# Abstractors and relational cross-attention: An inductive bias for explicit relational reasoning in Transformers

Awni Altabaa[1], Taylor W. Webb[2], Jonathan D. Cohen[3], John Lafferty[1]

[1]Yale University, [2]UCLA, [3]Princeton University

## High-level Goals

1. Understand why the Transformer architecture struggles in terms of sample efficiency when learning relational tasks.
2. Design mechanisms for explicit relational reasoning within the broader Transformer framework.

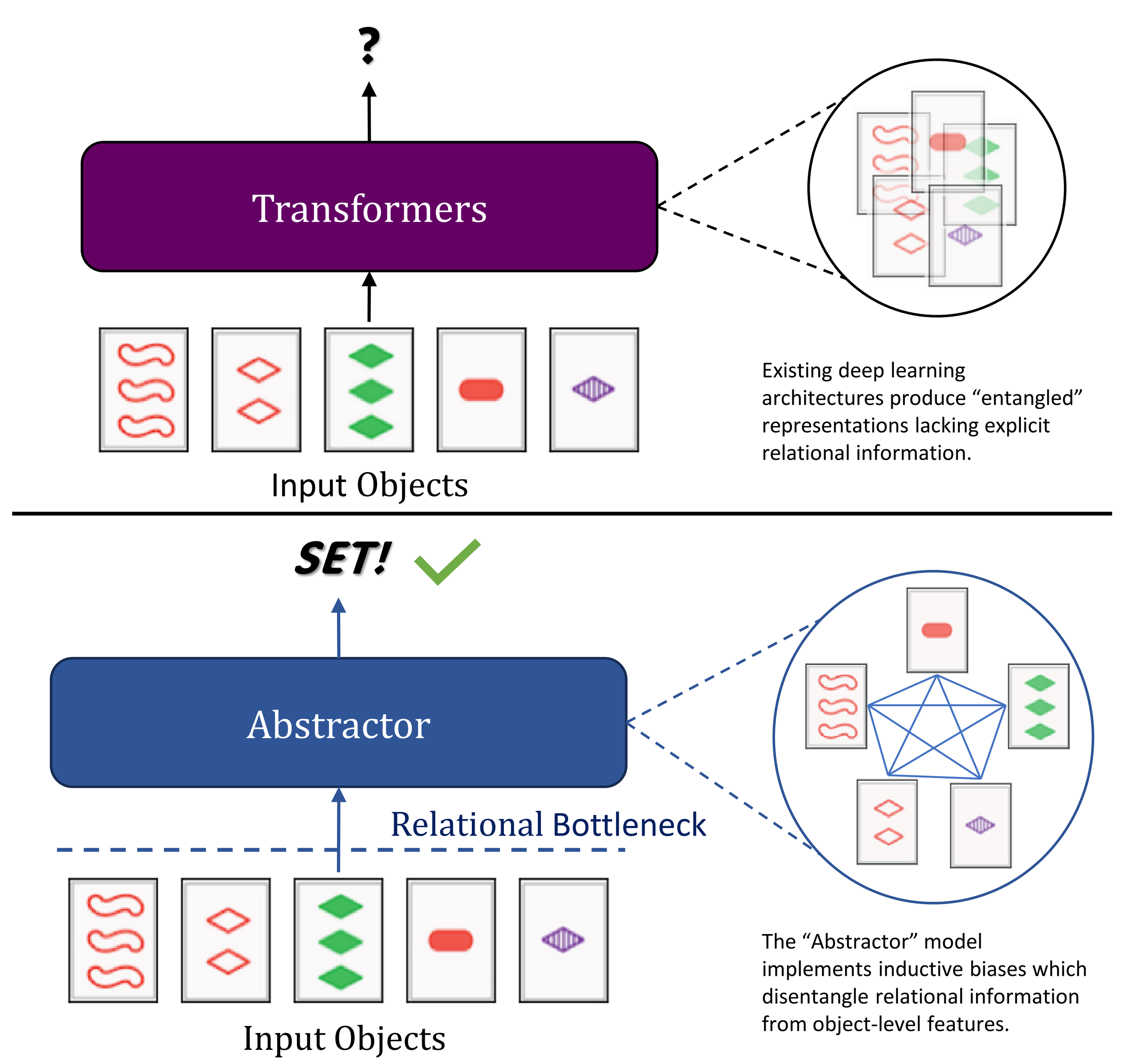
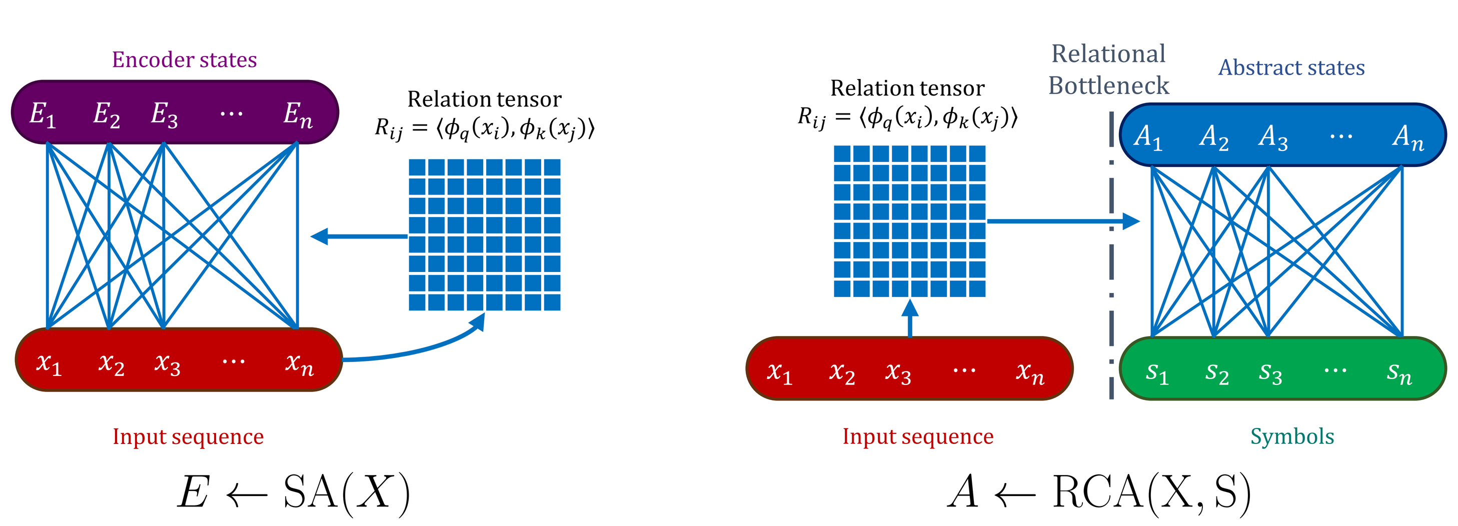



Figure 1. How the Abstractor differs from a standard Transformer: the Abstractor learns more organized and well-separated representations of relational features which are disentangled from object-level features.




Comparison of relational cross-attention (RCA) with self-attention (SA). RCA implements a relational information bottleneck, resulting in relational representations disentangled and from object-level features.

## Attention under the Neural Message-Passing Lens

Under the neural-message passing lens, self-attention is

$$\begin{aligned} y_i &\leftarrow \mathrm{Aggregate}(\{m_{j\to i} : j \in [n]\}), \\ m_{j\to i} &= (r(x_i, x_j), \phi_v(x_j)) \end{aligned}$$

The attention scores can be thought of as relations between pairs of objects that determine which object to "attend to" and retrieve information from.

In standard self-attention, computing these relations is merely an intermediate step in an information-retrieval operation.

The relations are *entangled* with object-level features. Because object-level features have much greater variability, they overwhelm the relational features in the attention representation.

## Relational Cross-Attention

We propose a modification of attention where the object-level features $\phi_v(x_j)$ are replaced with vectors $s_j$ we call symbols that *identify* objects, but do not encode their features.

Symbols live in a space with much smaller variability, inducing a relation-centric representation.

Relational cross-attention then implements the following neural message-passing operation,

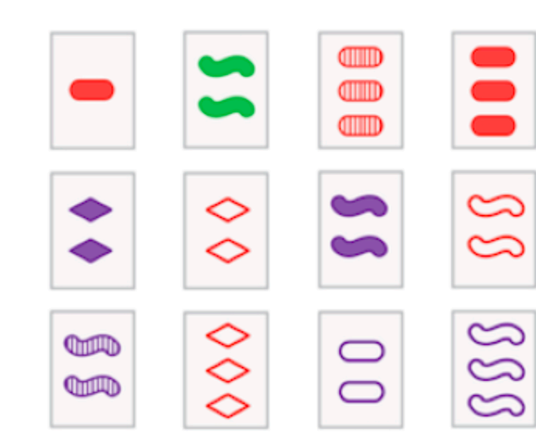
$$\begin{aligned} y_i &\leftarrow \mathrm{Aggregate}(\{m_{j\to i} : j \in [n]\}) \\ m_{j\to i} &\leftarrow (r(x_i, x_j), s_j) \\ s_1, \dots, s_n &\leftarrow \mathrm{SymbolAssignment}(x_1, \dots, x_n), \end{aligned}$$

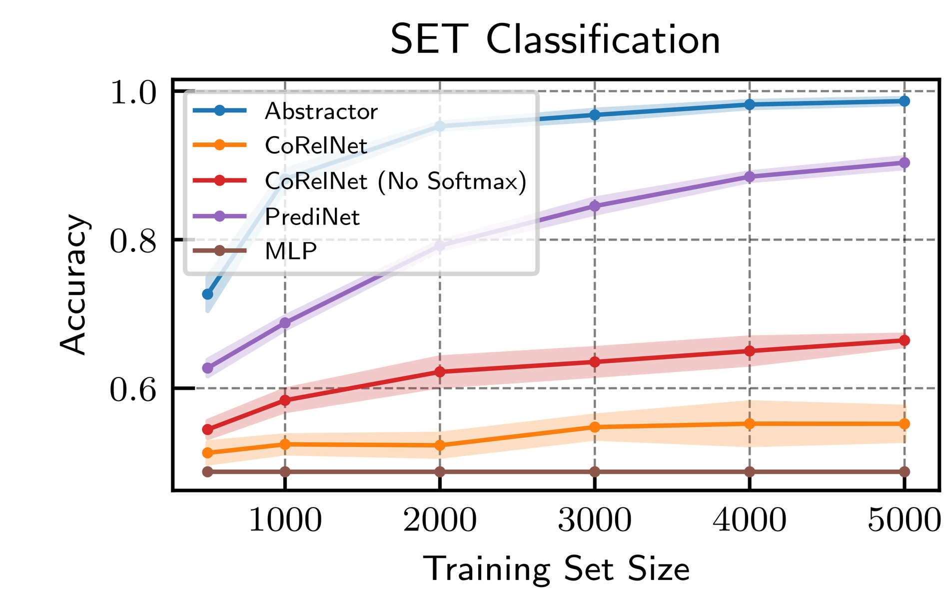
with the symbol $s_i$ identifying the object $x_i$ through position, relative position, and/or syntactic role.

This operation forms the core of the Abstractor module, which can be naturally incorporated into a broader Transformer-based architecture.

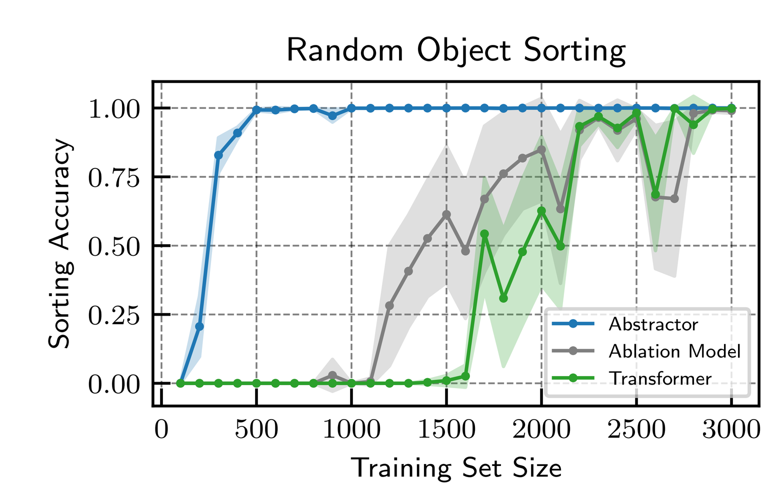
## Experiments

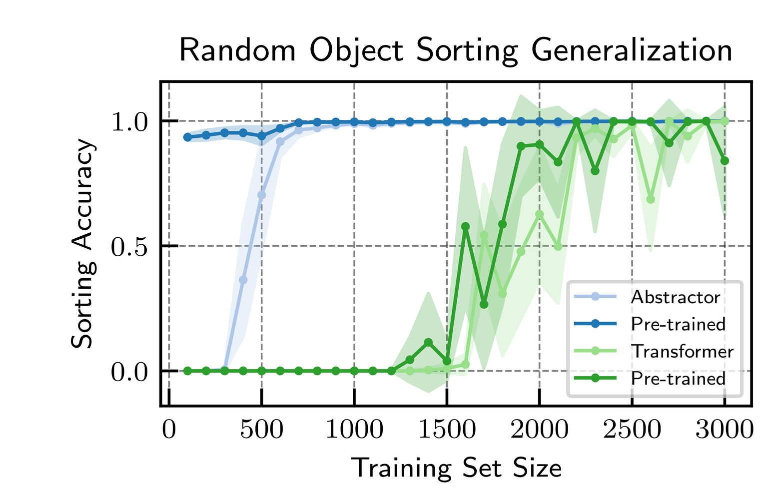
Empirical evaluation shows that the Abstractor is more sample-efficient and better able to generalize on tasks involving relational reasoning, compared to a standard Transformer.



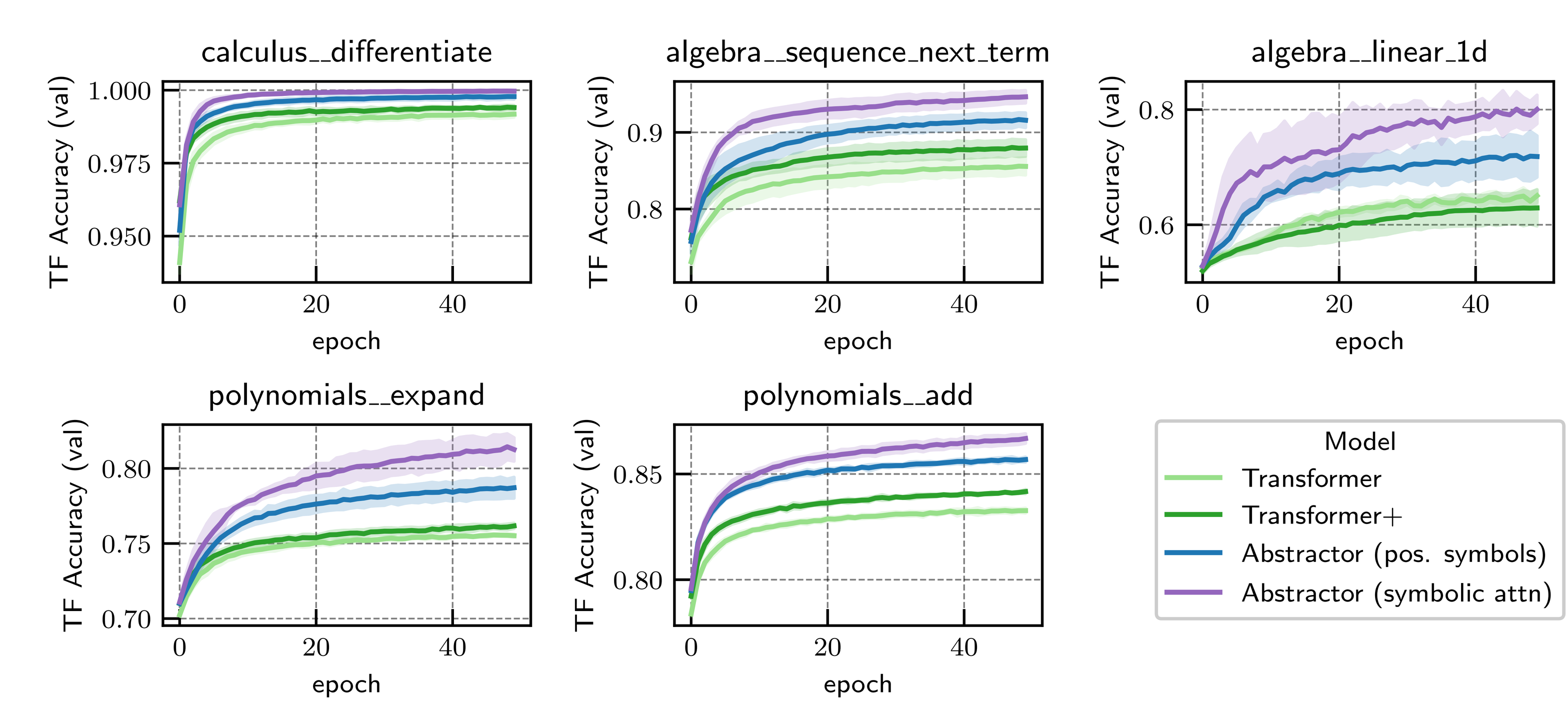



**Task:** *'SET!'*, a visual relational reasoning task. The learning curves show that the Abstractor outperforms existing relational architectures.




**Task:** learn to sort according to a latent order relation. Learning curves show that the Abstractor is more sample-efficient and generalizes to similar tasks.




**Task:** Character-level seq2seq task: Given a mathematical question, predict the answer. Abstractor learns faster and reaches higher accuracy.