A working man's merge

Evidence for restricted distributional composition in phrase semantics

Victor Zimmermann[1,2]
[1]Institute of Linguistics, Leipzig University
[2]Neuromorph Information Processing, ibid.

73. Stuts, Frankfurt (Main)
Saturday, 27th May 2023

- *RQ 1:* How are phrases constructed in a high-dimensional semantic space?

- *RQ 1:* How are phrases constructed in a high-dimensional semantic space?
- Competing representations:
    - Complex trained weight-distributions at intermediary layers between word embeddings and sentence task output (master thesis).
    - Naïve summation of constituent representations.

- *RQ 1:* How are phrases constructed in a high-dimensional semantic space?
- Competing representations:
  - Complex trained weight-distributions at intermediary layers between word embeddings and sentence task output (master thesis).
  - Naïve summation of constituent representations.
- *RQ 2:* Is there a (simple) composition function and if yes, how many?

- *RQ 1:* How are phrases constructed in a high-dimensional semantic space?
- Competing representations:
  - Complex trained weight-distributions at intermediary layers between word embeddings and sentence task output (master thesis).
  - Naïve summation of constituent representations.
- *RQ 2:* Is there a (simple) composition function and if yes, how many?
- Mediating approach: learning to predict selection of simple composition functions instead of composition output.

What I want to do in this talk.

- Talk about representation and semantics with you.
- Give a sketch of how to do theoretical linguistics computationally, despite everything.
- Share the pain of writing a thesis.

Section 1
Composition, distribution & representation

73. Stuts, Frankfurt (Main)
Saturday, 27^{th} May 2023

What is (semantic) composition?

What is (semantic) composition?

- Semantic composition mirrors syntactic structure.

What is (semantic) composition?

- Semantic composition mirrors syntactic structure.
- Distributional semantics approximates word meaning from word distributions.
    - Efficient vector encoding of coöccurence matrices (like *.jpeg*, but for word contexts).

What is (semantic) composition?

- Semantic composition mirrors syntactic structure.
- Distributional semantics approximates word meaning from word distributions.
    - Efficient vector encoding of coöccurence matrices (like *.jpeg*, but for word contexts).
- No natural mapping relation from distributional semantics to formal semantics.pause

Why use approximations?

[1]Box, G. E. P. (1976) 'Science and statistics'.

Why use approximations?

- All models are wrong.[1]

[1] Box, G. E. P. (1976) 'Science and statistics'.

Why use approximations?

- All models are wrong.[1]

[1] Box, G. E. P. (1976) 'Science and statistics'.

Why use approximations?

- All models are wrong.[1]
  - We have no idea how semantics *actually* works.

[1] Box, G. E. P. (1976) 'Science and statistics'.

Why use approximations?

- All models are wrong.[1]
    - We have no idea how semantics *actually* works.
    - The brain does not run on matrix multiplication.

[1] Box, G. E. P. (1976) 'Science and statistics'.

Why use approximations?

- All models are wrong.[1]
  - We have no idea how semantics *actually* works.
  - The brain does not run on matrix multiplication.

[1] Box, G. E. P. (1976) 'Science and statistics'.

Why use approximations?

- All models are wrong.[1]
  - We have no idea how semantics *actually* works.
  - The brain does not run on matrix multiplication.
  - But: statistical models eventually approach the correct input-output mapping.

[1]Box, G. E. P. (1976) 'Science and statistics'.

Why use approximations?

- All models are wrong.[1]
    - We have no idea how semantics *actually* works.
    - The brain does not run on matrix multiplication.
    - But: statistical models eventually approach the correct input-output mapping.
- Only feature-based semantics approach that can be induced from large-scale data (see also the Generative Lexicon).

[1] Box, G. E. P. (1976) 'Science and statistics'.

Why use approximations?

- All models are wrong.[1]
  - We have no idea how semantics *actually* works.
  - The brain does not run on matrix multiplication.
  - But: statistical models eventually approach the correct input-output mapping.
- Only feature-based semantics approach that can be induced from large-scale data (see also the Generative Lexicon).
- Interfaces nicely with other vague, hand-wavy models like neural networks.

[1] Box, G. E. P. (1976) 'Science and statistics'.

Assumptions for this approach:

Assumptions for this approach:

- Word embeddings are close enough to featural descriptions of lexical items to be useful for compositional semantics.

Assumptions for this approach:

- Word embeddings are close enough to featural descriptions of lexical items to be useful for compositional semantics.
- Representations of phrases are close enough to representations of lexical items to be approximated in the same vector space.

Assumptions for this approach:

- Word embeddings are close enough to featural descriptions of lexical items to be useful for compositional semantics.
- Representations of phrases are close enough to representations of lexical items to be approximated in the same vector space.
- There is a function or set of functions that describe the mapping from constituent representations to phrase representations.

Section 2

Compositional phrase embeddings from latent Tree-LSTM representations

73. Stuts, Frankfurt (Main)
Saturday, 27th May 2023

Motivation:

Motivation:

- Many uses for phrasal representations in alignment tasks like coreference and text reuse detection.

Motivation:

- Many uses for phrasal representations in alignment tasks like coreference and text reuse detection.
- Computational phrase semantics understudied since end-to-end approaches lack syntax.

Motivation:

- Many uses for phrasal representations in alignment tasks like coreference and text reuse detection.
- Computational phrase semantics understudied since end-to-end approaches lack syntax.

Approach:

Motivation:

- Many uses for phrasal representations in alignment tasks like coreference and text reuse detection.
- Computational phrase semantics understudied since end-to-end approaches lack syntax.

Approach:

- Use tree-structured recurrent neural network to force latent phrasal representations.

Motivation:

- Many uses for phrasal representations in alignment tasks like coreference and text reuse detection.
- Computational phrase semantics understudied since end-to-end approaches lack syntax.

Approach:

- Use tree-structured recurrent neural network to force latent phrasal representations.
- Probe phrase representations for semantic properties.

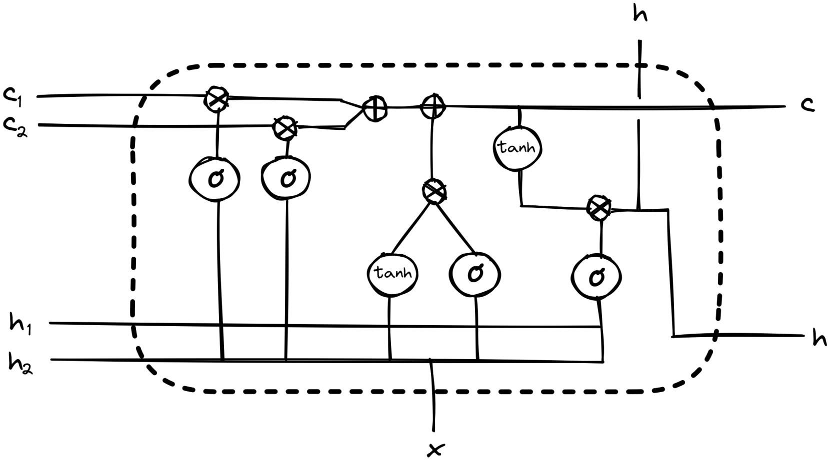
Motivation:

- Many uses for phrasal representations in alignment tasks like coreference and text reuse detection.
- Computational phrase semantics understudied since end-to-end approaches lack syntax.

Approach:

- Use tree-structured recurrent neural network to force latent phrasal representations.
- Probe phrase representations for semantic properties.
- Use phrase embeddings in downstream tasks.

Tree-LSTM



Sister node prediction (unsupervised):

[2]Ellie Pavlick, et al. (2015) 'PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification'

Sister node prediction (unsupervised):

- Predict embedding of left child node from right child node.

[2]Ellie Pavlick, et al. (2015) 'PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification'

Sister node prediction (unsupervised):

- Predict embedding of left child node from right child node.

Paraphrase classification (supervised):

[2] Ellie Pavlick, et al. (2015) 'PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification'

Sister node prediction (unsupervised):

- Predict embedding of left child node from right child node.

Paraphrase classification (supervised):

- Phrasal data from the Paraphrase Database (PPDB)[2].

[2] Ellie Pavlick, et al. (2015) 'PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification'

Preprocessing & model parameters

- Data set preparation: Splits, label extraction.

Preprocessing & model parameters

- Data set preparation: Splits, label extraction.
- CCG parser [1].

- Data set preparation: Splits, label extraction.
- CCG parser [1].
- NLTK tree parser & Chomsky Normal Form.

- Data set preparation: Splits, label extraction.
- CCG parser [1].
- NLTK tree parser & Chomsky Normal Form.
- Tree-linearisation.

- Data set preparation: Splits, label extraction.
- CCG parser [1].
- NLTK tree parser & Chomsky Normal Form.
- Tree-linearisation.
- Word embeddings [2], 300d.

- Data set preparation: Splits, label extraction.
- CCG parser [1].
- NLTK tree parser & Chomsky Normal Form.
- Tree-linearisation.
- Word embeddings [2], 300d.

- Sister node regressors: σ
- Paraphrase classifier: *ReLU*
- Filter classifier: *tanh*

Section 3

Limiting unlimited composition.

What if we don't need to learn the composition function?

What if we don't need to learn the composition function?

- Hard: find n-dimensional, complex mapping between two input and one output vector for a given task.

What if we don't need to learn the composition function?

- Hard: find n-dimensional, complex mapping between two input and one output vector for a given task.
- Maybe easier: select function from a list of simple functions to approximate complex mapping.

How to build an embedding engine that works smarter, not harder.

How to build an embedding engine that works smarter, not harder.

- Build model as before, with sweat, blood and computing hours.

How to build an embedding engine that works smarter, not harder.

- Build model as before, with sweat, blood and computing hours.
- Attach a cute little classifier to the cell input and output.

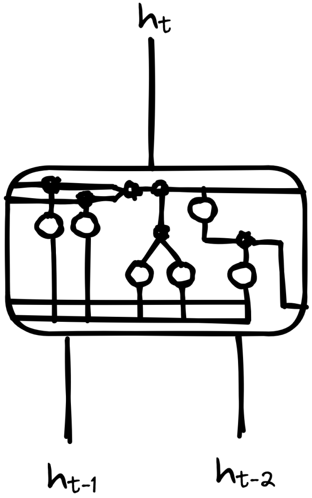
How to build an embedding engine that works smarter, not harder.

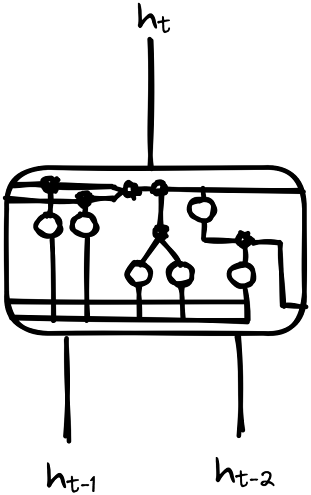
- Build model as before, with sweat, blood and computing hours.
- Attach a cute little classifier to the cell input and output.
- Rank functions by similarity to output of complex classifier.

How to build an embedding engine that works smarter, not harder.

- Build model as before, with sweat, blood and computing hours.
- Attach a cute little classifier to the cell input and output.
- Rank functions by similarity to output of complex classifier.
- Train classifier on function rankings.

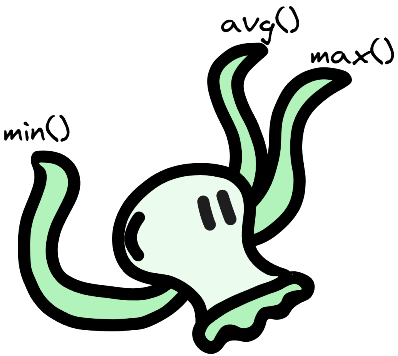
Parasitic composition (cont.)

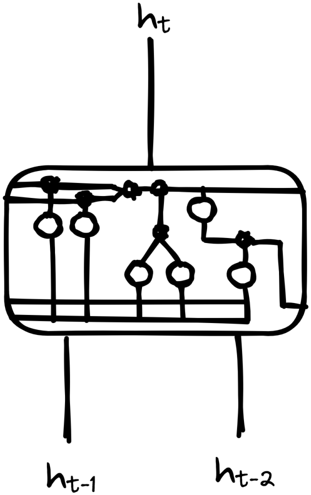




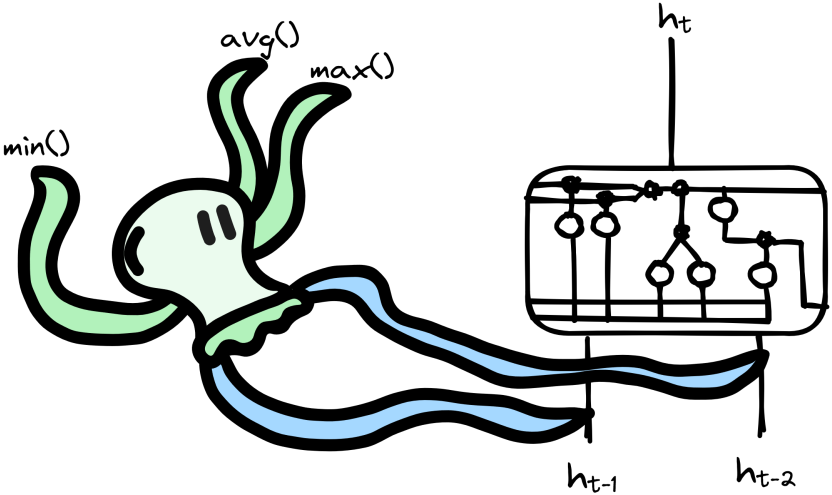
h_t
h_{t-1}
h_{t-2}

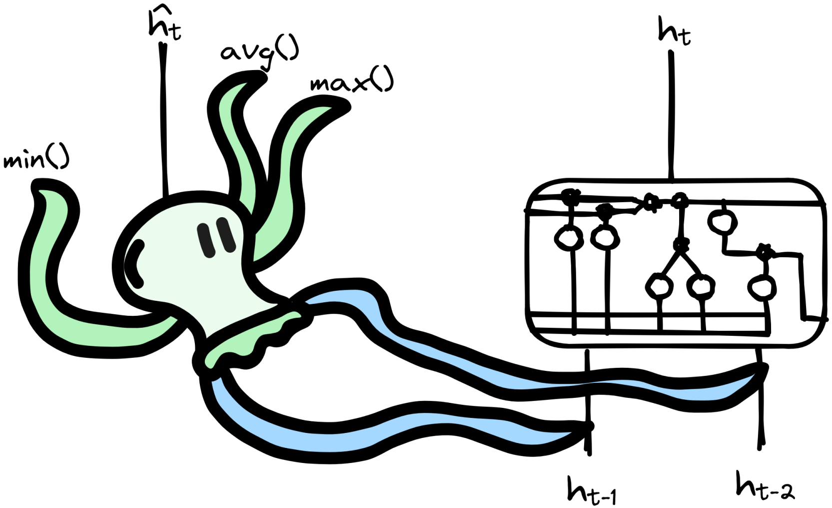
Parasitic composition (cont.)





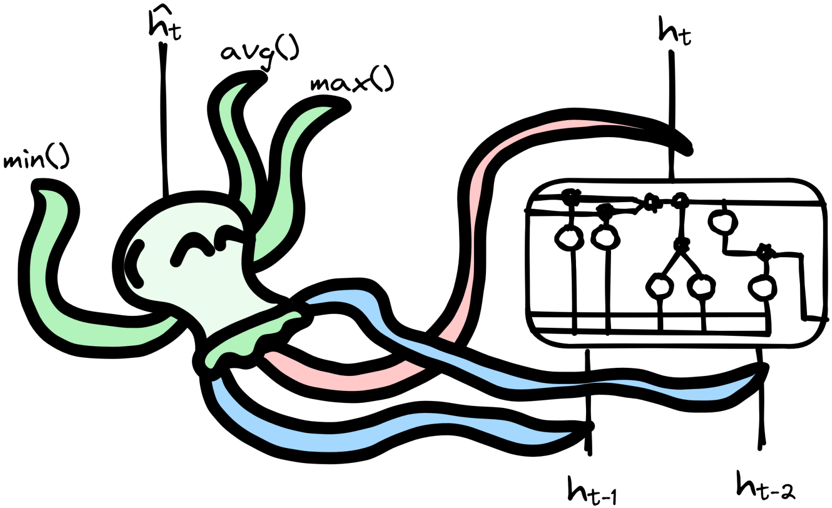
Parasitic composition (cont.)





$\hat{h}_t$
avg()
max()
min()
h_t
h_{t-1}
h_{t-2}

Parasitic composition (cont.)



Results

- Struggling with data handling: trying a lot of different parsers and datasets and I get different results.

- Struggling with data handling: trying a lot of different parsers and datasets and I get different results.
- Sister node prediction hard to evaluate.

Where I'm at.

- Struggling with data handling: trying a lot of different parsers and datasets and I get different results.
- Sister node prediction hard to evaluate.
- Not sure about motivation for parasite network, except "argmax hard".

Where I'm at.

- Struggling with data handling: trying a lot of different parsers and datasets and I get different results.
- Sister node prediction hard to evaluate.
- Not sure about motivation for parasite network, except "argmax hard".
- What am I even saying about linguistics?

- Struggling with data handling: trying a lot of different parsers and datasets and I get different results.
- Sister node prediction hard to evaluate.
- Not sure about motivation for parasite network, except “argmax hard”.
- What am I even saying about linguistics?
- Barely in control of the maths.