Abstract

While Deep Learning has become increasingly capable over the past years, current models are still lacking key features of human cognition, especially when it comes to energy and data efficiency. One important component of human reasoning might be the ability to quickly abstract high-level features and perform computations on them. The relational bottleneck Webb, Frankland, et al. (2023) introduces a form of abstract reasoning into deep learning architectures and has shown high data efficiency on relational reasoning tasks resembling human IQ tests. In this paper, we re-implement the Emergent Symbol Binding Network (ESBN) Webb et al. (2021) and evaluate its performance when reducing the key size, a hyperparameter that sets the dimensionality of the abstract representations in the model. In the second part of the paper, incorporate the ESBN into a classical transformer architecture and evaluate its performance on a zero-shot language generalization task. While we were able to reproduce the findings of Webb et al. (2021) in the re-implementation part of the study, we did not find significant improvements on the language task when introducing a purely relational processing stream into the transformer both with and without the ESBN.

1 Introduction

Modern large language models have made great leaps showing near human performance across various benchmarks. However, they typically require vast amounts of training data and computational resources. A recent line of work introduced the so called ”relational bottleneck” Webb, Frankland, et al. (2023) which is a novel way of implementing symbolic processing into deep neural networks without manually defining the symbolic representations. Architectures implementing this bottleneck show superior data efficiency on relational tasks such as ravens progressive matrices, same different relations and distribution of three tasks Webb, Frankland, et al. (2023). It has also been shown that these models work extremely well on tasks that require relational as well as semantic processing Altabaa et al. (2023).

Russin et al. (2019) could show that introducing separate information processing streams for relational and semantic information improves model performance on a language based generalization task. In this paper, we re-implement the ESBN model by Webb et al. (2021) and evaluate the impact of reducing key size on training accuracy. In a second experiment, we integrate the ESBN into a standard transformer Vaswani et al. (2023) and compare its performance against a transformer and an Abstractor Altabaa et al. (2023) architecture on the language-based generalization task scan Lake and Baroni (2018).

2 Distribution-of-Three Experiment

In the first experiment, we evaluate models on the distribution of three task. The objective of the task is for the model to choose the right image from four possibilities to complement the pattern in the presented data. As in the original study Webb et al. (2021), we use simple Unicode images and generate the data by constructing random subsets. First, we recreate the $m = 0$ and the $m = 95$ condition from the original study and after that, we evaluate how decreasing the key size of the model impacts performance on the $m = 95$ condition.

2.1 Task

In this experiment, we consider the distribution-of-three task. The data consists of Unicode images in sequences of nine. The first three images of the sequence represent the pattern. The two following images are contained in the first three images so that there is one image left to complete the pattern.

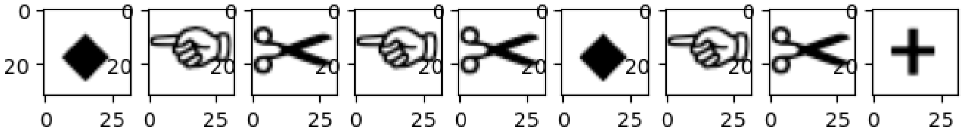

Figure 1: Sample from Distribution-of-Three layer. In this sample, the diamond, the hand and the scissor form one set. The subsequent two images are hand and scissor such that the image of the diamond remains to complete the set. Thus, the model should output index 0 since this is the index of the correct shape from the last four images.

Of the last four images, one is the image necessary to complete the pattern and three are random. The training objective is to select the index of the right image in the last four and can be expressed as a multiple choice task with four possibilities. The order of the images within the pattern does not matter in this task.
We recreate two conditions from the original study. In the m = 0 condition, all of the 100 Unicode images are used both in the train and evaluation data whereas in the m = 95 condition, 95% of the images are withheld from the training data. This condition is where generalization ability is most challenging for architectures such as the transformer Vaswani et al. (2023) and the LSTM Hochreiter and Schmidhuber (1997) as shown in Webb et al. (2021).

2.2 Model

In this experiment, we do a re-implementation of the ESBN model by Webb et al. (2021). The model has a convolutional encoder that creates 128 dimensional image embeddings. On the embedded image sequence, the model applies temporal context normalization Webb, Dulberg, et al. (2023) as did the authors in the original paper. The embeddings are then passed to the ESBN component of the model that is responsible for the relational processing. The ESBN has an LSTM controller component with a hidden size of 512 and from its output, 256 dimensional keys as well as the four dimensional predictions are produced at each timestep of the sequence. The model output are the last predictions that are derived from the LSTM output using a dense layer with the softmax activation function.

2.2.1 Emergent Symbol Binding Mechanism

At the core of the ESBN lie two separate memory stores M_k and M_v that grow dynamically when processing one sequence. The keys memorized in M_k are purely symbolic in nature since they are generated by a LSTM controller that operates only on relational information. The first key in M_k is generated by feeding in a vector of zeros to the LSTM controller. To compute the subsequent key in M_k, a weighted sum based on similarity scores of the elements in M_v is passed to the controller. The similarity scores for the weighted sum are calculated via the dot product between the input embeddings stored in M_v and are passed through a softmax nonlinearity. Additionally, a gain parameter g is calculated from a dense layer receiving the controller output and concatenated to the LSTM input. This mechanism allows only same-different information to be passed to the controller which is at the core of the relational reasoning capabilities of the network.

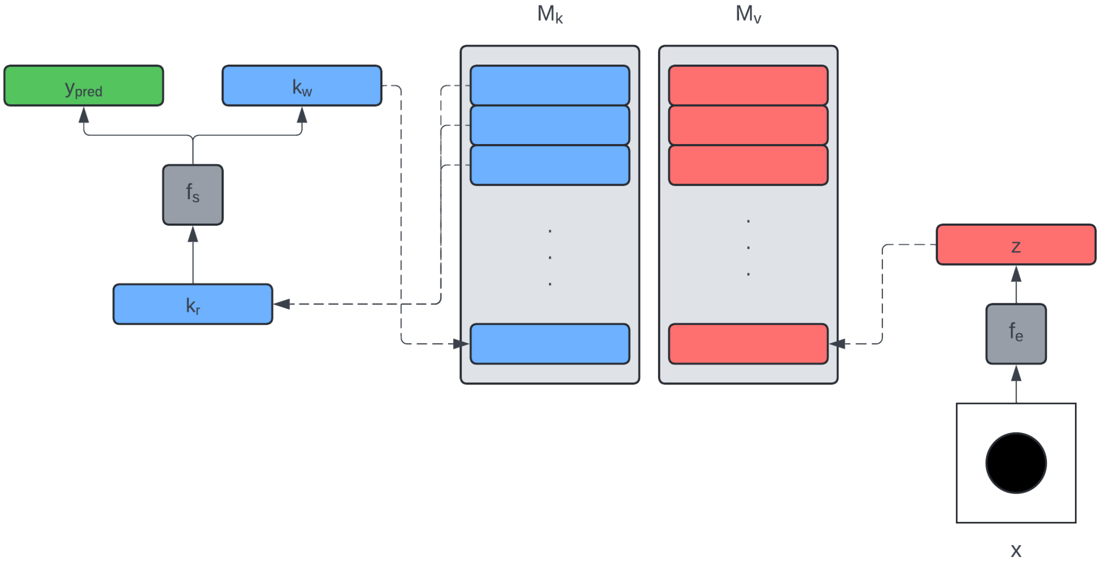

Figure 2: Emergent Symbol Binding Network as described by Webb et al. (2021). For simplicity, we do not show the dense network computing the gain parameter g from the LSTM output.

2.3 Results

2.3.1 Replication Study

For our replication study, obtain similar results as Webb et al. (2021) both for the m = 0 and the m = 95 condition. When evaluating the batch-level accuracy during training, we observe near perfect accuracy after only a some tens of iterations which is exactly what the original paper found as well.

2.4 Key Size Evaluation

To better understand what impact the key size has on the model performance, we systematically reduced this hyperparameter and compared validation accuracy for different key sizes. The key size is the size of the abstract symbols that the model uses and to better understand the properties of symbolic processing in the ESBN, it is interesting to see how varying this parameters impacts the model. For each condition, we trained ten models and compared the mean performance on the m = 95 condition.

2.5 Discussion

In the experiment, we successfully replicated the findings from Webb et al. (2021). The network architecture and the rationale behind it might open up interesting areas of research since it is an elegant combination of symbolic processing and bottom-up connectivist procedures.

Decreasing the key size up to a single dimension showed that the symbolic component of the architecture can tolerate extremely small representation sizes, at least for simple tasks. This further demonstrates the capability of the relational bottleneck as it does not necessarily rely on large symbolic representations to perform well.

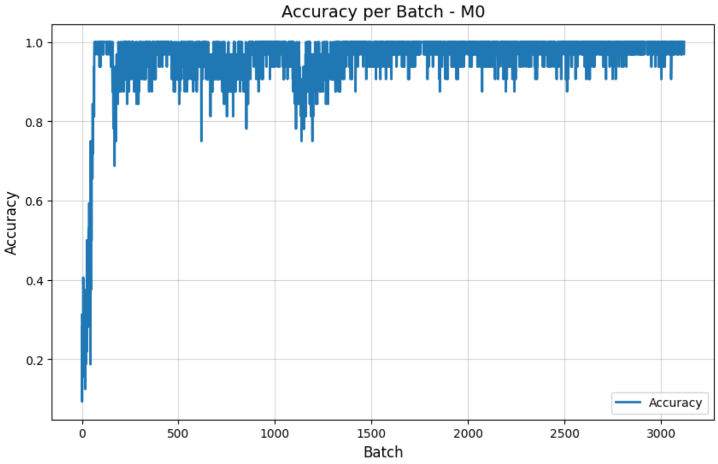

Figure 3: Train accuracy of the ESBN model in the m = 0 condition evaluated at the batch level. The plot shows the extremely rapid convergence of the model that was observed in the original study by Webb et al. (2021).

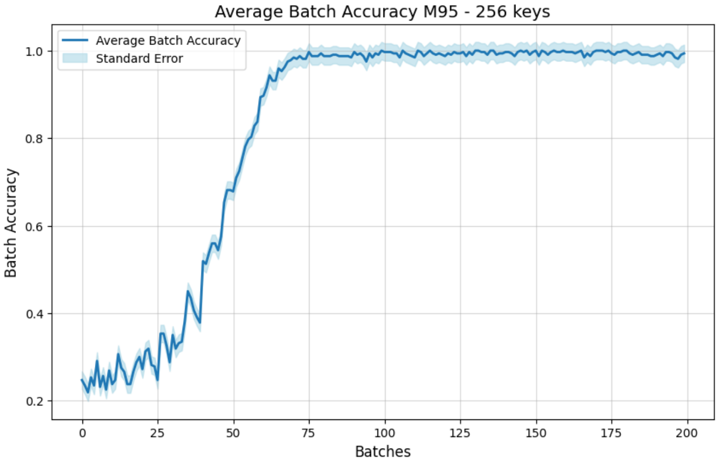

Figure 4: Mean train accuracy of ten train runs on the m = 95 condition with the original key size of 256 evaluated at the batch level.

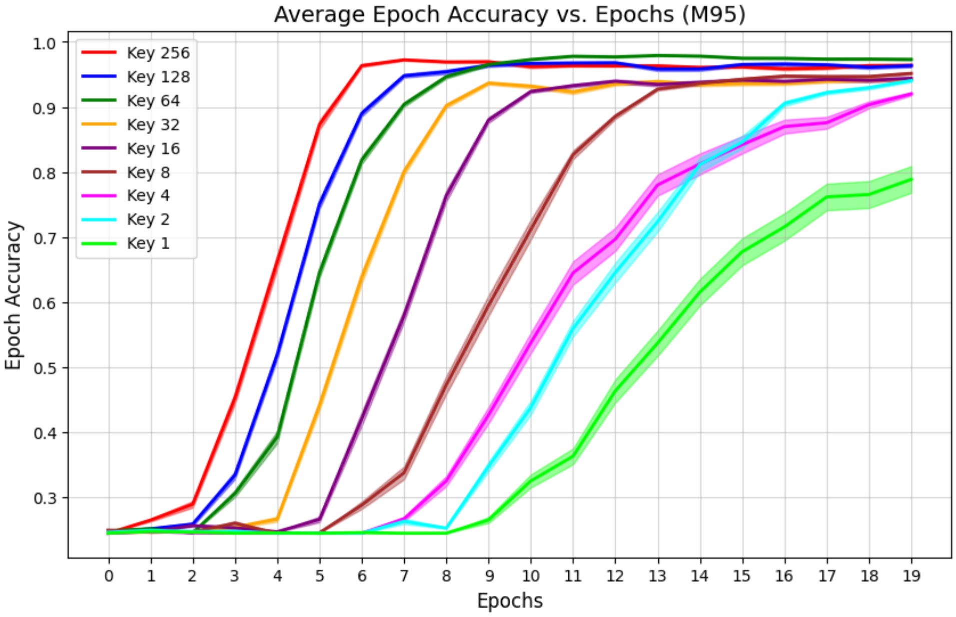

Figure 5: Model performance for different key sizes. Each line corresponds to the mean accuracy of ten models, shaded areas represent the standard error. The results suggest that decreasing the key size decreases the speed of convergence for the model, though the ESBN can tolerate even very small key sizes.

Also, the symbol binding mechanism is a biologically plausible simplified model of the processes that might happen in human reasoning. The human hippocampus receives highly processed information and it is able to recall associated information from memory. This whole process is modulated by the prefrontal cortex. This mechanism is mirrored by the ESBN whereby the memory components of the model are analogous to the hippocampus and the LSTM corresponds to the prefrontal cortex. The model is however a huge simplification and human processing is infinitely more complicated. One limitation, for instance, is that the model can only compare embedding via the dot product. In human reasoning, these comparisons should be able to take into account different features of the embeddings depending on the circumstances.

Another weakness of the ESBN model is that the LSTM is not ideal for handling very long sequences as it suffers from vanishing gradients and there are more adequate models that incorporate the relational bottleneck. One such model would be the Abstracter Altabaa et al. (2023) that builds on the transformer and allows symbolic processing to occur on a multitude of positions.

Also, as pointed out by Webb, Frankland, et al. (2023), the ESBN can only learn symmetric relations. This can be accounted for by stacking ESBN layers but the Abstracter Altabaa et al. (2023) is capable of learning asymmetric relations directly Webb, Frankland, et al. (2023).

3 Zero-Shot Generalization on Language Tasks

In this experiment, we use the ESBN to add a purely symbolic processing stream to a transformer. Since the ESBN shows superior performance on relational tasks (Webb 2023) it is plausible to assume that incorporating this architecture into a transformer would aid with generalization capabilities of the model.

One finding in this direction comes from (Russin 2019) who showed that the addition of a purely syntactic processing stream into a recurrent neural network improved the performance of the model on the generalization of language.

3.1 Task

The second experiment concerns the addprim_jump subset of the scan dataset Lake and Baroni (2018). The dataset tests models for zero-shot generalization on language tasks. It consists of verbal instructions for walking in different directions and the associated actions. In the addprim_jump subset, the word "jump" is not used in the instructions in the train set but it is used in the evaluation set. The model thus has to generalize to correctly interpret the word.
Our experiment is motivated by (Russing et. al, 2019) who used the dataset to evaluate generalization of a model that has seperate processing streams for semantics and syntax. Also, Altabaa et al. (2023) introduced the abstractor, a variant of the transformer that can incorporate the relational bottleneck at various positions in a transformer. In the paper, they argue that the standard transformer does in fact use relation based processing but that it entangles symbolic representations with embedding level features. It would be interesting to see if the ESBN could be used. Currently, large language models have made great leaps in their ability to reason at near human level (CITATIONS). This performance however requires vast amounts of both training data and computational resources. At the same time, the human brain runs on much less energy and is still able to generalize far better than any current AI system with low amounts of data. One possible contribution to this ability is the ability to do relational processing that facilitates abstract reasoning.
One interesting feature

3.2 Models

The baseline model in this section is a basic transformer Vaswani et al. (2023) with two layers, a model depth of 32, 4 attention heads and a feed forward dimension of 64. The encoder component of the model creates the context for the decoder from the input commands. The decoder then uses this context to compute the models predictions for the corresponding actions.
We compare this model against two variants that incorporate symbolic processing. The first variant uses an ESBN layer to generate symbols that are processed in a separate Abstracter Altabaa et al. (2023) module. We use a key size of 32 to facilitate later usage of the keys via attention and because Experiment 1 suggested that this dimensionality should not impact the ability to generalize.
Thus, the model generates two contexts for the encoder to attend to and for each context, we add a component to the decoder to perform cross-attention. Apart from that, the decoder works like the one introduced in Vaswani et al. (2023). The second variant works similar with the difference that symbols are not generated by an ESBN but are a set of learnable parameters of the model as was suggested by Altabaa et al. (2023).

3.3 Results

Due to computational restrictions, we ran only one training script per model with 20 epochs. In our experiment, we did not find significant differences in the average accuracy on the validation data.

3.4 Discussion

In our experiment, we did not find any convincing benefits of implementing the ESBN layer into a transformer architecture for generalization.
This might indicate that the transformer already combines relational and non-relational information

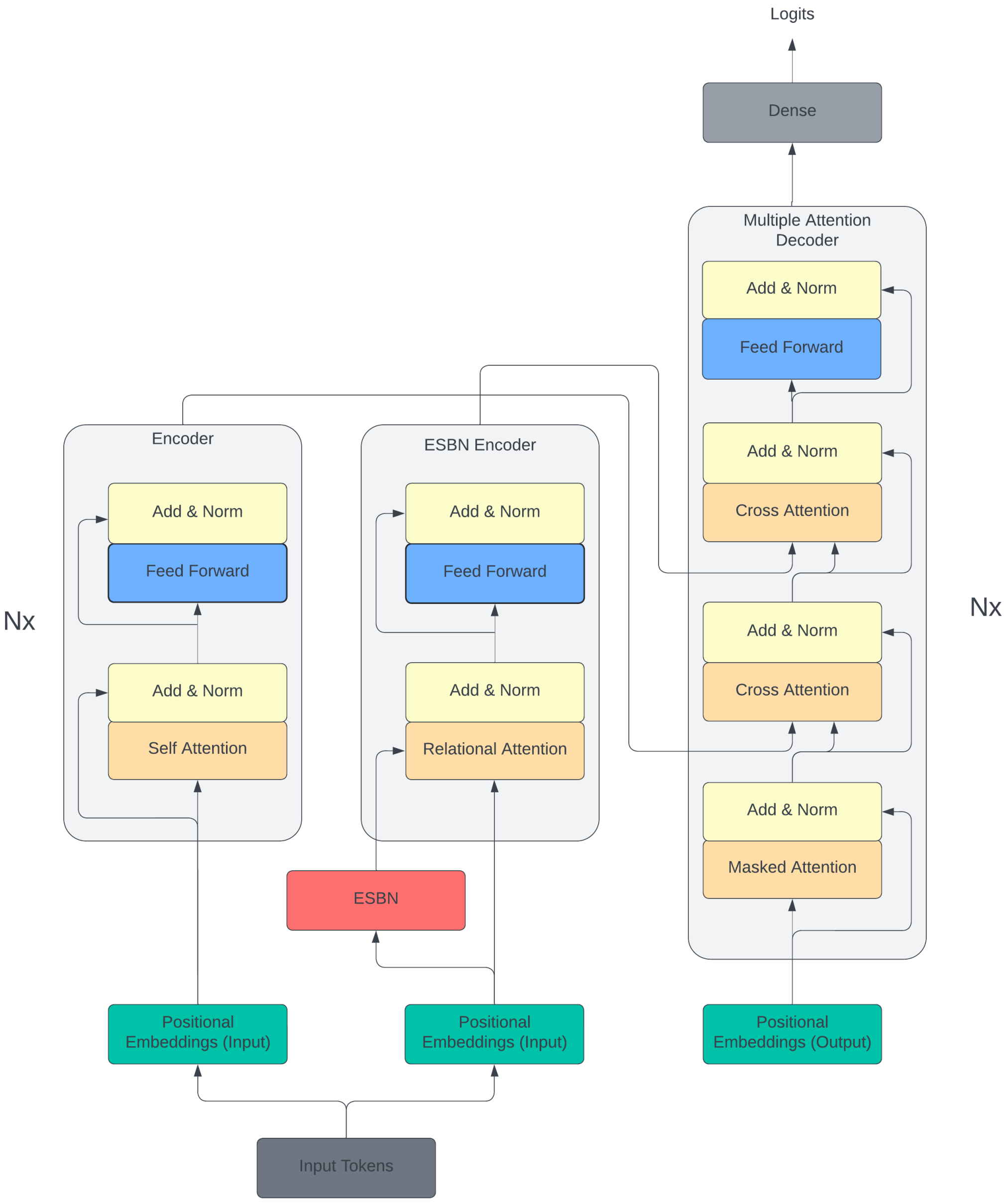

Figure 6: First variant of the transformer. The model has a second encoder that uses relational attention Altabaa et al. (2023) in order to allow for symbolic processing. The symbols are generated by an ESBN layer. The second variant differs only in the symbol generation since it uses learnable parameters as symbols.

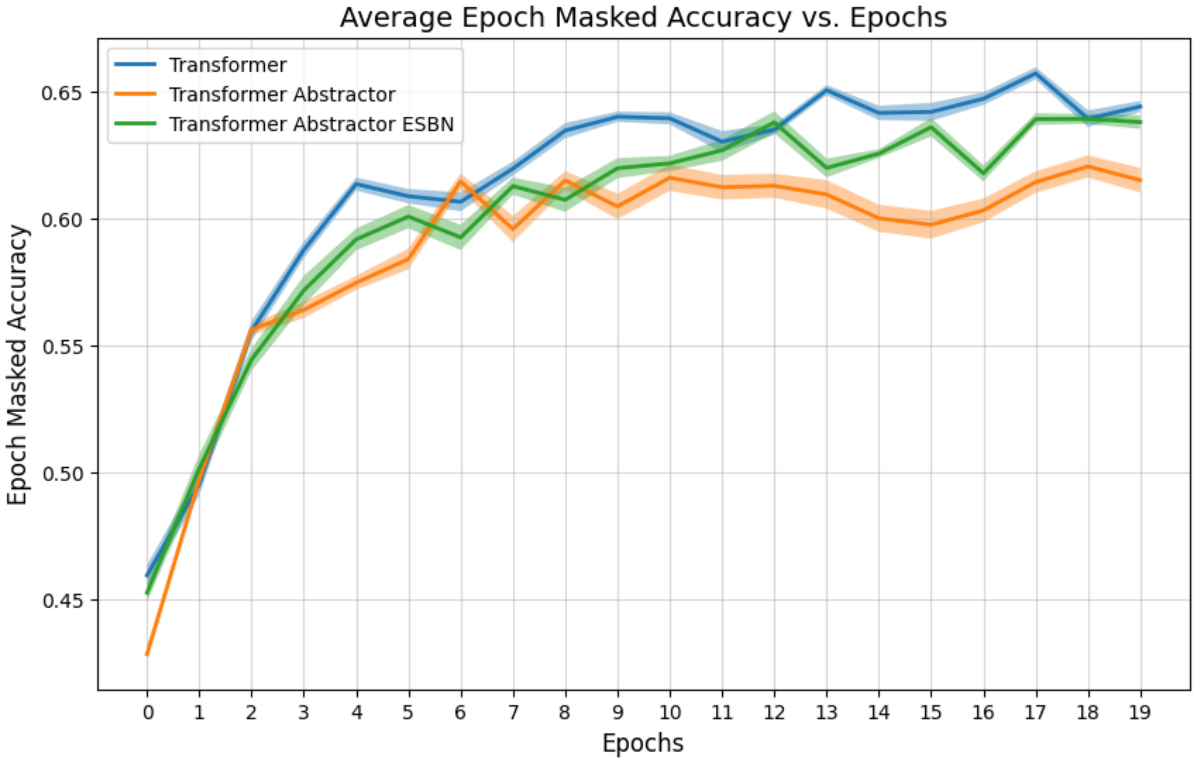

Figure 7: Average Masked Accuracy of the examined models on the scan task. Each model was trained five times, shaded areas show the standard error.

in a way that suit the task well. As Altabaa et al. (2023) point out, the transformer does indeed intertwine relational and non-relational features due to the attention mechanism used as the dot product between the key and query is a form of the relational bottleneck but the multiplication with the value component re-introduces low-level features. However, at least in our experiment, this might be actually help performance on the task.

Another explanation might be that the transformer does not need to rely on the symbolic component for training since it is already capable provide a near perfect fit for the training data without the symbolic component. So the model might ignore the symbolic information stream during training and does not learn to generalize.

4 General Discussion

The ESBN is one of several model architectures that implement the relational bottleneck Webb, Frankland, et al. (2023). The idea to include an inductive bias that predisposes the model to extract higher-level abstract features opens up many possibilities to extend generalization of models. One might reason that this ability is central to human cognition and integrating it into artificial neural networks might help both with modeling the human mind as well as improving the models themselves. One hint for the existence of a relational bottleneck in the human brain is the behavior that the ESBN displays in the Give-N-Task. It could be shown that the learning progression of the model, in contrast to other popular architectures, closely resembles the learning progression of actual human children Webb et al. (2021) as in Webb, Frankland, et al. (2023).

It remains to be seen how much current language models can benefit from symbolic processing but in our opinion, it could prove to be a powerful tool towards increasing data and computational efficiency of Large Language Models. Though, the ESBN is probably not ideally suited in that regard as it suffers from the same limitations as the LSTM and the memory component increases the required memory.

The Abstracter Altabaa et al. (2023) might be a more elegant choice to improve language models as it incorporates relational processing into transformer architectures more naturally and it is more flexible and memory efficient.

References

Altabaa, A., Webb, T., Cohen, J., & Lafferty, J. (2023, October). Abstractors and relational cross-attention: An inductive bias for explicit relational reasoning in Transformers [arXiv:2304.00195 [cs, stat]]. https://doi.org/10.48550/arXiv.2304.00195

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Kingma, D. P., & Ba, J. (2017, January). Adam: A Method for Stochastic Optimization [arXiv:1412.6980 [cs]]. https://doi.org/10.48550/arXiv.1412.6980

Lake, B. M., & Baroni, M. (2018, June). Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks [arXiv:1711.00350 [cs] version: 3]. Retrieved March 14, 2024, from http://arxiv.org/abs/1711.00350

Russin, J., Jo, J., O'Reilly, R. C., & Bengio, Y. (2019, May). Compositional generalization in a deep seq2seq model by separating syntax and semantics [arXiv:1904.09708 [cs, stat]]. https://doi.org/10.48550/arXiv.1904.09708

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023, August). Attention Is All You Need [arXiv:1706.03762 [cs]]. https://doi.org/10.48550/arXiv.1706.03762

Webb, T. W., Dulberg, Z., Frankland, S. M., Petrov, A. A., O'Reilly, R. C., & Cohen, J. D. (2023, September). Learning Representations that Support Extrapolation [arXiv:2007.05059 [cs]]. https://doi.org/10.48550/arXiv.2007.05059

Webb, T. W., Frankland, S. M., Altabaa, A., Krishnamurthy, K., Campbell, D., Russin, J., O'Reilly, R., Lafferty, J., & Cohen, J. D. (2023). The Relational Bottleneck as an Inductive Bias for Efficient Abstraction. https://doi.org/10.48550/ARXIV.2309.06629

Webb, T. W., Sinha, I., & Cohen, J. D. (2021, March). Emergent Symbols through Binding in External Memory [arXiv:2012.14601 [cs]]. Retrieved March 2, 2024, from http://arxiv.org/abs/2012.14601

5 Appendix

5.1 Code Availability

All the code used to create the experiments can be accessed via Github

5.2 Distribution-of-Three Experiment

5.2.1 Dataset Generation

The dataset generation procedure was adapted from Webb et al. (2021) and we generated the datasets from scratch for both the m = 0 and m = 95 condition. For the m = 0 condition, the train set size was 10000, for the m = 95 condition it was 320 since no more permutations can be produced from five shapes. Test set size was 10000 for both condition.

5.2.2 Hyperparameters

For the first part of the experiment, we have chosen the same hyperparameters as Webb et al. (2021) in order to properly recreate the experiment. The image encoder consists of three convolutional layers with 32 filters each and two dense layers with 256 and 128 units. Both of the dense layers use the relu activation function and we use a simple flatten layer to pass the output of the last convolutional layer to the dense layers.
The original model uses an embedding size of 128, a key size of 256 and a hidden size of 512 for the LSTM controller.
For all conditions, we used the Adam Kingma and Ba (2017) optimizer with a learning rate of 0.001. In the m = 0 condition, we trained for 10 epochs and for the m = 95 condition, we trained for 20 epochs. For all condition, a batch size of 32 was used both for training and evaluation.

5.2.3 Model evaluation

For the m = 95 condition, we performed a single training run and evaluated its performance on the batch level. For the key size evaluation, we performed ten training runs for each condition and used the mean and standard deviation of epoch accuracy to compare models.

5.3 Zero-Shot Generalization on Language Tasks

5.3.1 Dataset

We tested our models on the addprim_jump split of the scan Lake and Baroni (2018) dataset. The dataset consists of 14,670 examples for training and 7,706 for evaluation.

5.3.2 Hyperparameters

All the models had two layers, an embedding depth of 32, 64 units in the feed forward layers and four attention heads. For each attention block, we used dropout with a rate of 0.1. Also, we use layer normalization and addition of residual connections for better trainability.
We trained for 20 epochs using the Adam Kingma and Ba (2017) optimizer with a learning rate of 0.001 and a batch size of 32.

5.4 Model evaluation

We evaluated the mean masked accuracy for five training runs for each model.