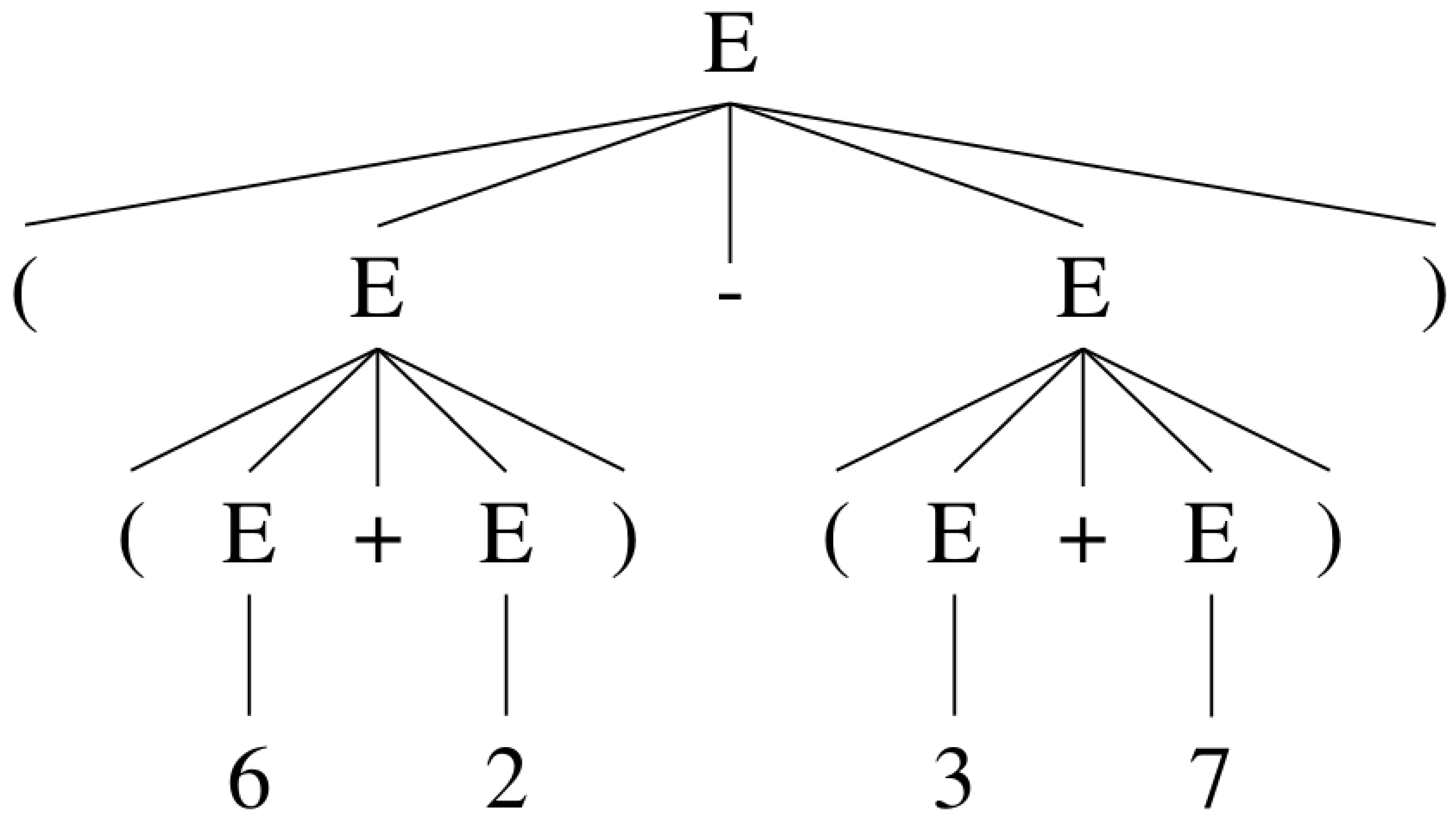

Collins, M., & Duffy, N. (2002). Convolution kernels for natural language.
In Advances in neural information processing systems (pp. 625-632).

# Simple synthetic language

- Well understood syntax and semantics
- Possible to fully learn by RNN
- Validate RSA approach

Similar to Hupkes et al (2018).

# Arithmetic expressions




((6+2)-(3+7))

# Syntax and semantics

| Syntax | Meaning |
|---|---|
| $E \to (\ E_1 + E_2\ )$ | $[E] = [E_1] + [E_2] \bmod 10$ |
| $E \to (\ E_1 - E_2\ )$ | $[E] = [E_1] - [E_2] \bmod 10$ |
| $E \to 0$ | $[E] = 0$ |
| $\vdots$ | $\vdots$ |
| $E \to 9$ | $[E] = 9$ |

Example:

```
((6+2)-(3+7))
(8 - 0)
8
```

# Representations and (dis)similarity functions

- This part is confusing
-
- Semantic
  - Expression value | absolute difference
- Syntactic (simple)
  - Expression depth | absolute difference
- Syntactic (full)
  - Expression tree | tree kernel

# Neural (LSTM) models

Identical encoder architecture

- Random (untrained)
- Semantic evaluation

  `(6 - (2 + 3)) → 1`

- Tree depth

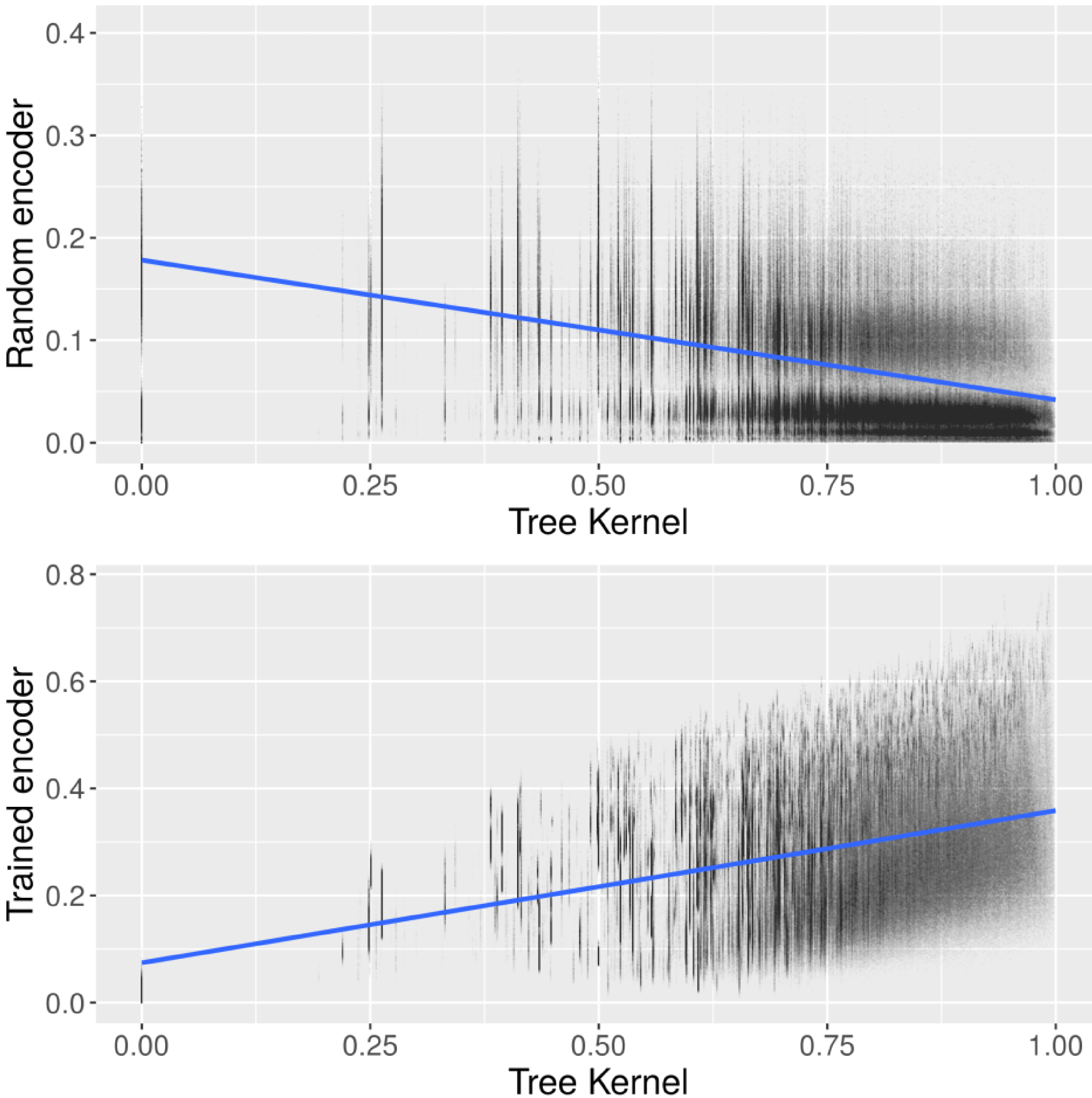
  `(6 - (2 + 3)) → 2`

- Infix-to-prefix

  `(6-(2+3)) → (-6(+23))`

# Results

| | | Diagnostic | | RSA | | |
|---|---|---|---|---|---|---|
| Encoder | Loss | Value | Depth | Value | Depth | TK |
| RANDOM | | -0.01 | 0.79 | 0.01 | 0.23 | -0.39 |
| SEMANTIC EVALUATION | 0.07 | **0.97** | 0.69 | **0.62** | 0.05 | -0.01 |
| TREE DEPTH | 0.00 | -0.01 | **1.00** | 0.01 | **0.72** | -0.20 |
| INFIX-TO-PREFIX | 0.00 | 0.01 | 0.96 | -0.00 | 0.64 | **0.39** |



Random encoder
0.4
0.3
0.2
0.1
0.0
0.00
0.25
0.50
0.75
1.00
Tree Kernel
Trained encoder
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
Tree Kernel

# $\mathrm{RSA}_{\mathrm{REGRESS}}$

- Need an example here
- 
- Given reference expressions *R*
- For each expression of interest, embed in vector space by measuring similarities to R
  - Source similarity $\sigma_k$ – cosine in neural represention space
  - Target similarity $\sigma_l$ – e.g. tree kernel
- Predict vectors embedded via $\sigma_l$ from vectors embedded via $\sigma_k$

$$\widehat{\mathbf{B}, \mathbf{a}} = \arg\min_{\mathbf{B},\mathbf{a}} \mathrm{MSE}(\mathbf{B}\sigma_k(X, R) + \mathbf{a}, \sigma_l(X, R))$$

# Properties of $RSA_{REGRESS}$

- Like **diagnostic model**, focus on **prediction**, not correlation of complete representation space
- Like **RSA**, easy access to **structured symbolic** representations via **similarity** embedding

# Results

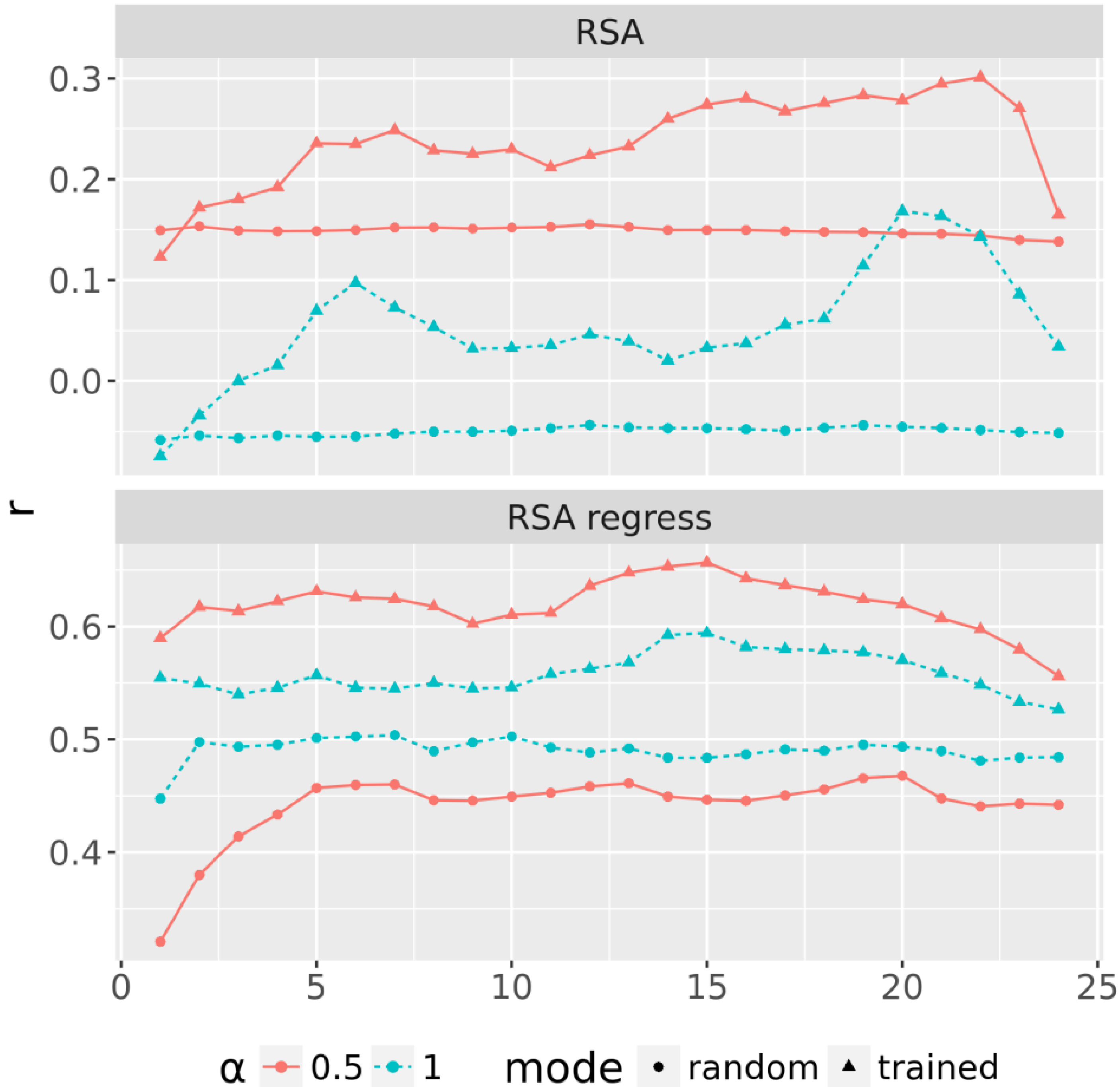
| | RSA | | | $RSA_{REGRESS}$ | | |
|---|---|---|---|---|---|---|
| Encoder | Value | Depth | TK | Value | Depth | TK |
| RANDOM | 0.01 | 0.23 | -0.39 | -0.02 | 0.56 | 0.67 |
| SEMANTIC EVALU. | **0.62** | 0.05 | -0.01 | **0.96** | 0.54 | 0.62 |
| TREE DEPTH | 0.01 | **0.72** | -0.20 | -0.02 | **0.97** | 0.82 |
| INFIX-TO-PREFIX | -0.00 | 0.64 | **0.39** | 0.02 | 0.88 | **0.89** |

# Results on English

- **Infersent** (Conneau 2017)
  - trained on NLI
- **BERT** (Devlin et al. 2018)
  - trained on cloze and next-sentence classification
- Random versions of these

| Encoder | Train | $\alpha$ | RSA | $\text{RSA}_{\text{REGRESS}}$ |
|---|---|---|---|---|
| BoW | | 0.5 | 0.19 | 0.44 |
| Infersent | $-$ | 0.5 | 0.23 | 0.44 |
| BERT last | $-$ | 0.5 | 0.14 | 0.44 |
| BERT best | $-$ | 0.5 | 0.16 | 0.47 |
| Infersent | $+$ | 0.5 | 0.27 | **0.67** |
| BERT last | $+$ | 0.5 | 0.17 | 0.56 |
| BERT best | $+$ | 0.5 | **0.30** | 0.66 |
| BoW | | 1.0 | 0.01 | 0.39 |
| Infersent | $-$ | 1.0 | 0.01 | 0.47 |
| BERT last | $-$ | 1.0 | -0.05 | 0.48 |
| BERT best | $-$ | 1.0 | -0.04 | 0.50 |
| Infersent | $+$ | 1.0 | 0.09 | 0.57 |
| BERT last | $+$ | 1.0 | 0.03 | 0.53 |
| BERT best | $+$ | 1.0 | **0.17** | **0.59** |

# BERT layers


RSA
RSA regress
r
0.3
0.2
0.1
0.0
0.6
0.5
0.4
0
5
10
15
20
25
α
0.5
1
mode
random
trained

- Paper

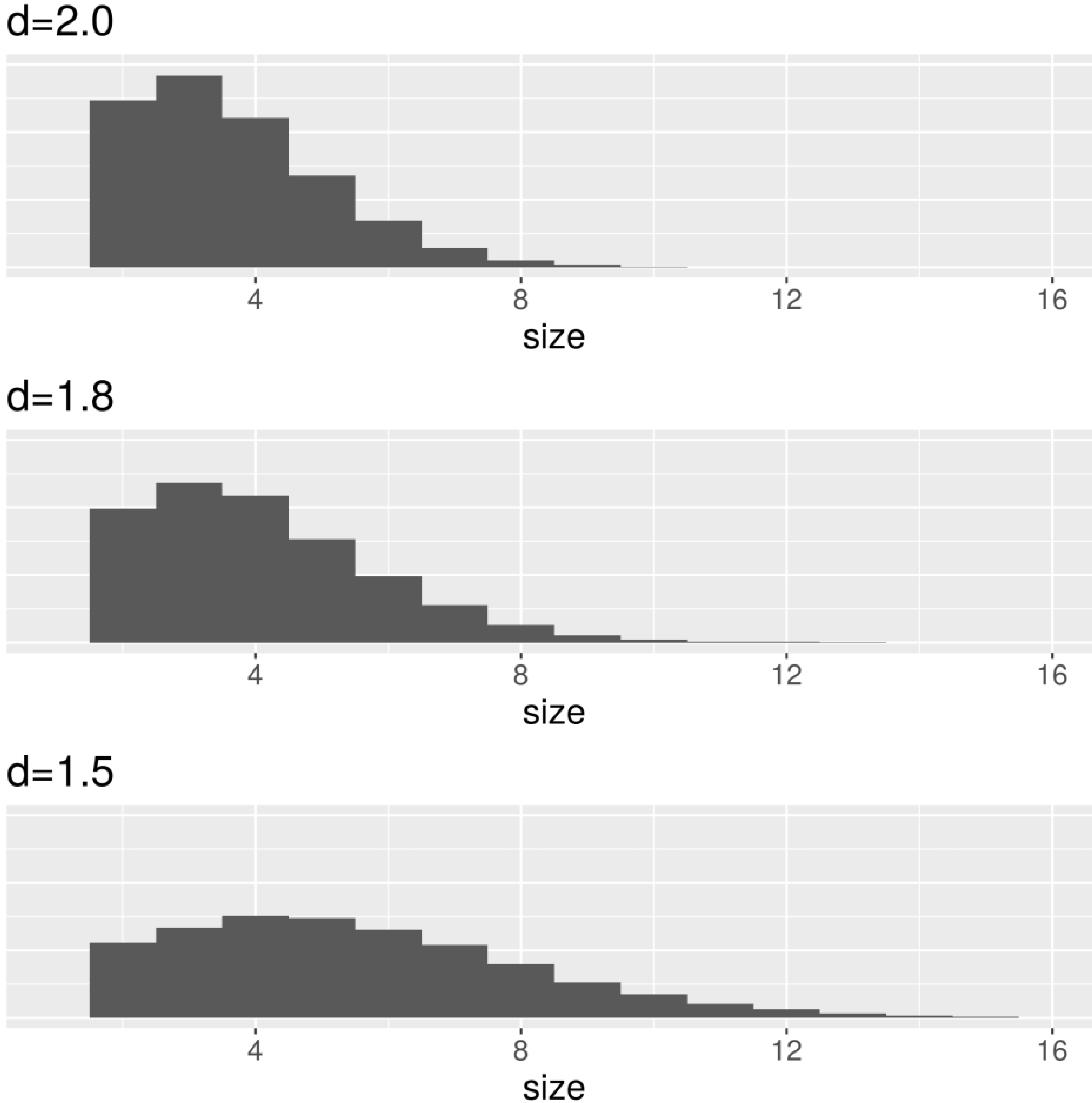
  **`openreview.net/forum?id=ryx35Ehi84`**
- Code

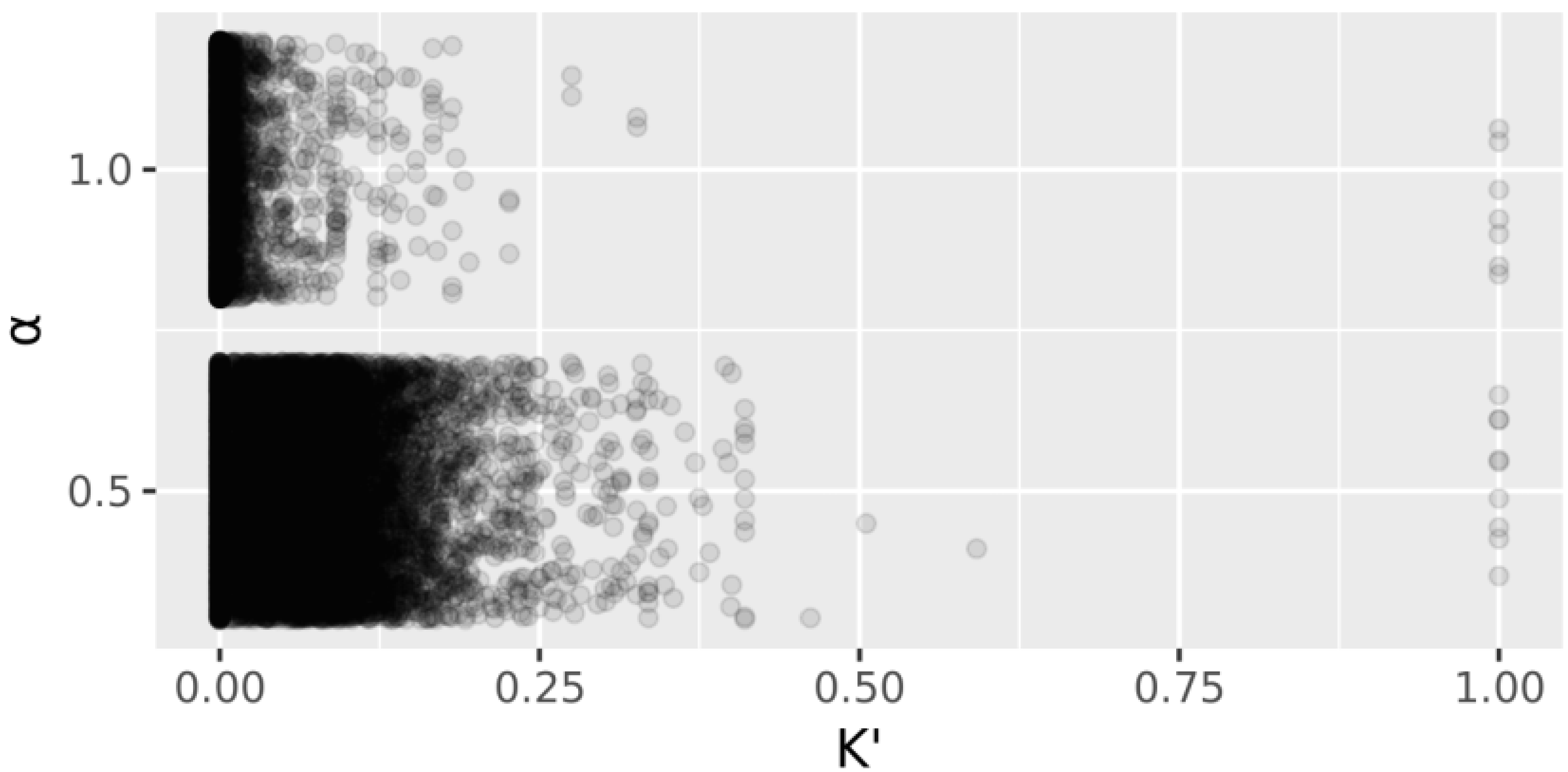
  coming soon

# Extras

**Algorithm 1** Recursive function for generating an expression of language $L$.

```
function GENERATE(p, d)
    branch ~ BERNOULLI(p)
    if branch then                                  ▷ Rules 1–2
        E_1 ← GENERATE(p/d, d)
        E_2 ← GENERATE(p/d, d)
        op ~ UNIFORM([+, −])
        return (E_1 op E_2)
    else                                            ▷ Rules 3–13
        digit ~ UNIFORM([0, . . . , 9])
        return digit
    end if
end function
```


d=2.0
4
8
12
16
size
d=1.8
4
8
12
16
size
d=1.5
4
8
12
16
size



1.0
0.5
α
0.00
0.25
0.50
0.75
1.00
K'