A Framework for Studying Communication Pathways in Machine Learning-Based Agent-to-Agent Communication

Sathish Purushothaman, Michael Granitzer, Florian Lemmerich, Jelena Mitrovic
University of Passau, Germany
{*sathish.purushothaman, michael.granitzer, florian.lemmerich, jelena.mitrovic*}*@uni-passau.de*

Keywords: Multi-Agent Learning, Deep Neural Networks, Language Emergence

Abstract: The rise of Large Language Models (LLMs) has increased the relevance of agent-to-agent communication, particularly in systems where agents learn from their interactions. However, current LLMs offer limited insights into the communication dynamics among multiple agents, especially in large-scale settings when multiple agents are involved. Particularly training LLMs - in contrast to in-context learning - becomes nearly infeasible without large-scale computing infrastructure. In our work we present a machine-learning based agent framework to investigate the role of different communication pathways for studying language emergence between machine learning-based agents. We designed a transformer-based image auto-encoder as the agent architecture. A Gumbel SoftMax layer encodes images in form of symbols forming the language between agents. We study two pathways: In the first pathway, the sender reads an image and sends a message to the receiver. The receiver uses the message to reconstruct the sender's image. In the second pathway, the sender and receiver read an image and minimize the distance between the generated symbols. In the first pathway, language emerges with the Levenshtein distance of ≤ 2 for 96% of messages. In the second pathway, no language emerges with the Levenshtein distance of ≤ 2 for 3% of messages.

1 INTRODUCTION

In the deep learning era, the study of emergent communication (Lazaridou and Baroni, 2020) explores how intelligent, machine learning-based agents interact to address specific problems. Large language models (LLMs) have demonstrated a form of agency (Wu et al., 2023) and the ability to learn from one another (Dubois et al., 2023). However, findings in this area tend to be more anecdotal or algorithmically derived rather than conceptual. Moreover, the expansive parameter space of LLMs constrains training updates, often relegating learning from communication to mere in-context adaptation. Additionally, reliance on single models restricts the scope of insight to various contexts of a single model, which may involve interdependent contexts in contrast to engaging multiple independent models with distinct separation points. This limitation is particularly salient in studying phenomena such as the emergence of languages or the transfer of knowledge between agents through language. Just as LLMs may not always convey their "thoughts" explicitly (Turpin et al., 2023), it is plausible that a singular, albeit very large, model does not fully capture the emergent communication phenomena. Thus, while LLMs exhibit the capability for agency and inter-agent communication, they do not constitute an ideal platform for systematically investigating agent-to-agent communication among a large number of agents.

Therefore, we propose a small and scalable multi-agent communication framework to study communication pathways and structures among multiple agents. The framework consists of multiple agents, communication pathways, and a world model defining perceptions and associations of images and associated labels. A successful communication emerges between two agents when a receiver agent can understand the sender agent's message in the way the sender intended based on the sender's perception. A message consists of a sequence of symbols used by agents for communication. In our problem, we have a set of images, their corresponding semantics, and symbols learned by agents through communication. In particular, two agents successfully communicate with each other when a receiver agent can reconstruct the image (or the semantics of the image) seen by the sender through the sender's message only. If both agents can reconstruct the images using each other's symbols, we can conclude that a shared language be-

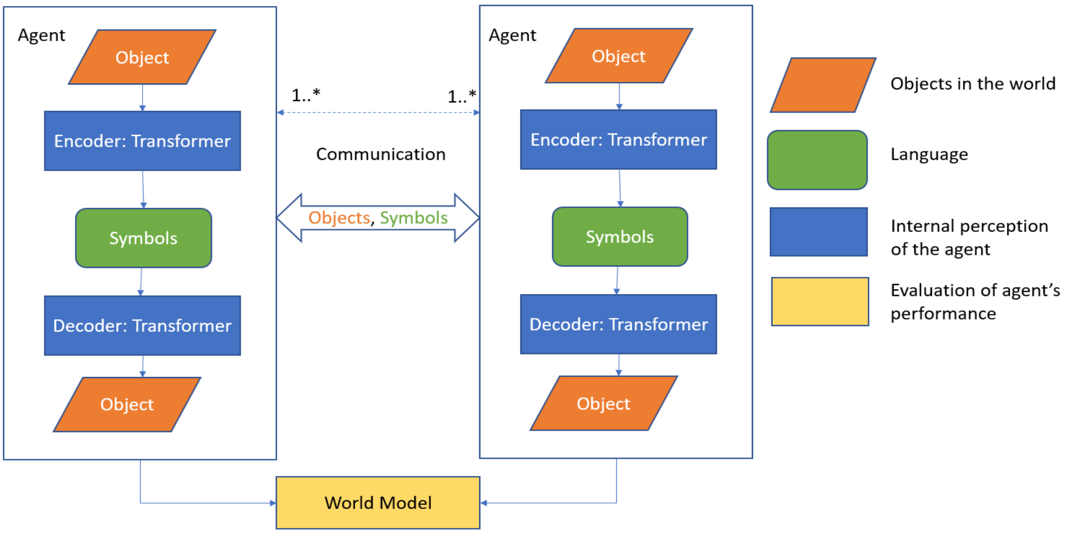

Figure 1: An agent is a transformer-based encoder-decoder architecture. The Gumbel SoftMax layer generates the symbols of the language. Each agent can share an object and a message with another agent. The communication can be among many agents. The communication framework allows us to study agent efficiency, convergence, and similarities under different communication pathways.

tween the agents has emerged. We do not fix the symbolic representation but let each agent learn it in an auto-encoder-like setting.

We study two communication pathways given the above setup. In the first communication pathway, the agents pay attention to the receiver's intention instead of the symbol similarity. The receiver agent's objective is to reconstruct an image using only the sender's symbols. In the second communication pathway, agents pay attention only to the similarity of the symbols and not to the receiver's intention. Each agent's objective is to develop a shared language representation.

Given this setup, we conduct experiments to answer the following research questions.

1. In the first communication pathway, does the successful reconstruction of images based on symbols shared between two agents lead to the emergence of language?
2. In the second communication pathway, does the sharing of symbols between two agents lead to the emergence of language?

In our work, we create a transformer-based multi-agent communication framework where multiple agents can communicate through symbols, as shown in Figure 1. We designed the agents to represent what each agent sees in the world in a sequence of symbols (e.g., image to text) and represent what it hears in objects in the world (e.g., text to image). Symbols impose a discrete bottleneck in communication. Agents reuse the same symbols to represent an infinite combination of objects in the world. The agents communicate with each other through a symbolic layer implemented using Gumbel SoftMax (Jang et al., 2017). The objective function is to minimize the mean squared error over the two agent messages and the binary cross entropy between the original and reconstructed image. We provide a world model which validates the performance of each agent.

Our results indicate that agents can develop a shared language within the Levenshtein distance of 2. The receiver agent can reconstruct the image using the symbols communicated by the sender agent. The structural similarity index measure (SSIM) between the sender's perceived image and the receiver's reconstructed image is 0.60. Our shared code is located in the github repository[1].

2 RELATED WORK

The study of emergent language using deep learning in a multi-agent setup can be subdivided into reinforcement learning and iterative learning (Lazaridou and Baroni, 2020; Brandizzi, 2023). In a reinforcement learning setup, multiple agents communicate with each other in a referential game or a reconstruction game (Batali, 1998; Kottur et al., 2017). In a referential game, a sender agent produces a message that refers uniquely to a specific object. The receiver

[1] https://github.com/sathishpurushothamanin/transformer-based-agent-communication-framework

agent gets the sender's message and a mixture of objects containing the sender's perceived image.

The receiver agent should refer to the correct object pointed by another agent among a group of dissimilar objects. Both sender and receiver agents get rewarded when the receiver agent can correctly identify the object pointed to by the sender.

In a reconstruction game, a sender agent produces a message that refers uniquely to a specific object. The receiver agent has to reconstruct the specific object using the sender's message. Both agents get rewarded when there is a successful reconstruction of the object.

In iterative learning (Rita et al., 2022; Li and Bowling, 2019; Ren et al., 2020), a student agent learns from a teacher agent or a population of agents pass on knowledge over each generation.

Recently, there are a growing number of frameworks proposed to study the multi-agent communication (Jinxin et al., 2023; Wu et al., 2023; Park et al., 2023; Li et al., 2023).

To study compositionality, (Chaabouni et al., 2019) and (Evtimova et al., 2018) used LSTM network architecture with and without attention, (Słowik et al., 2020) used a graph-based deep neural network, recently (Ri et al., 2023) investigated the role of attention-based deep neural networks.

3 TRANSFORMER-BASED MULTI-AGENT COMMUNICATION FRAMEWORK

Humans use natural language to describe the world using various word combinations with limited vocabulary. As shown in Figure 1, we model the problem as multiple agents interacting with each other without a human in the loop. According to the information bottleneck principle (Tishby et al., 2000), the communication bottleneck with limited communication capacity forces agents to convey only the shortest information possible. Each agent reuses the symbols in the vocabulary to refer to novel objects.

3.1 Communication Pathways

The communication framework presents two key challenges corresponding to different information flows between the agents. The first challenge is the ability of the receiver agent to recreate the image seen only by the sending agent. The second challenge is how the agents can develop a shared language effectively.

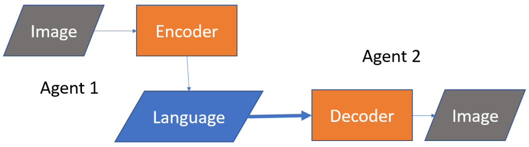

Figure 2: Agent 1 shares its representation of the world with Agent 2 using a symbolic message. Agent 2 decodes the message to generate its perception of the world. The first pathway allows us to study if two agents develop a shared language when each agent recreates an image seen by the other agent.

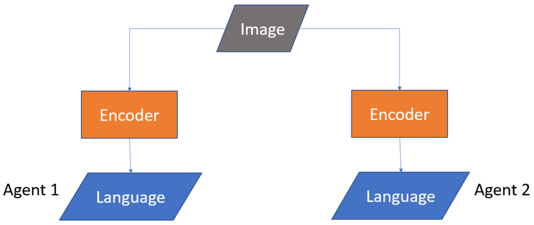

Figure 3: Both agents share their symbolic representation of the world. The second pathway allows us to study if two agents sharing symbols develop a shared language.

3.1.1 Communication Pathway 1: Semantic Exchange

As shown in Figure 2, the sender agent can use the encoder to convert the images into a sequence of symbols and send them to the receiver. The challenge for the sender is to convey only relevant semantic information because of the limited discrete symbolic message. The receiver can use the symbols to reconstruct the images. The challenge for the receiver is to use only the relevant semantic information to reconstruct the image. Successful communication emerges when the receiver can reconstruct the original image without losing the semantic information.

3.1.2 Communication Pathway 2: Shared Language

As shown in Figure 3, each agent perceives the images and generates a sequence of symbols. For instance, if two humans look at the digit '9' image, they may use the symbols '9' or 'nine' or 'IX' etc. The agents may develop a shared language that may look dissimilar to our natural language. Successful communication emerges when both agents generate similar symbolic representations.

3.2 Agent Architecture

Our hypothesis is that languages evolved by paying attention to various visual features. We introduce a Transformer-based agent with skills to generate text based on visual stimuli and to generate images based on text stimuli. With this skill the agent is able to share the message among each other various symbols based on the stimuli present.

As shown in Figure 1, we designed the agent as transformer-based encoder-decoder architecture with a discrete latent space.

3.2.1 Transformer Encoder

The encoder transforms the given image into a sequence of symbols. Thus, the agent can speak about what it sees in the world.

Unless we state otherwise, all the variables X, W, etc., in the encoding process are real-continuous variables, that is $(X,W) \in \mathbb{R}$. As shown below, we follow the feature transformation of the encoder from image to symbols.

- We transform the image as a sequence of pixels.
- We add learned positional embedding to the sequence of pixels.

$$X^{BHW} = X^{BHW} + \text{POSITIONAL\_EMBEDDING}$$

- We linearly transform the features into fixed hidden dimension size M.

$$X^{BM} = X^{BHW}W$$

- We apply dropout to improve learning. (optional)

$$X^{BM} = Dropout(X^{BM}, 0.5)$$

- We apply multi-headed attention.

$$X^{BM} = MultiHeadedAttention(X^{BM})$$

- Finally, we apply layer normalization and linear transformation to generate the symbolic message.

$$X^{BM} = LayerNorm(X^{BM}, 0.5)$$
$$X^{BLC} = X^{BM}W$$

3.2.2 Transformer Decoder

The decoder transforms a sequence of symbols into a specific object. That is, the agent can generate images using the symbols.

- We add learned positional embedding to the sequence of symbols.

$$X^{BLC} = X^{BLC} + \text{POSITIONAL\_EMBEDDING}$$

- We transform the message into fixed feature dimension M.

$$X^{BM} = X^{BLC}W$$

- We apply multi-headed attention.

$$X^{BM} = MultiHeadedAttention(X^{BM})$$

- We apply layer normalization and linear transformation to features. Finally, we use the Sigmoid function to generate the image.

$$X^{BM} = LayerNorm(X^{BM}, 0.5)$$
$$X^{BHW} = X^{BM}W$$
$$X^{BHW} = sigmoid(X^{BHW})$$

3.2.3 Gumbel SoftMax

We used Gumbel SoftMax distribution to overcome the challenges concerning backpropagating through discrete latent dimensions.

Traditional variational auto-encoders learn the data distribution (mean, variance) through the reparameterization trick. During training, we draw the samples from the data distribution learned parameters (mean, variance) and some Gaussian random noise. In our problem, we want to backpropagate through a set of stochastic neurons, representing the hidden representation. Ideally, we want the hidden representation to express a symbolic message. For this purpose, we use Gumbel-SoftMax distribution, also known as Concrete distribution. During training, the temperature of Gumbel-SoftMax distribution is allowed to anneal from 1 to close to 0. Slow annealing allows the distribution to collapse to a categorical distribution gradually. We predefined K categories, that is, the symbol sequence length. Thus, we learn the symbols using the Gumbel SoftMax trick that enables indirect differentiation through the discrete space.

3.3 Training Procedure

1. Get a batch of images, say X.
2. Generate batch of symbols S_i using an agent i encoder: $S_i = encoder_i(X)$.
3. Reconstruct the images using the agent's decoder: $\tilde{X} = decoder_i(S_i)$.
4. Calculate the reconstruction loss as a binary cross-entropy loss between the original and the reconstructed images: $L_{BCE} = \text{binary-cross-entropy}(\tilde{X}, X)$.
5. Calculate the shared language loss as the mean of the squared error between both agent's symbols: $L_{MSE} = \text{mean-squared-error}(S_1, S_2)$.

6. As shown in Figure 4, we calculate the combined loss by adding reconstruction loss and shared language loss: $L_{combined} = L_{MSE} + L_{BCE}$.
7. Repeat the above till a fixed number of iterations or the combined loss goes below a certain predefined threshold.

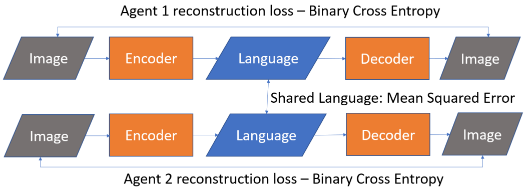
Agent 1 reconstruction loss – Binary Cross Entropy
Image
Encoder
Language
Decoder
Image
Shared Language: Mean Squared Error
Image
Encoder
Language
Decoder
Image
Agent 2 reconstruction loss – Binary Cross Entropy

Figure 4: Each agent minimizes the binary cross entropy loss between the recreated and the original image. For shared language to emerge, we minimize the mean squared error between two agent's languages.

3.4 World Model

To evaluate the reconstructed images of the agents, we created a convolution-based classifier that acts as a World Model. Detailed architecture of the World Model is shown in Figure 5. The World Model receives a batch of images as input, passes through the convolutional neural network, and outputs the classification of the digits.

World Model classifies the agents reconstructed images. If the World Model can classify an agent-reconstructed image correctly, we can learn that the agent can communicate meaningful semantic information through its symbols. Because we train the World Model independently, we obtain an unbiased performance estimate of the agent.

4 EXPERIMENTS

In all our experiments, we restrict the number of symbols allowed to 2 (i.e., 0 or 1) and the communication sequence length to 12. Thus, we use the terms bits and symbols interchangeably.

4.1 General Configuration Settings

Following the guidelines of the reproducible experiment, we fix the random seed of pytorch, cudnn, and numpy. We initialize the Gumbel SoftMax function with a temperature of 1. We gradually decrease the temperature at a rate of 0.00003 per iteration. We turn off the categorical representation by setting hard to False during training. We turn on the categorical representation by setting hard to True during the test. In the transformer architecture, we fixed the number of multi-headed attention blocks to 6 and the number of heads to 8. We fixed the training and test batch sizes to 100. We repeat the training procedure for 200 epochs.

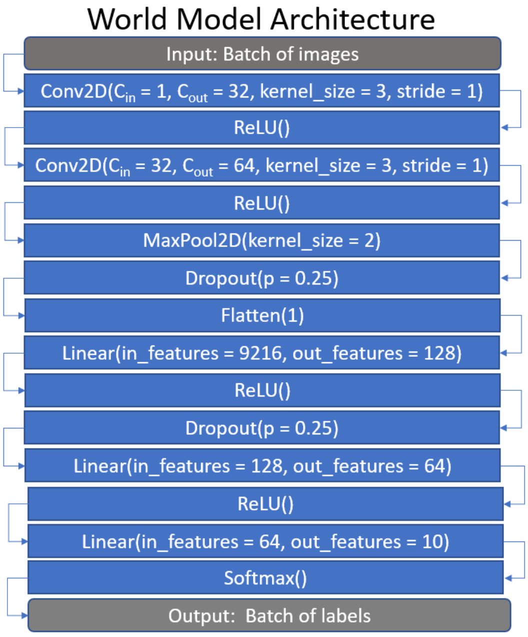
World Model Architecture
Input: Batch of images
Conv2D(C_{in} = 1, C_{out} = 32, kernel_size = 3, stride = 1)
ReLU()
Conv2D(C_{in} = 32, C_{out} = 64, kernel_size = 3, stride = 1)
ReLU()
MaxPool2D(kernel_size = 2)
Dropout(p = 0.25)
Flatten(1)
Linear(in_features = 9216, out_features = 128)
ReLU()
Dropout(p = 0.25)
Linear(in_features = 128, out_features = 64)
ReLU()
Linear(in_features = 64, out_features = 10)
Softmax()
Output: Batch of labels

Figure 5: World Model architecture is an MNIST convolutional neural network classifier.

4.2 Evaluation Metrics

The agents communicate with each other by sharing generated images and symbols. To assess the agent reconstructed image, we use F1 metric (Davis and Goadrich, 2006) and structural similarity index measure (Wang et al., 2003). We evaluate the shared artificial language using the Levenshtein distance.

F1 Metric F1 Score in Equation 1 is the harmonic mean of precision and recall.

$$\text{f1-score} = 2 * \frac{precision * recall}{precision + recall} \qquad (1)$$

Structural Similarity Index Measure SSIM measures the structural difference between any two given images. SSIM values range from 0 to 1, where near 0 values indicate that the two images are not structurally similar. The SSIM value near 1 indicates that the two images are structurally similar. We compare SSIM between the test image and the agent-recreated test image. SSIM metric complements the World Model classification metrics in quantitatively expressing how

Table 1: We measure the World Model F1 score and the SSIM on the receiver agent reconstructed images under different communication pathways. If there is no communication between agents or when the agents share messages alone, agents cannot reconstruct an image using each other's symbols. Thus, the reconstructed images have a low F1 score and SSIM. When the receiver agent learns to reconstruct a sender-perceived image, the reconstructed images achieve an F1 score of circa 0.86 and an SSIM of 0.60.

Communication\Metrics	**F1 Score**	**SSIM**
No Communication	0.10	0.09
Shared Reconstruction	0.87	0.60
Shared Message	0.11	0.09
Shared Reconstruction and Message	0.86	0.60

Table 2: We measured the Levenshtein distance between two agents under different communication pathways. When there is no communication between agents or when the agents share only messages, agents do not develop a shared language as the Levenshtein distance between the sender and receiver agents is longer than 2 for all the test images. When the sender and receiver agents learn to reconstruct the images using each other symbols, the sender and receiver develop a shared language as the distance between the symbols is less than 2 for the majority of the test images.

Communication\Edit Distance	**0**	**1**	**2**	**3**	**≥4**
No Communication	0	0	2	8	90
Shared Reconstruction	31	36	29	3	1
Shared Message	0	1	2	13	84
Shared Reconstruction and Message	28	43	23	6	0

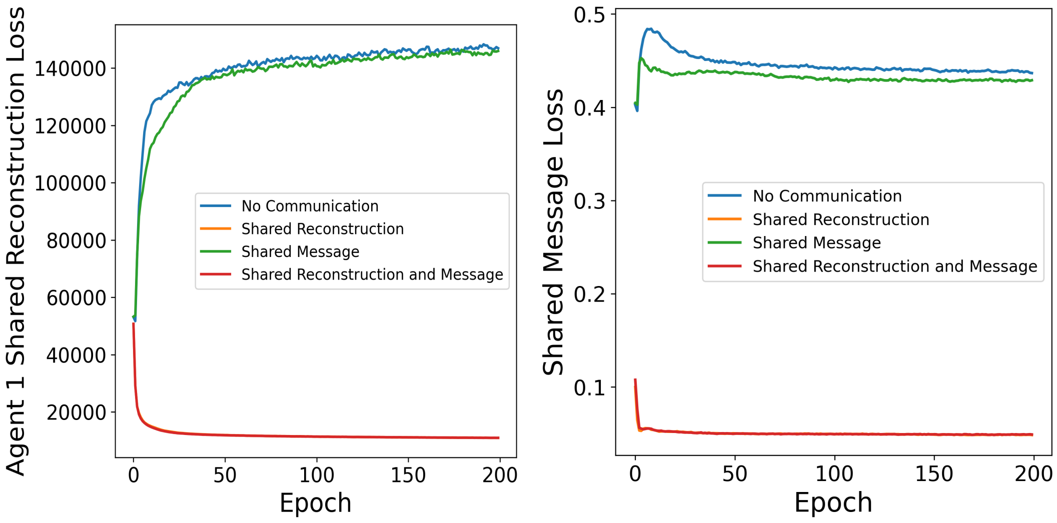

Figure 6: Left: Agent 1 shared reconstruction loss measures the binary cross entropy loss between Agent 2 perceived image and Agent 1 reconstructed image using the symbols. Right: Shared message loss measures the mean squared error between two agent representations of the digits in the image. Throughout the training, the agent shared reconstruction and shared message loss increase even if the agents optimize to generate a similar semantic mapping. The loss of the agents when they share the symbols alone is comparable to the agent loss that does not communicate at all. Both shared reconstruction loss and shared message loss converge when we train each agent to reconstruct the image using the other agent's symbols.

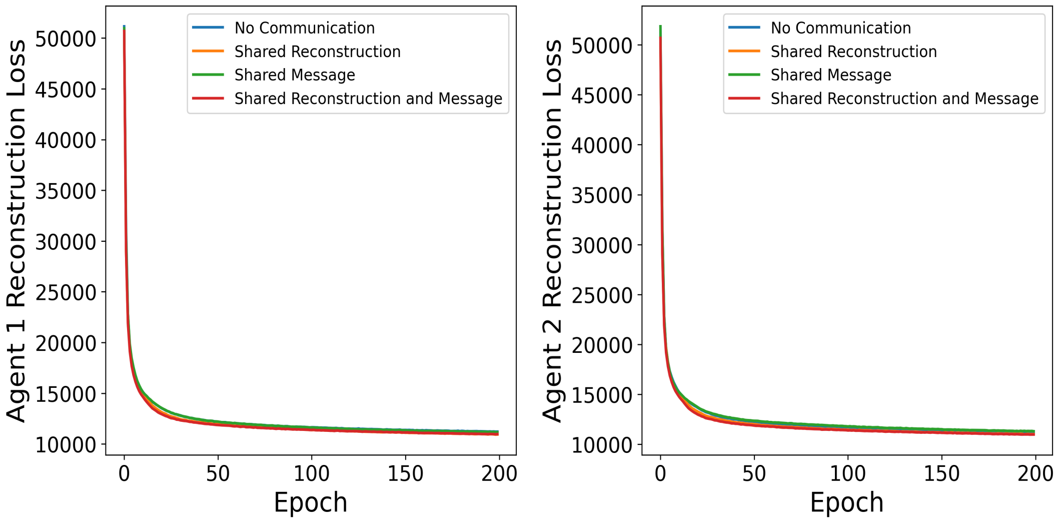

Figure 7: Left: Agent 1 reconstruction loss measure the binary cross entropy loss between the perceived and the reconstructed image using the symbols. Right: Agent 2 reconstruction loss measure the binary cross entropy loss between the perceived and the reconstructed image using the symbols. We measured the reconstruction loss for each agent under different communication pathways. Our results show that the internal perception of the agents remains consistent throughout training.

far the agent captures the semantic and structural information.

Levenshtein Distance Metric Levenshtein distance metric (Levenshtein et al., 1966; Hyyrö, 2001) measures the least number of inserts, delete, and substitute operations required to change one string into another string. Levenshtein metric allows us to measure how far apart the languages developed by each agent in a multi-agent communication framework.

4.3 Communication Pathways

To answer our research questions, we created four experiments. Each experiment studies the learning behavior of the agents under four scenarios. For each scenario, we measure the shared reconstruction loss as the binary cross entropy loss between the receiver-generated image and the perceived image of the sender agent. We measure the shared message loss as the mean squared error between the first and the second agent symbols.

1. No Communication: We train the agents to reconstruct images using their language. Thus, a classic encoder-decoder setting with a binary bottleneck layer.
2. Shared Reconstruction: We train the agents to reconstruct images using their language. Then, each agent reconstructs the image using the symbols of another agent. The objective function minimizes the shared reconstruction loss.
3. Shared Message: We train the agents to reconstruct the images using their language. The objective function minimizes the shared language loss.
4. Shared Reconstruction and Shared Message: We train the agents to reconstruct images using their language. Then, each agent reconstructs the image using the symbols of another agent. The objective function minimizes the shared reconstruction loss and shared message loss.

As shown in Figure 6, the shared language loss goes down if and only if the shared reconstruction loss goes down. As shown in Table 1, the agent-generated images perform worse on the test set if we do not optimize for the shared reconstruction loss. The World Model accuracy is circa 86, and SSIM is 0.60 only when the model optimizes with the shared reconstruction loss. As shown in Table 2, the agents generate similar symbols if the receiver agent can reconstruct the sender images. To answer our first research question, yes, the receiver agent can successfully reconstruct an image perceived by the sender only when the agents learn the symbols concerning the sender's image.

Even optimizing specifically for shared language does not result in language emergence and follows the loss trajectory of agents that do not communicate. A similar effect is observed in the test set as shown in Table 2. There exist no similar symbols even when optimizing for shared messages. To answer our second research question, no, agents sharing only the symbols do not develop a shared language. Without

Table 3: Mean loss of the last 50 epochs of different communication pathways repeated for ten random iterations. Overall, the shared reconstruction loss and the shared message loss do not show more deviation. The results suggest that language emergence in the first communication pathway and no language emergence in the second communication pathway are not due to random variations in the training procedure.

Pathways\Metrics	**Agent 1 Shared Reconstruction Loss**	**Shared Message Loss**
No Communication	137019.29 +/- 9275.42	0.42 +/- 0.01
Shared Reconstruction	11159.43 +/- 63.15	0.05 +/- 0.00
Shared Message	138954.43 +/- 15739.21	0.43 +/- 0.03
Shared Reconstruction and Message	11538.64 +/- 0.0	0.08 +/- 0.00

Table 4: We conducted ablation studies on different architecture choices for the agents. We report the World Model F1 score and the SSIM on the agent-reconstructed images on the test set. The transformer-based autoencoder architecture outperformed the convolution-based and fully connected network-based auto-encoder architecture in all metrics.

Metric\Architecture	**Convolution**	**Fully Connected**	**Transformer**
F1 Score	0.78	0.83	**0.88**
SSIM	0.40	0.46	**0.61**
Sequence Length	30	30	**12**
Number of Symbols	10	2	**2**

intention, the language can not emerge between two agents. Optimizing the agents concerning symbols but ignoring what those symbols represent results in poor language learning.

Moreover, each agent has an internal perception of the world. As shown in Figure 7, we measured the reconstruction loss for each agent under different communication pathways. Our results show that the internal perception of the agents remains consistent throughout training.

To better obtain confidence in the results, we repeated the experiment for ten random iterations for each communication pathway, and the loss measure is shown in Table 3. There is less variation when there is no communication between agents and when the agents share only the symbols.

4.3.1 Ablation Studies

We created a fully connected network-based and convolution-based auto-encoder to benchmark against our transformer-based agent architecture. The architecture of fully connected network-based agent architecture is shown in the Table 5 and 6 in the Appendix. The convolution-based agent architecture is shown in the Table 7 and 8 in the Appendix.

We increased the number of symbols and sequence length because the lower number of symbols results in poor representation in fully connected and convolution-based agents. As shown in Table 4, the transformer-based network has learned better symbolic representation and has a higher F1 score and SSIM.

5 CONCLUSIONS

In this paper, we have introduced a framework for studying the communication between transformer-based deep learning agents. In our experiments, we could observe the emergence of a shared language between the agents if the receiver successfully reconstructs the image using the message from the sender agent. We have shown that two agents can develop a shared language within the Levenshtein distance of 2. The reconstructed images achieve a 0.87 F1-score obtained from an independent World Model. We found that agents focusing only on increasing the similarity of symbols do not result in the emergence of language.

In the current stage, we see several limitations with our studies, which we will aim to address in future work: First, we have only studied the framework concerning agents cooperating to develop a shared language, neglecting potential other relations between models. Next, we assume agents to have a fixed vocabulary throughout the training and test times. Natural language is variable in sequence length. Finally, we experimented with only one pair of agents. An avenue for future work would be to expand the framework to many agents interacting with each other.

REFERENCES

Batali, J. (1998). Computational simulations of the emergence of grammar. *Approaches to the Evolution of Language-Social and Cognitive Bases-*.

Brandizzi, N. (2023). Towards More Human-like AI Communication: A Review of Emergent Communication

Research. *arXiv e-prints*, page arXiv:2308.02541.

Chaabouni, R., Kharitonov, E., Lazaric, A., Dupoux, E., and Baroni, M. (2019). Word-order biases in deep-agent emergent communication.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240.

Dubois, Y., Li, X., Taori, R., Zhang, T., Gulrajani, I., Ba, J., Guestrin, C., Liang, P., and Hashimoto, T. B. (2023). Alpacafarm: A simulation framework for methods that learn from human feedback.

Evtimova, K., Drozdov, A., Kiela, D., and Cho, K. (2018). Emergent communication in a multi-modal, multi-step referential game.

Hyyrö, H. (2001). Explaining and extending the bit-parallel approximate string matching algorithm of myers. Technical report, Citeseer.

Jang, E., Gu, S., and Poole, B. (2017). Categorical reparameterization with gumbel-softmax.

Jinxin, S., Jiabao, Z., Yilei, W., Xingjiao, W., Jiawen, L., and Liang, H. (2023). CGMI: Configurable General Multi-Agent Interaction Framework. *arXiv e-prints*, page arXiv:2308.12503.

Kottur, S., Moura, J. M., Lee, S., and Batra, D. (2017). Natural language does not emerge'naturally'in multi-agent dialog. *arXiv preprint arXiv:1706.08502*.

Lazaridou, A. and Baroni, M. (2020). Emergent multi-agent communication in the deep learning era. *arXiv preprint arXiv:2006.02419*.

Levenshtein, V. I. et al. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.

Li, F. and Bowling, M. (2019). Ease-of-teaching and language structure from emergent communication.

Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., and Ghanem, B. (2023). Camel: Communicative agents for "mind" exploration of large scale language model society.

Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. (2023). Generative agents: Interactive simulacra of human behavior.

Ren, Y., Guo, S., Labeau, M., Cohen, S. B., and Kirby, S. (2020). Compositional languages emerge in a neural iterated learning model.

Ri, R., Ueda, R., and Naradowsky, J. (2023). Emergent communication with attention.

Rita, M., Strub, F., Grill, J.-B., Pietquin, O., and Dupoux, E. (2022). On the role of population heterogeneity in emergent communication.

Słowik, A., Gupta, A., Hamilton, W. L., Jamnik, M., Holden, S. B., and Pal, C. (2020). Structural Inductive Biases in Emergent Communication. *arXiv e-prints*, page arXiv:2002.01335.

Tishby, N., Pereira, F. C., and Bialek, W. (2000). The information bottleneck method.

Turpin, M., Michael, J., Perez, E., and Bowman, S. R. (2023). Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *arXiv preprint arXiv:2305.04388*.

Wang, Z., Simoncelli, E. P., and Bovik, A. C. (2003). Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A. H., White, R. W., Burger, D., and Wang, C. (2023). Autogen: Enabling next-gen llm applications via multi-agent conversation.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Zhang, S., Zhu, E., Li, B., Jiang, L., Zhang, X., and Wang, C. (2023). AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation Framework. *arXiv e-prints*, page arXiv:2308.08155.

APPENDIX

5.1 Fully Connected Network-based Architecture

We created a simple fully connected network-based auto-encoder architecture for the agents. As shown in Table 5, the encoder agents receive an image with a fixed image height and width. The encoder then transforms the features into a series of fully connected layers with a fixed hidden dimension size specified in hidden_dim. Finally, the encoder outputs the sequence of symbols via the Gumbel SoftMax layer.

As shown in Table 6, the decoder receives a sequence of symbols and outputs the image after passing through a series of fully connected layers with a fixed hidden dimension of size hidden_dim.

5.2 Convolution-based Architecture

We created a convolution-based auto-encoder architecture for the agents. As shown in Table 7, the encoder agents receive an image with a fixed image height and width. The first two convolutional layers extract the features in the image using 2D convolution with a kernel size of 3. Next, we apply the MaxPool2D layer with a kernel size of 2. During training, the dropout function on the output feature with a probability of 0.25. We flatten the features before applying the fully connected layer. Next, we apply two fully connected layers with an optional dropout layer. Finally, the encoder outputs the sequence of symbols via the Gumbel SoftMax layer.

As shown in Table 8, The decoder receives a sequence of symbols and outputs the image after passing through a series of fully connected layers with a

Table 5: We created a fully connected network-based encoder to test various choices for the multi-agent communication framework. '-' denotes that there are no trainable parameters in that layer. The height and width represent the image size, and hidden_dim is the hidden dimension size. laten_dim and categorical_dim represent the sequence length and the number of symbols.

Layer	**Input**	**Output**
Linear	image: height*width	hidden_dim
ReLU	-	-
Linear	hidden_dim	hidden_dim
ReLU	-	-
Linear	hidden_dim	laten_dim*categorical_dim
Gumbel SoftMax	laten_dim*categorical_dim	language: sequence of symbols

Table 6: We created a fully connected network-based decoder to test various choices for the multi-agent communication framework. '-' denote that there are no trainable parameters in that layer. The height and width represent the image size, and hidden_dim is the hidden dimension size.

Layer	**Input**	**Output**
Linear	language: sequence of symbols	hidden_dim
ReLU	-	-
Linear	hidden_dim	hidden_dim
ReLU	-	-
Linear	hidden_dim	image: height*width

Table 7: We created a Convolution-based agent encoder architecture to test various choices for the multi-agent communication framework. '-' denotes that there are no trainable parameters in that layer. The height and width represent the image size, and hidden_dim is the hidden dimension size. latent_dim and categorical_dim represent the sequence length and the number of symbols.

Layer	**Input**	**Output**
Conv2D	image: height*width	32
ReLU	-	-
Conv2D	32	64
ReLU	-	-
MaxPool2D	-	-
Dropout	-	-
Flatten	-	9216
ReLU	-	-
Linear	9216	128
ReLU	-	-
Dropout	-	-
Linear	128	64
ReLU	-	-
Linear	64	laten_dim*categorical_dim
Gumbel SoftMax	laten_dim*categorical_dim	language: sequence of symbols

fixed hidden dimension of size hidden_dim and 9216.

Table 8: We created a fully connected network-based agent to test various choices for the multi-agent communication framework. '-' denotes that there are no trainable parameters in that layer. The height and width represent the image size, and hidden_dim is the hidden dimension size. latent_dim and categorical_dim represent the sequence length and the number of symbols.

Layer	**Input**	**Output**
Linear	language: sequence of symbols	hidden_dim
ReLU	-	-
Linear	hidden_dim	hidden_dim
ReLU	-	-
Linear	hidden_dim	9216
ReLU	-	-
Linear	9216	image: height*width