COMPOSITIONALITY AS LEARNING BIAS IN GENERATIVE RNNS SOLVES THE OMNIGLOT CHALLENGE

Anonymous authors
Paper under double-blind review

ABSTRACT

One aspect of learning to learn concerns the development of compositional knowledge structures that can be flexibly recombined in a semantically meaningful manner to analogically solve related problems. We focus on learning to learn one-shot/few-shot generation and classification tasks of handwritten character trajectories, as described in the Omniglot challenge. We show that solving the challenge becomes possible, by suitably fostering a generative LSTM network to develop well-structured, compositional encodings, which can be quickly reassembled into new, unseen but related character trajectories. This is a major improvement compared to the original approach, which explicitly provided character components. We believe that the development of similarly compressed, compositional structures may also be highly useful to address related learning to learn challenges in other dynamic processing, prediction, and control domains.

1 INTRODUCTION

The current machine learning literature suggests that combinatorial generalization is unlikely to emerge in relatively generic recurrent neural networks (RNNs) (Lake et al., 2017). Here we disprove this widely believed assumption: we show that a generative long short-term memory network (LSTM) (Hochreiter & Schmidhuber, 1997) is in fact able to develop representations of components and to recombine them in a new but compositionally meaningful manner, which in turn shapes future learning.

Humans are very good at combinatorial generalization. When viewing, for example, a blackbird, children decompose it into its components, like wings, beak, feet etc. (Gopnik, 2019). As a result, they recognize these components in other blackbirds, and even other bird species, resulting in the correct classification of bird. Furthermore, children can rearrange these components in creative ways, imagine new blackbirds or even invent fictitious bird types. For machine learning systems, on the other hand, the ability of such efficient learning is still a major challenge. Therefore, the demand to include compositional capabilities into machines becomes more and more apparent (Franklin et al., 2020; Gopnik, 2019; Lake et al., 2017). Battaglia et al. (2018) even go as far as to 'suggest that a key path forward for modern AI is to commit to combinatorial generalization as a top priority'. In order to motivate researchers to investigate how human-like efficient learning based on compositionality, but also causality and learning to learn can be realized within machine learning algorithms, the Omniglot challenge has been introduced six years ago (Lake et al., 2015). It consists of the following generation and classification tasks of handwritten character trajectories: (i) one-shot regeneration of a character, (ii) one-shot generation of concept variants, (iii) one-shot classification, (iv) and few-shot generation of new concepts. In the same work (Lake et al., 2015), the researchers provided a model with a general idea on how to draw a character, by providing basic motor components, like half circles or straight lines, using Bayesian program learning. Since the release of the Omniglot challenge, many researchers from Google DeepMind, the MIT, and other universities, including Geoffrey Hinton and Josh Tenenbaum, aimed at solving the challenge without providing such basic components (Edwards & Storkey, 2016; Eslami et al., 2016; Fabi et al., 2020; Feinman & Lake, 2020; George et al., 2017; Gregor et al., 2016; Hewitt et al., 2018; Lake et al., 2015; Rezende et al., 2016; Shyam et al., 2017; Snell et al., 2017; Vinyals et al., 2016). In a summary about the progress on the Omniglot challenge within the last years, Lake et al. (2019) concluded that models' performance on one-shot classification has been largely improved (Shyam et al., 2017; Snell et al., 2017; Vinyals et al., 2016), whereas the progress on the other tasks was very limited. Various generated examples of the same concept

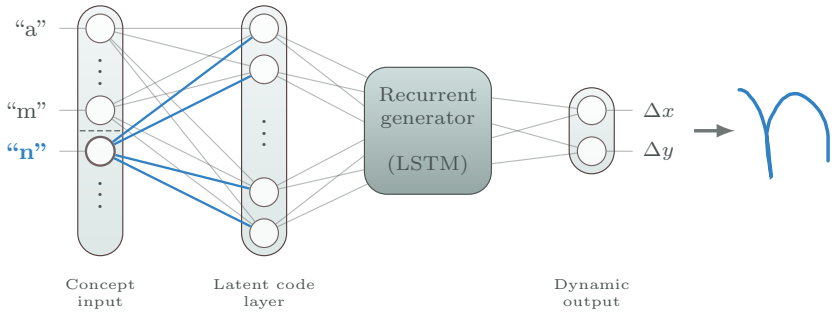


Figure 1: Illustration of the *recomp* mechanism. Only the blue weights that map the concept indicator (here of the new concept "n") onto a generative latent code are trained. The others parts of the network remain unchanged. Thus, the resulting trajectory is a composition of dynamical primitives from previously learned concepts.

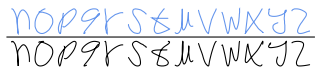
or of new concepts were either very similar or too dissimilar, so that one could not recognize them any more (George et al., 2017; Hewitt et al., 2018; Rezende et al., 2016). In other cases, only one task was tackled and no model was able to perform all the tasks at once (Edwards & Storkey, 2016; Eslami et al., 2016; Gregor et al., 2016). What seemed promising for solving the Omniglot challenge, though, was putting strong inductive biases about compositional structures into the models (Feinman & Lake, 2020; Lake et al., 2015). In their overview article, Lake et al. (2019) encourage the inclusion of causality (by applying sequential instead of pictorial data), learning to learn, and compositionality into more neurally-grounded architectures that can perform all instead of just some of the tasks.

In this paper, we present a way to solve the Omniglot challenge on sequential drawing instead of pictorial data. Thereby, we do not provide basic motor primitives, but we foster their development within an LSTM-based model. We incorporate the ability to recombine previously learned components in a meaningful manner when confronted with new concepts to accelerate their learning.

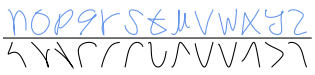
2 Model and *recomp* mechanism

In order to solve the Omniglot challenge's tasks, we applied a generative RNN as shown in Figure 1. This RNN consists of a variable-sized input layer, a linear latent embedding layer with 100 neurons, a recurrent generator module with 100 LSTM units (Hochreiter & Schmidhuber, 1997), and a linear output layer with two neurons. The input layer represents particular characters in form of one-hot encoded vectors. The number of input neurons corresponds to the number of concepts. Each input neuron projects its activity onto the next layer with its own set of weights. Thus, a concept indicator induces a specific activity pattern within the latent code layer. This code, which can be seen as the motor program encoding of the network, seeds and continuously shapes the unfolding dynamics within the recurrent generator. Eventually, the hidden dynamics are mapped onto the output layer, generating a change in x and y position at every time step.

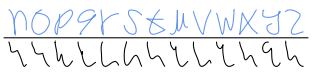
During training of 10 epochs on 440 trajectories of the first half of the Latin alphabet ("a" to "m"), the model learned to generate trajectories out of one-hot encoded inputs. Since the examples per character varied, the training resulted in the generation of average characters. When tackling the Omniglot challenge, the tasks should be solved with very few examples, which is why, after training, the model was presented with one example of a new character ("n" to "z"). If it had learned components during training as expected, it should be able to reassemble these representations compositionally in order to generate new trajectories. Therefore, when presented with new characters, we allowed only the first weights into the first feedforward layer (cf. blue weights in Figure 1) to adapt for 1 000 iterations per character. This should re-arrange the already learned representations of components, leaving the remaining parts of the network, including the recurrent layer, untouched. We call this procedure *recomp* mechanism, since it allows the network to *recom*bine the *comp*onents that we expect to be represented within the LSTM layer. We will provide further evidence for this hypothesis throughout this work.



(a) Pre-trained model plus *recomp* mechanism

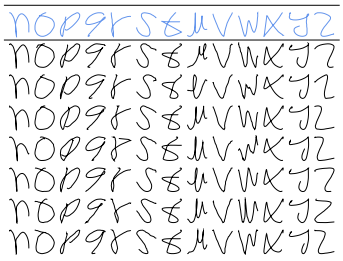


(b) *recomp* mechanism without pre-trained model

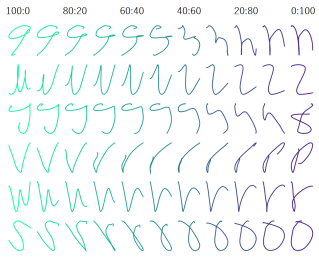


(c) Pre-trained model without *recomp* mechanism

Figure 2: Human handwritten (blue) and regenerated trajectories (black)



(a) Generating letter variants (black) after seeing the blue one once.



(b) Generation of new concepts through blended input vectors.

Figure 3: Generation tasks of the Omniglot challenge

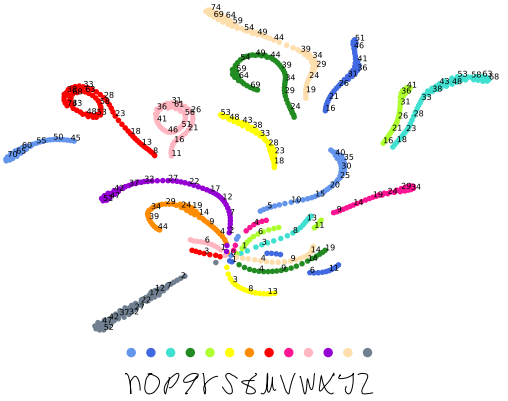
The generative LSTM models, together with the *recomp* mechanism, were able to regenerate new character trajectories, which have not been part of the training set and of which only one example was presented (cf. Figure 2). Applying the *recomp* mechanism on an untrained model did not lead to readable character generations, showing how important the training was and supporting our hypothesis that sequence components are learned that can later on be recombined in a compositional manner. It led to even worse results than the trained model without the *recomp* mechanism, showing that the *recomp* mechanism cannot be viewed as a generic training of the network. Rather, it compositionally rearranges previously encoded sequence dynamics.

3 Tackling the Omniglot challenge

To generate new variants of a character concept, after having applied the *recomp* mechanism, we added normally-distributed noise with a scale between 0.009 and 0.15 onto the one-hot encoded input vectors. The generation was successful and various variants per character can be viewed in Figure 3a. For the classification task, instead of a one-hot encoded input, the network got a zero vector of length 26 for every time step. The error between the generated and the trajectory of the presented variant was calculated and the gradient was backpropagated onto the input vector, which was then passed forward through the network again. This was repeated $10\,000$ times for every variant. The highest input activation represented the network's classification. If tested on the variants of Figure 3a, the mechanism classified $96,7\%$ correctly (88 out of 91 characters). Looking at the three mistakes more closely, they were not even implausible (e.g., the second "u" was classified as an "f"). For the last generation task of new concepts, the model was confronted with blended input vectors that indicated which character should be included into the mixture to which extent. The results can be seen in Figure 3b and show no abrupt changes, but very smooth blendings between two characters, supporting our hypothesis of compositionality. In short, the generative LSTM model, together with the *recomp* mechanism, was able to solve the tasks of the Omniglot challenge. This is impressive, since previous attempts to solve the Omniglot challenge used large amounts of background alphabets, complex algorithms, and often tackled only one instead of all tasks (Lake et al., 2019; Rezende et al., 2016; Edwards & Storkey, 2016).

4 Analysis of the LSTM hidden states

In order to further investigate our hypothesis that solving the Omniglot challenge was possible because the initial training led to representations of general components of characters in the hidden



(a) Projection into 2d space of cell states when generating different characters. The numbers represent the corresponding timesteps.

(b) Hidden states h. The corresponding trajectory parts are drawn in black with a cross marking their beginning. The respective trajectory plots are centered on the first respective cluster position. Note that the size of the trajectory parts is not representative.

Figure 4: Results of the t-SNE analysis of the cell and hidden states.

states of the LSTM layer, we analyzed the respective LSTM cell and hidden states when generating characters "n" to "z". The analysis we used was t-distributed stochastic neighbour embedding (t-SNE) (Hinton & Roweis, 2003; **?**) with 1 000 iterations.

The 2d-representations of the cell states are clearly clustered with respect to their corresponding character Figure 4. Thus, the c-states might be an important indicator for the network to stay in this attractor and generate this one character. Focusing on the "w", the spiral reflects the two similar components of which the trajectory is made. Other components shared between characters can also be identified in close proximity, like the half circle and downwards stroke in "q" and "y", or the stroke from bottom to top in "r" and "p" that look very similar in the current trajectory variants. The projection of the hidden states h onto the 2d space identifies clear character components, because similar components led to sequences in close proximity Figure 4. For example, on the left, there is a group of bottom to top trajectory parts, curves in specific directions are clustered next to each other, and the "u" encoding in the middle reflects the fact that it is generated by two very similar components, which are encoded in the almost overlapping red circles. This speaks for our hypothesis that components are represented in the LSTM hidden states.

5 Conclusion

The Omniglot challenge can be solved with a generative LSTM model without providing it any knowledge about specific motor components. During training on some characters, the model formed representations of components that it could recombine with the help of the *recomp* mechanism when it was confronted with new characters, facilitating future learning. By visualizing the hidden states of the LSTM cells, we found evidence that such compositional structures developed within the hidden states, making the mechanisms within the model more explainable. Learning to learn is incorporated in that way that the gradient signal (thus the learning signal) is directly shaped by the previously learned representations, thus contrasting pure transfer learning approaches. With respect to no-free lunch, we are making the assumption that all dynamic patterns can be suitably compressed into compositional structures, which can then be reused to learn related types of problems fast.

Ultimately, this research did not only lead to the resolution of all tasks of the Omniglot challenge with one fully connected LSTM model, but can be seen as a step towards bringing a specific machine learning architecture towards closer resemblance to human cognitive mechanisms, namely combinatorial generalization and thus learning to learn.

REFERENCES

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Harrison Edwards and Amos Storkey. Towards a neural statistician. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

S. Fabi, S. Otte, J.G. Wiese, and M.V. Butz. Investigating efficient learning and compositionality in generative lstm networks. In I. Farkas, P. Masulli, and S. Wermter (eds.), *Artificial Neural Networks and Machine Learning - ICANN 2020*, pp. 143–154. Springer, 2020.

Reuben Feinman and Brenden M Lake. Learning task-general representations with generative neuro-symbolic modeling. *arXiv preprint arXiv:2006.14448*, 2020.

Nicholas T. Franklin, Kenneth A. Norman, Charan Ranganath, Jeffrey M. Zacks, and Samuel J. Gershman. Structured event memory: A neuro-symbolic model of event cognition. *Psychological Review*, 127(3):327–361, 2020. ISSN 1939-1471(Electronic),0033-295X(Print). doi: 10.1037/rev0000177.

Dileep George, Wolfgang Lehrach, Ken Kansky, Miguel Lázaro-Gredilla, Christopher Laan, Bhaskara Marthi, Xinghua Lou, Zhaoshi Meng, Yi Liu, Huayan Wang, et al. A generative vision model that trains with high data efficiency and breaks text-based CAPTCHAs. *Science*, 358(6368), 2017.

A. Gopnik. AIs versus four-year-olds. In J. Brockman (ed.), *Possible Minds: Twenty-five ways of looking at AI*. Penguin Press, New York, 2019.

Karol Gregor, Frederic Besse, Danilo Jimenez Rezende, Ivo Danihelka, and Daan Wierstra. Towards conceptual compression. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Luke B Hewitt, Maxwell I Nye, Andreea Gane, Tommi Jaakkola, and Joshua B Tenenbaum. The variational homoencoder: Learning to learn high capacity generative models from few examples. In *Uncertainty in Artificial Intelligence*, 2018.

Geoffrey E Hinton and Sam Roweis. Stochastic Neighbor Embedding. In S. Becker, S. Thrun, and K. Obermayer (eds.), *Advances in Neural Information Processing Systems*, pp. 857–864. MIT Press, 2003.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. The omniglot challenge: A 3-year progress report. *Current Opinion in Behavioral Sciences*, 29:97–104, 2019.

Danilo Rezende, Ivo Danihelka, Karol Gregor, Daan Wierstra, et al. One-shot generalization in deep generative models. In *International Conference on Machine Learning*, pp. 1521–1529. PMLR, 2016.

Pranav Shyam, Shubham Gupta, and Ambedkar Dukkipati. Attentive recurrent comparators. In *International Conference on Machine Learning*, pp. 3173–3181. PMLR, 2017.

Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

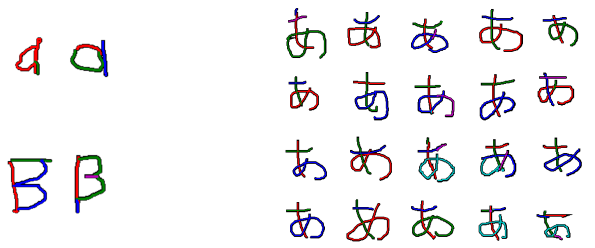


Figure 5: Examples of the sequential Omniglot dataset provided by Lake et al. (2019). Colors represent consecutive strokes in the following order: red, green, blue, purple, turquoise. Note how "a" and "beta" as well as the first character of the Japanese katakana alphabet are drawn with unusually many strokes and in an inconsistent sequential manner.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

A Appendix

A.1 Applying the original Omniglot dataset

The Omniglot data was originally pictorial data containing 50 alphabets, with 20 variants per character (Lake et al., 2015). To include stronger forms of compositionality and causality, Lake et al. (2019) added a sequential stroke dataset, for which 20 Amazon Mechanical Turk participants traced the pictures of the original characters. The introduction of the Omniglot dataset and the Omniglot challenge, as well as the further introduction of the sequential dataset was of tremendous importance for the Machine Learning community. Nevertheless, we want to criticize the sequential dataset in a certain regard. On the left handside of Figure 6, there are two examples of "a" and "beta" with the different strokes highlighted in different colors. It becomes apparent that the characters were not naturally drawn with a pen, but traced with a computer mouse, leading to "a"s and "beta"s that are composed of three or four different and rather arbitrary strokes instead of just one, which would resemble a natural writing movement. This problem might be even larger for unknown alphabets, about the generation of which the Amazon Mechanical Turk participants had no background knowledge. It was most problematic for alphabets with a manifold of different strokes instead of just a few, which is illustrated by the heterogeneous stroke orders of the first character of the Japanese alphabet (cf. right handside of Figure 6).

Because of these shortcomings, we applied the new dataset of handwritten character trajectories of the Latin alphabet in the main paper. With this, we want to ensure that the characters are produced by experts of the alphabet, that they are generated freely instead of tracing previously drawn characters, leading to consistent, natural, and correct trajectories. Furthermore, instead of a rather imprecise computer mouse, the participants of the new dataset used a dedicated pen on a touch-sensitive surface, making their drawings more realistic. Furthermore, the 20 variants of the Omniglot dataset are very similar, whereas the new dataset provides more natural variability in 440 examples per character from 10 different subjects, including script and print characters.

Nevertheless, we also applied our network architecture to the sequential Omniglot dataset (Lake et al., 2019). To do this, we transformed the trajectory data into difference values of x and y positions. We furthermore deleted the information about when a new stroke ended in order not to prime the network, but to let it develop its own compositional representations. Because of the shortcomings of the sequential Omniglot dataset described above, which are worse in characters that are composed of lots of different strokes, we selected alphabets that are complex, but originally not composed of too many strokes.

When tackling the omniglot challenge most researchers applied 30 or more background alphabets for training (e.g., Hewitt et al., 2018; Rezende et al., 2016). Since humans do not need so many background alphabets and since most of the components are already represented in very few alphabets,

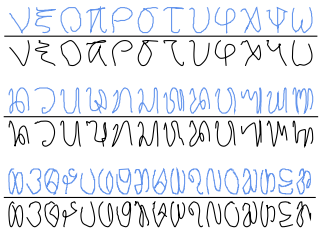


Figure 6: Original (blue) and regenerated (black) character trajectories of second half of the Greek, Balinese, and Burmese alphabets, after being trained on the respective first halves.

Figure 7: Original (blue) and regenerated (black) character trajectories of the Greek, when trained on the Latin alphabet.

we hypothesized that our compositional approach does not need training on that many character concepts. Therefore, we decided to train on only one alphabet, giving us the additional opportunity to perform experiments with different alphabet combinations, that either do or do not share components. Nevertheless, it must be mentioned that the variants of one character concept are unnaturally similar in the Omniglot dataset, whereas humans are confronted with more varying examples (as represented in the new dataset used in the main paper).

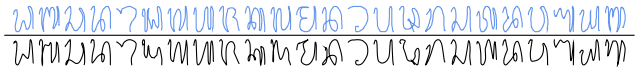
First, we tested whether the LSTM model and the *recomp* mechanism also perform well on single alphabets of the Omniglot dataset. Therefore, we trained the models on the first half of the Greek, the Balinese, or the Burmese alphabet for 500 epochs. Then, we provided 2000 iterations of the *recomp* mechanism per character of the second half of these alphabets. Even though in this experiment, the network was only trained on 20 similar variants instead of 440 varying ones per character, the results for the one-shot regeneration of the characters of the second half of these alphabets look quite promising, as can be seen in Figure 7, as well as in the Dynamic Time Warping (DTW) distances: 0.356 (Greek), 0.378 (Balinese), 0.241 (Burmese).

Next, we tested whether the model could also use previously learned components of another alphabet. Therefore we selected two alphabets of the original Omniglot dataset which share components: We trained the model on the Latin alphabet for 500 epochs and then presented it with one variant of each character of the Greek alphabet. Together with the *recomp* mechanism (2000 iterations per character), the model was able to regenerate most of the characters well with a DTW value of 0.329 (cf. Figure 8).

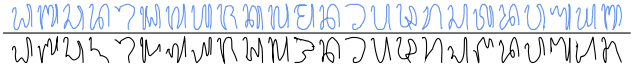
To investigate further whether recombining previously learned compositional representations leads to the success in learning new characters efficiently, we selected alphabets of the original Omniglot dataset for training and test with similar or differing components. For 500 epochs, we trained the generative LSTM network on the Burmese, Greek, or Latin alphabet, or the first half of the Balinese alphabet and tested its performance when confronted with one variant of the characters of the Balinese alphabet. Since the components of the first and the second half of the Balinese alphabet should be the most similar, we expected best performance for this combination, followed by the Burmese-Balinese combination, since their characters share lots of components. Not so many components are shared between the Balinese and the Latin, or Greek alphabets, which is why we expected worst performance here, assuming our compositionality hypothesis is true. The results are depicted in Figure 9. Supporting our hypothesis, the *recomp* mechanism led to the best performance for training on the first half of the Balinese alphabet, followed by the Burmese, Latin, and Greek alphabet (DTW distances: 0.378 vs. 0.384 vs. 0.427 vs. 0.509)

The model together with the *recomp* mechanism was able to generate most of the characters of the second half of various different Omniglot alphabets when having been trained on the first half. It was further able to generate characters of one alphabet when it had been trained on an alphabet with shared components. This is impressive because we applied only one instead of many background alphabets

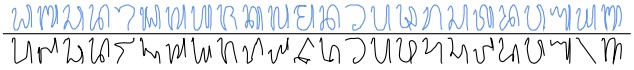
(a) Training on first half of the Balinese alphabet provided all necessary components leading to best regeneration.



(b) Training on the Burmese alphabet provided most necessary components leading to good regeneration.



(c) Training on the Latin alphabet did not provide all necessary components leading to worse regeneration.



(d) Training on the Greek alphabet provides even less components leading to worst regeneration.

Figure 8: Original (blue) and regenerated (black) character trajectories of the Balinese alphabet, trained on the first half of the Balinese, or the whole Burmese, Latin, or Greek alphabet.

and used the sequential, somewhat erroneous instead of the formerly mostly applied pictorial dataset (in order to foster stronger forms of causality). Supporting our hypothesis that during training, compositional representations are formed, which can later on be recombined when confronted with new characters, the regeneration performance decreased when trained and tested on alphabets that do not share similar components.