# Deciphering Compositionality
# Compositionality in Shift Operations on the Alphabet

UNIVERSITY OF AMSTERDAM

Berend Jansen & Tom Lotze

Artificial Intelligence - University of Amsterdam

## Introduction

***Compositionality***: *The meaning of an expression can be derived from the meaning of its constituents and syntactic rules to combine them [1].*

The notion of compositionality in language is essential for deeper language understanding and language modelling. This research aims to give more insight into compositionality in neural sequence-to-sequence models, by training and evaluating on a new artificial language.

## Approach

- Our compositional language allows shift operations on letters in the alphabet.
- Train two models (LSTM and Transformer) on the generated data and perform two compositionality tests: substitutivity & localism.

### Artificial language

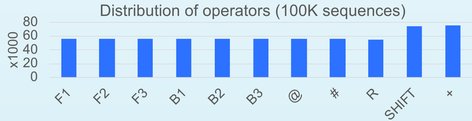
- Shift ciphering alphabetical strings
- Syntax by PCFG (left table)
- Semantics by interpretation rules (right table)

| Terminal Rules | |
|---|---|
| Y → | a \| b \| c \| … \| x \| y \| z |
| **Non-Terminals Rules** | |
| S → | Fu S \| SHIFT Y S \| S + S \| X |
| Fu → | F1 \| F2 \| F3 \| B1 \| B2 \| B3 \| R \| @ \| # |
| X → | X X \| Y |

| Interpretation Examples ( separator \|\| ) | |
|---|---|
| F1 a \|\| F1 z \|\| B1 a \|\| B1 z | b \|\| a \|\| z \|\| y |
| R a b c \|\| @ a b c \|\| # a b c | c b a \|\| c a b \|\| b c a |
| SHIFT **a** b f \|\| SHIFT **b** b f | c g \|\| d h |
| a b c + d e f | a b c d e f |
| F2 b c + SHIFT d a b | d e e f |

### Generated dataset

- Data generation using PCFG
- 100.000 (sequence, label)-pairs
- No duplicates
- Split: 85% / 5% / 10% (training, validation, test)
- On average 12.4 tokens (std: 12.9) per sample

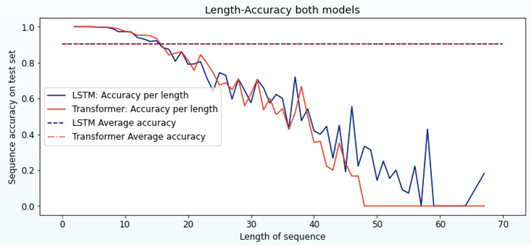



## Experiments & Results

### Models & General Performance

LSTM model (OpenNMT, 50k training steps)
- 2 encoder- & decoder layers of size 500
- Word embedding size 500

Transformer model (OpenNMT, 30k training steps)
- 4 encoder- & decoder layers of size 512
- Transformer feed forward of size 2048
- Word embedding size 512




Surprisingly, the LSTM outperforms the Transformer on longer sequences.

Common mistakes: often multiple "R", "@" or "#" operators, one of them is not applied. Possible reason is that these operators can cancel out.

### Substitutivity

Synonymity between "F1/F2/F3" and "SHIFT a/b/c"

Two synonymous datasets "F" vs. "SHIFT"
- Two versions: Clean (only "F" or "SHIFT") and complete (including other operators)
- PCFG generation with sequence validation

| Dataset | LSTM accuracy | LSTM consistency | Transformer accuracy | Transformer consistency |
|---|---|---|---|---|
| Clean "F" | 0.9784 | 0.7671 | 0.9633 | 0.9514 |
| Clean "SHIFT" | 0.7678 | | 0.9471 | |
| Complete "F" | 0.9415 | 0.9519 | 0.9332 | 0.9535 |
| Complete "SHIFT" | 0.9312 | | 0.9257 | |

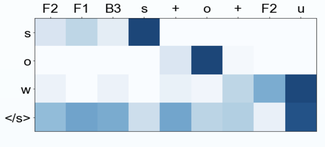
- Clean dataset: LSTM performs worse on SHIFT dataset
- Complete dataset: high consistency scores

## Localism

**Additive**: Force recursive processing using "+"
- Sequence format: "S + S + S"

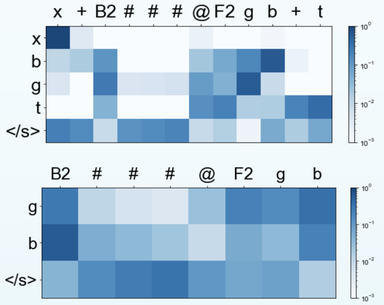
Process constituents separately and solve merged outcomes **vs**. process original sequence at once.

| Model | Original accuracy | Merged Accuracy | Consistency |
|---|---|---|---|
| LSTM | 0.9350 | 0.9583 | 0.9412 |
| Transformer | 0.9439 | 0.9715 | 0.9415 |




*Attention plot showing local processing*

- Both models perform slightly better if they are forced to process the input recursively
- High consistency: Model works relatively locally




*Attention plots for recursive localism. Top graph shows the original sequence (*wrong*), bottom one the middle constituent (*correct*).*

In the top plot we barely see attention for the "#", whereas in the correctly predicted bottom constituent there is some attention on the "#".

**Sequential**: Force the model to sequentially apply the operators on previous outcome.

| Model | Original accuracy | Sequential accuracy | Consistency |
|---|---|---|---|
| LSTM | 0.8605 | 0.976 | 0.8555 |
| Transformer | 0.8605 | 0.993 | 0.8625 |

There is a relatively large performance gap between the forced sequential approach and the general approach, so we conclude that the model is not fully sequential.

## Conclusion

- This research compares the capacity of an LSTM and Transformer model to exhibit compositional behaviour. On the **main task**, both model perform similarly. Surprisingly, the LSTM model scores slightly better on longer sequences.
- In terms of **substitutivity**, especially the Transformer is able to learn synonymous operator combinations, attested by the high consistency score of 95.2%. The LSTM on the other hand only scores 76.7%.
- Two forms of **localism** were tested: Local processing in additive operators is observed in both models. However, forced sequential processing of all other operators improves performance significantly, suggesting the models do not process every operator locally.
- **Attention plots** show that errors on complete sequences might occur because an operator is forgotten by the network. Forced recursive processing often solves this issue.

## Discussion

- The Transformer model is known to be sensitive to hyperparameters, further tuning might lead to better performance.
- More compositionality tests could be performed on this artificial language.
- Another interesting aspect would be to analyze the embedding space, to investigate whether the model learns the sequence of the alphabet.

[1] Partee, B. (1995). Lexical semantics and compositionality. An invitation to cognitive science: Language, 1:311–360.