It's About Time: Temporal References in Emergent Communication

OLAF LIPINSKI*, University of Southampton, United Kingdom
ADAM J. SOBEY, The Alan Turing Institute, United Kingdom and University of Southampton, United Kingdom
FEDERICO CERUTTI, University of Brescia, Italy
TIMOTHY J. NORMAN, University of Southampton, United Kingdom

Emergent communication enables agents to develop bespoke languages that improve communication efficiency. Despite the known importance of temporal structure in natural language, there is no existing evidence of temporal references in emergent communication. This paper addresses this gap, by exploring how agents communicate about temporal relationships. We analyse three potential factors for the emergence of temporal references: environmental, external, and architectural. Our experiments demonstrate that altering the loss function is insufficient for temporal references to emerge; rather, architectural changes are necessary. A minimal change in agent architecture, using a different batching method, allows the emergence of temporal references. This modified design is compared with the standard architecture in a temporal referential games environment, which emphasises temporal relationships. The analysis shows that over 95% of the agents with the modified batching method develop temporal references, without changes to their loss function. We consider temporal referencing necessary for future improvements to the agents' communication efficiency, enabling future agents to use a closer to optimal coding as compared to purely compositional languages. These insights provide the basis for incorporation of temporal references into other emergent communication settings, and investigation of other aspects of language.

JAIR Associate Editor: Davide Grossi

JAIR Reference Format:
Olaf Lipinski, Adam J. Sobey, Federico Cerutti, and Timothy J. Norman. 2026. It's About Time: Temporal References in Emergent Communication. *Journal of Artificial Intelligence Research* 86, Article 11 (June 2026), 24 pages. DOI: 10.1613/jair.1.19795

1 Introduction

In emergent communication, autonomous agents develop their language from scratch. This contrasts with formally defined languages (Vieira et al. 2007), where the agents are prescribed what and how to communicate (Rochlin and Sarne 2015). The emergent communication approach results in a vocabulary that is tailored to the specific environment in which the agents have been trained, reflecting the tasks the agents perform, the actions available to them and the other agents they interact with. These properties make the emergent language memory and bandwidth efficient, as the agents can optimise their vocabulary size and word length to their specific task, providing an advantage over a general communication protocol (Rita, Chaabouni, et al. 2020). This approach also allows for inherent vocabulary alignment (Chocron and Schorlemmer 2020), even allowing for alignment

*Corresponding Author.

Authors' Contact Information: Olaf Lipinski, o.lipinski@soton.ac.uk, ORCID: 0000-0002-2023-7617, University of Southampton, Southampton, United Kingdom; Adam J. Sobey, ajs502@soton.ac.uk, ORCID: 0000-0001-6880-8338, The Alan Turing Institute, London, United Kingdom and University of Southampton, Southampton, United Kingdom; Federico Cerutti, federico.cerutti@unibs.it, ORCID: 0000-0003-0755-0358, University of Brescia, Brescia, Italy; Timothy J. Norman, t.j.norman@soton.ac.uk, ORCID: 0000-0002-6387-4034, University of Southampton, Southampton, United Kingdom.

DOI: 10.1613/jair.1.19795

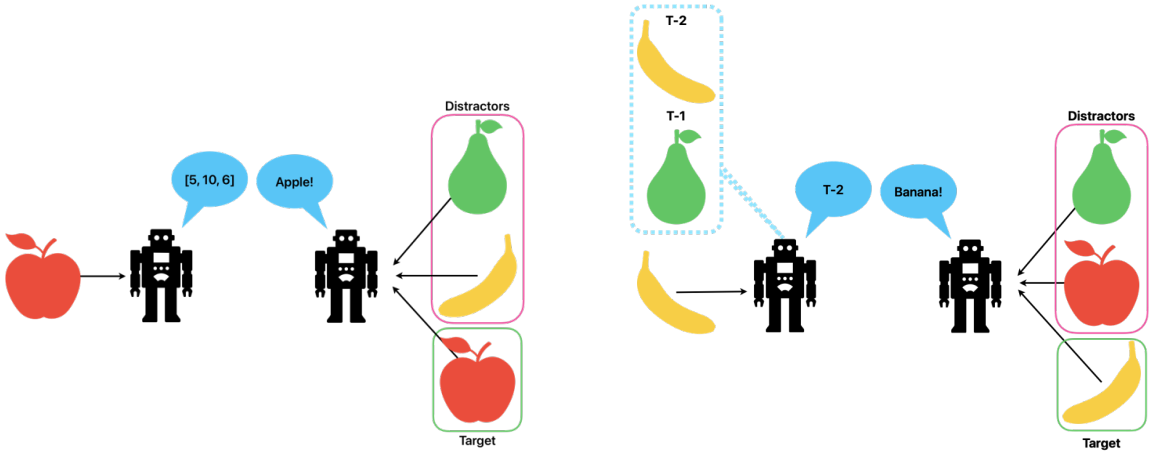
between agents which were not trained together through population approaches (Michel et al. 2022; Rita, Michel, et al. 2024; Rita, Strub, et al. 2022), enabling convergence to a common language across a population of agents.

Many aspects of emergent language have been explored (Boldt and Mortensen 2024; Lazaridou and Baroni 2020), with a particular focus on improving communication efficiency (Chaabouni, Kharitonov, Dupoux, et al. 2019; Kang et al. 2020; Rita, Chaabouni, et al. 2020). Kang et al. (2020) demonstrate how using the minimal deviation between subsequent time steps allows for more concise communication by reducing redundant information transfer. Investigation of the contextual information of the resulting language offers a further improvement in agent performance by using the time step similarity together with optimisation of the reconstruction of the speaker's state (Kang et al. 2020). There is, however, no existing research investigating or reporting on the emergence of temporal referencing strategies, where agents could communicate about relationships between different time steps.

Such temporal references, together with the general characteristics of emergent languages, have the potential to enhance the agents' bandwidth efficiency and task performance in a variety of situations. Temporal references are the only way that agents can coordinate in tasks that require temporal understanding. For example, in a turn-based strategy game an agent might need to refer back to "the opponent's move two turns ago" to communicate past behaviour; in multi-robot monitoring one drone could report "the sensor reading from three scans ago" to its partner; in multi-agent coordination, agents must be able to schedule synchronisation times ("Sync next at t=10") to preserve bandwidth (Wang et al. 2023). The simplified environment used in our study provides a controlled and interpretable setting that allows us to systematically probe the specific factors driving the emergence of such temporal references. While this setting enables clearer insights into the underlying mechanisms, applying these findings to more complex, real-world scenarios, such as social deduction games (Brandizzi et al. 2021; Kopparapu et al. 2022; Lipinski et al. 2022), may introduce additional challenges due to the richer dynamics and strategic complexity involved. Nonetheless, our work offers a foundational step toward understanding how temporal communication strategies could emerge in such environments.

Temporal references would also allow agents to develop more efficient methods of communication by assigning shorter messages to events that happen more often. This is similar to Zipf's Law in human languages (Zipf 1949), which states that the most commonly used words are the shortest. This strategy would be particularly effective when the distribution of observations would be non-uniform, which means that certain objects appear more often than others. Specialised messages, used only for temporal references, would then also become more frequent than others. From information theory, we know that (adaptive) Huffman coding (Huffman 1952; Knuth 1985; Vitter 1987) can assign shorter bit sequences to more frequent messages, thereby compressing them more efficiently than less common messages. Consequently, the incorporation of temporal references can enhance the efficiency of transmitting emergent language, optimising communication.

Our contribution lies in examining when temporal references emerge between agents. In this work we perform a focused study of three potential factors of temporal references — environmental pressures, external losses, and architectural biases — so as to pinpoint their individual effects without conflating other factors. The agents are trained in both the regular referential game (Lazaridou, Peysakhovich, et al. 2017) and on an environment which encourages the development of temporal references through embedded environmental pressures (Section 2.3). The effect of an external pressure to develop temporal referencing is explored via an additional loss applied to the agents (Section 5). Three types of architecture are evaluated (Section 3). The baseline *Base* (Section 3.1) agent, based on the commonly used EGG agents (Kharitonov et al. 2019), provides us with a reference performance for both the emergence of temporal references, and performance in an environment. This baseline is compared to a *Temporal* (Section 3.2) agent, which features a sequentially batched LSTM, instead of the parallel batching used in EGG, which allows the agents to build an understanding of the target sequence. Additionally, the *TemporalR* (Section 3.3) agent combines the information from the sequential LSTM and the parallel batched LSTM from the



(a) Referential Game (b) Temporal Referential Game

Fig. 1. Structure of the referential game and temporal referential game.

Base agent. This allows it to process information about the objects, without needing to focus on their order in the sequence at the same time.

2 Referential Game Environments

Referential games (Lazaridou, Hermann, et al. 2018; Lewis 1969) provide the most commonly used framework to study emergent communication, employing two agents: a sender and a receiver (Kharitonov et al. 2019). The sender begins the game by observing a target object, and then generates a message. This message is passed to the receiver, along with the target object and a number of distractor objects. The receiver's task is to discern the target object from among the objects it observes, using the information contained in the message it receives. This exchange is repeated every episode. The referential game is presented in Figure 1a. For comprehensive reviews of these environments, we refer to Lazaridou and Baroni (2020) or Brandizzi (2023).

In our referential games, we use attribute-value vectors to represent objects, enabling us to isolate and control factors that might impact agent performance. This approach (Chaabouni, Kharitonov, Bouchacourt, et al. 2020; Kharitonov et al. 2019; Ueda et al. 2022) serves as an effective test-bed for investigating temporal properties of emergent language.

2.1 Definitions

In our referential games, agents identify objects from an *object space* V, represented as attribute-value vectors $\boldsymbol{x} \in V$. The *value space* $S = \{0, 1, 2 \dots N_{val}\}$ defines possible variations for each attribute, where N_{val} is the *number of values*. The *object space* is defined as $V = S_1 \times \cdots \times S_N = \{(a_1, \dots, a_{N_{att}}) \mid a_i \in S_i \text{ for every } i \in \{1, \dots, N_{att}\}\}$, where N_{att} is the *number of attributes*.

For intuition, consider an object as an abstraction of an image of a circle. Attributes might include line style (dashed/solid), line colour, and background colour. Values represent variations of these attributes (e.g., blue, red, or black for colour). A blue solid-line circle on a red background could be represented as [blue, solid, red] or as integers $[2, 1, 3]$.

The characters available to the agents (*i.e., the symbol space*) is $\omega = \{0, 1, 2 \dots N_{vocab} - 1\}$ where N_{vocab} is the *vocabulary size*. The *message space*, or the space that all messages must belong to, is defined as $\xi = \omega_1 \times \cdots \times \omega_L =$

$\{(c_1, \ldots, c_L) \mid c_i \in \omega_i \text{ for every } i \in \{1, \ldots, L\}\}$, where L is the maximum message length. The message space ξ represents all possible symbol combinations that could constitute a message.

The agents' language maps objects in V to messages in ξ. The exchange history is a sequence $\tau = \{(\boldsymbol{m}_n, \boldsymbol{x}_n)\}_{n \in \{1,\ldots,t\}}$ such that $\forall n, \boldsymbol{m}_n \in \xi \wedge \boldsymbol{x}_n \in V$, with t representing the episode of the last exchange.

2.2 Temporal Logic

We use temporal logic to formally define our environment's behaviour and analyse agent communication. Specifically, we employ a form of Linear Temporal Logic (LTL) (Pnueli 1977) called Past LTL (PLTL) (Lichtenstein et al. 1985).

While LTL focuses on future-present connections using operators like "next" $\bigcirc$, PLTL extends this to include past relationships through operators like "previously" $\ominus$. The "previously" operator must satisfy Equation (1) (Maler et al. 2008), where σ refers to an agent's message at time t, and ϕ signifies the object seen by the agent.

$$(\sigma, t) \models \ominus\phi \leftrightarrow (\sigma, t-1) \models \phi \tag{1}$$

We use the shorthand notation $\ominus^n$ to indicate the $\ominus$ operator applied n episodes back (*e.g.*, $\ominus^4\phi \leftrightarrow \ominus\ominus\ominus\ominus\phi$).

2.3 Temporal Referential Games

In our temporal referential game, the sender might observe either a new random object or a previously seen object from earlier episodes. For example, in a sequence of episodes, the objects shown to the sender might be $[\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{a}, \boldsymbol{c}, \boldsymbol{b}]$, where $\boldsymbol{a}$ and $\boldsymbol{b}$ repeat from previous episodes. This temporal dimension introduces the possibility for agents to develop communication that references objects across time, potentially using more efficient representations than describing each object from scratch.

This temporal version of the referential games (Lazaridou, Peysakhovich, et al. 2017; Lewis 1969) is based on the "previously" ($\ominus$) PLTL operator.[1] At every game step s_t, the sender agent is presented with an input object vector $\boldsymbol{x}$ generated by the function $X(t, c, h_v)$, with a random *chance* parameter c, the *previous horizon* value h_v, and the current episode t.

$$X(t, c, h_v) = \begin{cases} \boldsymbol{x} & c = 0 \\ \ominus^{h_v}\boldsymbol{x} = \tau_{t-h_v} & c = 1 \end{cases} \tag{2}$$

The *previous horizon* value is uniformly sampled, taking the value of any integer in the range $[1, h]$, where h is the *previous horizon* hyperparameter, to allow agents to develop temporal references of varying temporal horizons. The function $X(t, c, h_v)$ selects a target object to be presented to the sender using Equation (2), either generating a new random target object or using the old target object. This choice is facilitated using the *chance* parameter c, which is sampled from a Bernoulli distribution, parametrised by the repetition chance parameter p, *e.g.*, $p = 0.5$. If $c = 1$ a previous target object is used, and if $c = 0$ a new target object is generated. Both c and h_v are sampled every time a target object is generated.

For example, consider an episode at $t = 4$ with the sampled parameters $c = 1$ and $h_v = 2$. Suppose the agent has observed the following targets: $[\boldsymbol{j}, \boldsymbol{k}, \boldsymbol{l}]$. Given that $c = 1$, further to Equation (2), the $\ominus^2$ ($\ominus^{h_v}$) target is chosen. The target sequence becomes $[\boldsymbol{j}, \boldsymbol{k}, \boldsymbol{l}, \boldsymbol{k}]$, with the target $\boldsymbol{k}$ being repeated, as it was the second to last target. Now suppose that c was sampled to be $c = 0$ instead. Further to Equation (2), a random target $\boldsymbol{a}$ is generated, and the target sequence becomes $[\boldsymbol{j}, \boldsymbol{k}, \boldsymbol{l}, \boldsymbol{a}]$.

[1] Code is available at https://github.com/olipinski/TRG

This behaviour describes the environment *TRG*, which represents the base variant of temporal referential games, where targets are randomly generated with a p chance of repetition of a target from the previous horizon $[1, h]$. We additionally implemented five more environment variants:

TRG : Base temporal referential game with a chance p of object repetition.
TRG Hard : Same as TRG but with targets that differ from distractors in only one, randomly chosen attribute.
RG : Classic referential game.
RG Hard : Classic referential game with targets that differ from distractors in only one, randomly chosen attribute.
Always Same : A subset[2] of all possible targets repeated sequentially, used to verify consistent temporal messaging.
Never Same : No target repetition, used to verify if temporal messages are used only in temporal contexts, and a as sanity check.

In all datasets where repetition may occur, only the target object is repeated, while the distractor objects are randomly generated for each object set. Sample inputs and expected outputs for all environments are provided in Section C.1.

Since RG, RG Hard, and Never Same environments have minimal target repetitions (*cf.*, Section C, Figure 4), we do not expect temporal references to emerge in these settings. We use these environments to validate our results, and to provide a baseline performance on the environments usually used in emergent communication research (Boldt and Mortensen 2024).

3 Architectures

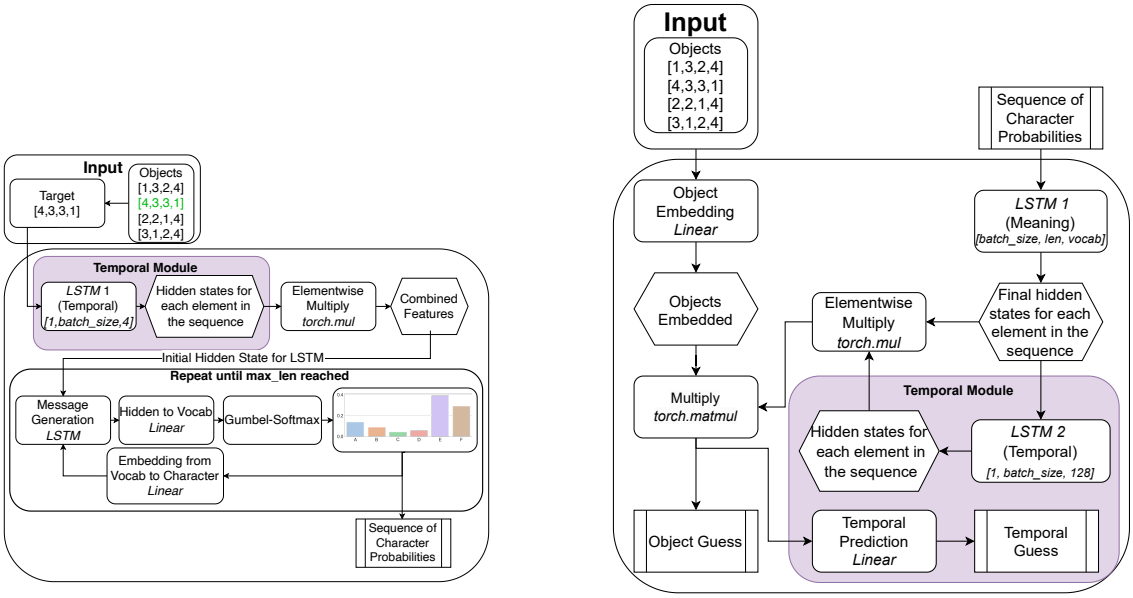
In emergent communication, both sender and receiver agents are typically built around a single recurrent neural network (Kharitonov et al. 2019). In this paper, these standard agent architectures, based on an LSTM (Hochreiter and Schmidhuber 1997), are compared to temporal LSTM architectures, which use a different batching strategy. The *Base* agent, used as a baseline, is the commonly used LSTM-based agent (Hochreiter and Schmidhuber 1997; Kharitonov et al. 2019). Our two novel architectures, the *Temporal* and *TemporalR* agents, feature a sequentially batched LSTM. This additional module allows the agents to gather information about the sequence of objects itself, instead of the regular LSTM which processes all objects in parallel. While the *Temporal* agents use just the sequential LSTM to process their input, the *TemporalR* agents combine both the *Base* and *Temporal* approaches, combining the outputs of a regular LSTM, together with the sequentially batched LSTM. This allows the *TemporalR* agents to process the information that may be present in the objects themselves, as well as the order of their appearances, without needing to store this information in a single LSTM hidden state.

While many architectures may work for processing temporal information within datasets, we opt for a distinct batching strategy for two key reasons. Firstly, it entails the least modification to the *Base* agent design, facilitating direct comparisons between the two architectures and enabling straightforward application of our architectural modifications to other contexts. Secondly, it offers a straightforward framework for examining the emergence of temporal references. Although our experiments initially involved more complex architectures, including attention-based agents, we observed no significant differences in any metrics from those of LSTM-based networks. Consequently, our focus in this study remains on LSTM-based agents.

3.1 *Base* Agent

In common with other approaches (Auersperger and Pecina 2022; Chaabouni, Kharitonov, Dupoux, et al. 2019; Kharitonov et al. 2019), each of the *Base* sender and receiver agents are constructed around a single LSTM (Hochreiter and Schmidhuber 1997). First, the sender's LSTM receives each target and distractor set individually,

[2]A subset is used as the target space grows exponentially with the number of attributes and values.



(a) The *Temporal* LSTM sender architecture.

(b) The *Temporal* LSTM receiver architecture.

Fig. 2. The *Temporal* LSTM sender and receiver architectures, with the temporal modules highlighted in purple.

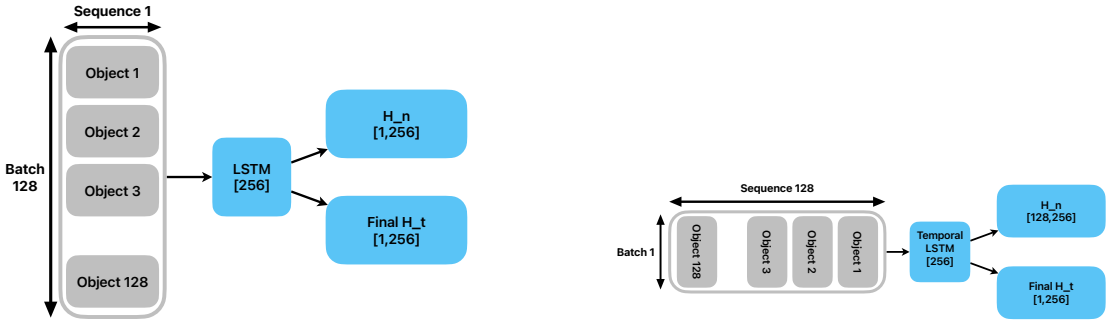
processing each object separately. The result is the initial hidden state for the message generation LSTM. For message generation, the same method is followed as used in previous work (Kharitonov et al. 2019), and messages are generated character by character, using the Gumbel-Softmax trick (Jang et al. 2017). These messages are then passed to the receiver. The receiver's architecture contains an object embedding linear layer and a message processing LSTM, similar to the most commonly used architecture (Kharitonov et al. 2019). In the receiver agent, a hidden state is computed for each message by the LSTM. This output is combined with the output of the object embedding linear layer to create the referential game object prediction. The architecture diagram for the *Base* agent is available in Section F, Figure 7.

3.2 *Temporal* Agent

The *Temporal* agent extends the *Base* architecture by introducing a sequential LSTM module in both the sender and receiver networks (Figure 2). This module enables the agents to capture temporal relationships across objects by processing sequences rather than treating each object as independent, similar to the sequential learning of language in humans (Christiansen and Kirby 2003). The module's aim is to allow the agents to develop a temporally grounded understanding of the input, supporting the emergence of temporal references.

In standard LSTM architectures (Kharitonov et al. 2019), each object is processed independently in parallel. This corresponds to input batches of shape, *e.g.*, $[128, 1, 6]$, where 128 is the batch size and each object has 6 attributes. In contrast, the *Temporal* agent uses a sequential batching strategy, changing the batch shape to $[1, 128, 6]$, or a single sequence of 128 objects. This allows the LSTM to process the objects in temporal order and develop a representation of the sequence as a whole. Visualisations of both approaches are shown in Figure 3.

In the sender (Figure 2a), the output of the sequential LSTM provides the initial hidden state for message generation. The message generation procedure follows the standard approach, where messages are produced



(a) Regular batching of an LSTM.

(b) Temporal batching of an LSTM.

Fig. 3. Examples of regular and temporal batching strategies.

character by character using the Gumbel-Softmax trick (Jang et al. 2017). These messages are then passed to the receiver.

The receiver (Figure 2b) mirrors the sender's design. It first processes the message using a standard LSTM to obtain a set of hidden states, then feeds these into a sequential LSTM. This additional LSTM captures temporal dependencies in the message structure and supports the identification of recurring objects. The resulting temporal representation is combined with object embeddings for referential prediction.

To further promote temporal reasoning, the receiver may include a temporal prediction module. This module computes a label representing how far back (in number of episodes) an object was last seen, up to a fixed previous horizon h. This label is predicted using a single linear layer. For instance, if an object last occurred 5 episodes ago and $h = 8$, the label is 5 since $5 \leq h$. If instead $h = 4$, the label defaults to 0, as the previous occurrence lies beyond the horizon.

Training includes an auxiliary loss term for this prediction, combined with the standard referential game loss. The total loss function is defined as $L_t = L_{rg} + L_{tp}$, where L_{rg} is the referential loss (cross-entropy between the receiver's guess and the sender's target), and L_{tp} is the temporal prediction loss (cross-entropy between the predicted and true temporal labels). This auxiliary objective helps focus learning on temporal relationships, encouraging the development of temporally grounded communication. Further analysis of its effect is presented in Section 5.

3.3 *TemporalR* Agent

The *TemporalR* agent combines the *Base* and the *Temporal* architectures. The sequential LSTM from the *Temporal* agent is added to the *Base* architecture, merging both the sequential understanding from the temporal module, with the parallel understanding of the target objects from the regularly batched LSTM. The hidden states of both LSTMs are combined through an element-wise multiplication and fed into the message generation LSTM. The receiver agent is the same as the *Temporal* receiver, and includes the temporal prediction module. The architecture diagram for the *TemporalR* agent is available in Section F, Figure 8.

4 Metrics

4.1 Temporality Metric

To evaluate the development of temporal references, we propose a new metric, $M_{\ominus^n}$, which measures how often a given message was used as the "previous" operator. Given an exchange history (the sequence of objects shown to the sender and messages sent to the receiver), τ, it checks when an object has been repeated within a given

horizon h_v, and records the corresponding message sent to describe that object. The objective of the $M_{\ominus^n}(\boldsymbol{m})$ metric is to measure if the message can give reference to a previous episode; *e.g.*, if a message is used similarly to the sentence "The car I can see is the same colour as the one mentioned two sentences ago".

Let $C_{\boldsymbol{m}}\ominus^n$ count the times the message $\boldsymbol{m}$ has been sent together with a repeated object for $h_v = n$:

$$C_{\boldsymbol{m}\ominus^n} = \sum_{j=1}^{t} \mathbb{I}(\boldsymbol{m}_j = \boldsymbol{m} \wedge \text{objectSame}(\boldsymbol{x}_j, n)) \tag{3}$$

where $\mathbb{I}(\cdot)$ is an indicator function that returns 1 if the condition is true and 0 otherwise, and $\text{objectSame}(\boldsymbol{x}_j, n)$ is a function that evaluates to true if the object $\boldsymbol{x}_j$ is the same as the object n episodes ago.

Let $\mathcal{T}_{\boldsymbol{m}}$ denote the total count of times the message $\boldsymbol{m}$ has been used:

$$\mathcal{T}_{\boldsymbol{m}} = \sum_{j=1}^{t} \mathbb{I}(\boldsymbol{m}_j = \boldsymbol{m}) \tag{4}$$

where $\mathbb{I}(\cdot)$ is an indicator function selecting the message $\boldsymbol{m}$ in the exchange history τ.

The percentage of previous messages that are the same as $\boldsymbol{m}$ can then be calculated using $M_{\ominus^n}(\boldsymbol{m})$:

$$M_{\ominus^n}(\boldsymbol{m}) = \frac{C_{\boldsymbol{m}}\ominus^n}{\mathcal{T}_{\boldsymbol{m}}} \times 100 \tag{5}$$

The $M_{\ominus^n}$ metric is sensitive to repeated object appearances due to dataset structure, which may lead to false positives in environments with high repetition probability. If a language does not have temporal references and uses a given message to describe an object consistently, this message will be repeated every time this object appears. This means that for every repetition object, the message could be considered to indicate a *previous* episode, whereas, in reality, it is just a description of the object. For example, if a dataset contains 75% repetitions, on average, each message will be used as an accidental $\ominus^n$ 75% of the time.

Therefore, in Section 5, the emergence of temporal references is defined as the appearance of messages that reach 100% on the $M_{\ominus^n}$ metric. This strict threshold helps avoid ambiguity: a message used 30% or 70% of the time in a temporal context might reflect genuine referencing, but it could just as easily result from consistent object labelling or a high repetition rate in the dataset. Simply comparing against the base repetition rate in the dataset is also not enough; while the chance of repetition can be estimated (and is close to 0% in RG environments, as shown in Section C), this does not help distinguish between intentional temporal reference and accidental reuse of a message. Without additional heuristics or manual inspection, it would be difficult to differentiate the false positives from the true positives. By focusing only on messages that appear exclusively in reference to previous episodes, *i.e.*, those with a perfect 100% score, we ensure that the signal is both interpretable and robust.

Importantly, even if only a single message reaches this 100% threshold, we still consider it valid evidence for the emergence of temporal reference. The only scenario in which a message could reach 100% without carrying temporal meaning is a degenerate one — a dataset where the same object is repeated every time. Otherwise, consistent and exclusive use of a single message in repeated contexts indicates that the agents have developed a specific, maximally efficient marker for temporal reference; something highly desirable, as such a message is both highly compressible and interpretable.

Example. Consider an object sequence $[\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{y}, \boldsymbol{y}, \boldsymbol{y}, \boldsymbol{x}]$ with three possible corresponding message sequences:

(1) $[\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_2, \boldsymbol{m}_4, \boldsymbol{m}_4, \boldsymbol{m}_1]$
(2) $[\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_4, \boldsymbol{m}_4, \boldsymbol{m}_4, \boldsymbol{m}_1]$
(3) $[\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_2, \boldsymbol{m}_2, \boldsymbol{m}_2, \boldsymbol{m}_1]$

Assume horizon $n = 1$ (i.e., $\ominus^1$). There are two repetitions in the sequence of objects: the second and third $\boldsymbol{y}$ following the sequence of $[\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{y}]$.

In message sequence (1), $\boldsymbol{m}_4$ is used exactly for the second and third repetitions of $\boldsymbol{y}$ and never otherwise: $C_{\boldsymbol{m}_4}\ominus^1 = 2$, $\mathcal{T}_{\boldsymbol{m}_4} = 2$, and $M_{\ominus^1}(\boldsymbol{m}_4) = 100\%$.

In message sequence (2), $\boldsymbol{m}_4$ is also used for the initial (non-repeating) instance, giving $C_{\boldsymbol{m}_4}\ominus^1 = 2$, $\mathcal{T}_{\boldsymbol{m}_4} = 3$, and $M_{\ominus^1}(\boldsymbol{m}_4) = 66\%$.

In message sequence (3), $\boldsymbol{m}_2$ is used both for the initial and repeated instances of $\boldsymbol{y}$: $C_{\boldsymbol{m}_2}\ominus^1 = 2$, $\mathcal{T}_{\boldsymbol{m}_2} = 4$, and $M_{\ominus^1}(\boldsymbol{m}_2) = 50\%$.

This example further illustrates why only messages scoring 100% are considered to signify a temporal reference.

4.2 Correctness Metric

Correctness quantifies how often the receiver agent correctly identifies the target object upon receiving a given message. For a message $\boldsymbol{m}$ used N times, if the receiver correctly guesses the target in K of those cases, the correctness score is:

$$\text{Correctness}(\boldsymbol{m}) = \frac{K}{N} \times 100\% \tag{6}$$

This metric captures mutual intelligibility between sender and receiver: a high score indicates consistent, shared interpretation of the message.

4.3 Compositionality Metrics

The emergent languages are also analysed in terms of their compositionality scores, using the topographic similarity metric (Brighton and Kirby 2006), commonly employed in emergent communication. Topographic similarity measures the Spearman Rank correlation (Spearman 1904) between the distances of messages and objects in their respective spaces. For example, a message describing a "blue circle" should be closer to "blue triangle", than to "red square", given the language is compositional.

The languages are also evaluated using metrics which account for languages where the symbols themselves carry all the information: *posdis*, which use the positional information of individual characters, and *bosdis*, for permutation invariant languages, (Chaabouni, Kharitonov, Bouchacourt, et al. 2020). Posdis intuitively measures if, for example, the first symbol always refers to a property of the object, such as in the English phrases "blue circle" or "red square", versus "circle blue" or "square red". Bosdis measures whether a symbol carries all the information independent of the position of this symbol, such as in the English conjunctions example from Chaabouni, Kharitonov, Bouchacourt, et al. (2020), "dogs and cats" and "cats and dogs", where both constructions are valid and convey the same information.

5 Experiments

To study the impact of the architecture, as well as the external and internal pressures of the agents, we propose three hypotheses to guide our experiments:

Hypothesis 1 (H1) All agents can develop some form of temporal references, with agents that include the temporal prediction loss more likely to do so.

Hypothesis 2 (H2) Temporal references will increase the agent's performance in environments which include temporal relationships.

Hypothesis 3 (H3) No temporal references will be detected with the $M_{\ominus^n}$ metric in environments where there are no temporal relationships.

Hypothesis 1 is investigated by using multiple agent types and applying a temporal prediction loss. Hypothesis 2 is analysed by using two environments, TRG and TRG Hard, where temporal relationships are explicitly introduced. We then compare the performance of agents which develop temporal references, to those which do not to determine the impact of temporal references on the task accuracy. We include all evaluation environments,

including those where there are no target repetitions when evaluating the $M_{\ominus^n}$ to validate Hypothesis 3, which acts as a sanity check.

To test our hypotheses, the following architectures are trained and evaluated:

Base The same as the EGG (Kharitonov et al. 2019) agents, used as a baseline for comparisons (Section 3.1);
Temporal The base learner with the sequential LSTM *instead* of the regularly batched LSTM (Section 3.2); and
TemporalR The base learner with the regularly batched LSTM *and* the sequential LSTM (Section 3.3)

Each agent type is additionally trained with and without the temporal prediction loss. The agents that include the temporal prediction loss have an explicit reward to develop temporal understanding. There is no additional pressure to develop temporal references for agents that do not include the temporal prediction loss, except for the possibility of increased performance on the referential task. The aim of the additional loss is to investigate if it would aid in the temporal reference emergence, or if it improves agent performance. We show all the evaluated agent configurations in Section A.

All agent configurations were trained for the same number of epochs and on the same environments during each run. Agent pairs are trained in either the RG or TRG environment. Evaluation is performed after the training has finished in all environments Section 2.3. All target objects are uniformly sampled from the object space V. Each possible configuration was run ten times, with randomised seeds for both the agents and the datasets. We vary the target repetition chance, $p \in \{0.25, 0.5, 0.75\}$, in the training datasets to examine how repetition frequency affects the development of temporal references. Section B provides further details on all training hyperparameters.

5.1 Evaluation and Results

The metric value distribution of each network type is analysed, using the Kruskal-Wallis H-test (Kruskal and Wallis 1952), as scores are not guaranteed to be distributed normally. Conover-Iman (Conover and Iman 1979) post-hoc analysis is performed, with Holm-Bonferroni (Holm 1979) corrections applied, to verify which of the different network types differ significantly from each other. Using these methods, all results reported in the following sections have been verified to be statistically significant, with $p < 0.05$. The detailed results of the significance analyses are shown in the supplemental material.

Task Accuracy. All agents achieve high task accuracy, with all achieving over 95% in the Referential Games (RG) environment. Both Hard variants (*i.e.*, RG Hard and TRG Hard) present a challenge to the agents. All agents perform significantly worse in these two evaluation environments, achieving approximately 72% accuracy on average for the RG Hard environment, and 85% accuracy for the TRG Hard environment. We observe no increase in the task accuracy for the agents which develop temporal references, proving H2 incorrect. We provide a discussion of the possible reasons behind this in Section 6. The detailed accuracy distributions are provided in Section D.

Temporal Analyses. In line with H3, the values of $M_{\ominus^n}$ for the Never Same, RG, and RG Hard environments consistently register at 0%. These results reduce the likelihood of significant issues with the $M_{\ominus^n}$ metric, given that the probability of target repetition in these environments is near zero. Since the results remain constant and at 0% for these environments, they are omitted from the subsequent sections for brevity. Detailed results for these environments are available in both Sections D and E, and supplemental material.

Table 1 illustrates the $M_{\ominus^4}$ metric values, referring to an observation four messages in the past, of all agent types over the evaluation environments (*cf.*, Sections 4 and 5), where $M_{\ominus^4} \geq 0\%$.[3] Table 1 indicates that the temporally focused processing of the input data makes the agents predetermined to develop temporal references. Only the

[3]The value of 4 is chosen arbitrarily, to lie in the middle of the explored range of h. More detailed analysis from $h_v = 1$ to $h_v = 8$ is provided in the supplemental material

sequential LSTM (*Temporal* and *TemporalR*) agents, are capable of producing temporal references. Temporal references emerge in these networks, regardless of the training dataset. Even in a regular environment, without additional pressures, temporal references are advantageous. No messages in the *Base* architectures are used 100% of the time for $\ominus^4$, irrespective of the dataset they have been trained on. The temporal prediction loss is not enough, and the ability to process observations temporally is the key factor to the emergence of temporal references. These results partially confirm H1, indicating that all agents capable of explicitly processing temporal relationships develop temporal references and that no additional pressures are required. While we expected that the additional temporal prediction loss would improve the development of temporal references, this analysis indicates that it is not sufficient, nor necessary.

Additionally, messages that are used for $\ominus^4$ have a high chance of being correct, with most averaging above 90% correctness (Section 4.2). A detailed overview of the distribution of message correctness is provided in the supplemental material.

Table 1. Maximum value of the $M_{\ominus^4}$ metric for each network/loss/training environment combination. "AS" is Always Same, "RG" is the regular Reference Game environment, "TRG" is the Temporal Reference Game, and "TRG Hard" is TRG with the target differing in a single attribute with respect to the distractors.

Network	Loss	Training Env	AS	TRG	TRG Hard
Base	Reg	RG	60%	85%	85%
Base	Reg	TRG	60%	85%	85%
Base	Reg+T	RG	60%	85%	85%
Base	Reg+T	TRG	60%	85%	85%
Temporal	Reg	RG	**100%**	**100%**	**100%**
Temporal	Reg	TRG	**100%**	**100%**	**100%**
Temporal	Reg+T	RG	**100%**	**100%**	**100%**
Temporal	Reg+T	TRG	**100%**	**100%**	**100%**
TemporalR	Reg	RG	**100%**	**100%**	**100%**
TemporalR	Reg	TRG	**100%**	**100%**	**100%**
TemporalR	Reg+T	RG	**100%**	**100%**	**100%**
TemporalR	Reg+T	TRG	**100%**	**100%**	**100%**

Most messages are used only in the context of the current observations, with *Temporal* and *TemporalR* networks using a more specialised subset of messages to refer to the temporal relationships. Only *Temporal* and *TemporalR* variants develop messages that reach 100% on the $M_{\ominus^4}$ metric. The distribution also suggests that these messages could be a more efficient way of describing objects, as the number of temporal messages is relatively small. Since only a small number of messages are needed for the temporal references, they could be used more frequently. This message specialisation, combined with a linguistic parsimony pressure (Rita, Chaabouni, et al. 2020), could lead to a more efficient way of describing an object: sending the object properties requires more bandwidth than sending only the time step the object last appeared. A detailed breakdown of the messages used is provided in the supplemental material.

The percentage of networks that develop temporal messaging is shown in Table 2. The percentages shown are absolute values, calculated by taking the total number of runs and checking whether at least one message has reached $M_{\ominus^n} = 100\%$ for each run. That number of runs is divided by the total number of runs of the corresponding configuration to arrive at the quantities in Table 2.

The *Temporal* and *TemporalR* network variants reach over 96% of runs that have converged to a strategy which uses at least one message as the $\ominus^n$ operator. In contrast, the *Base* networks never achieve such a distinction. However, some runs have not converged to a temporal strategy in the case of the *TemporalR* network. These instances account for only 3% of the total number of runs, and the differences are not statistically significant from the *Temporal* network, showing that the emergence of temporal references is still very likely, if somewhat dependent on the network initialisation. These results indicate that the ability to build a temporally focused representation of the input data is the deciding factor in the emergence of temporal references.

Table 2. Percentage of networks that develop temporal messages.

Network Type	Loss Type	Percentage
Base	Regular loss	0%
Base	Regular + Temporal loss	0%
Temporal	Regular loss	100%
Temporal	Regular + Temporal loss	100%
TemporalR	Regular loss	98.66%
TemporalR	Regular + Temporal loss	97%

When the repetition chance p increases, the percentage of messages that are used for $\ominus^n$ increases for all agent variants. We observe no other notable differences (*e.g.*, differing temporal strategies or emergence rates) between the different values of p; therefore, this is the only difference reported. On average, *Base* networks demonstrate the same chance of using a message for $\ominus^n$ as the dataset repetition chance. This means that while the percentage increases, it is only due to the increase in the repetition chance. In contrast to the *Base* networks, for *Temporal* and *TemporalR* networks, the average percentage does reach 100%. This means that messages the agents designate for $\ominus^n$ are used more often than the repetition chance.

Language Analyses. All agents create compositional languages with varying degrees of structure, which shows that learning to use temporal references does not negatively impact compositionality. All agents reach values between 0.1 and 0.2 (the higher, the more compositional the language is) on the topographic similarity metric (Brighton and Kirby 2006; Rita, Tallec, et al. 2022), where a score of 0.4 has been considered high in previous research (Rita, Tallec, et al. 2022). We provide a visual representation of the topographic similarity scores in Section E. These results indicate that temporal references have no negative effect on the languages' compositionality, showing that their emergence does not necessitate a trade-off in the possible generalisation ability of the emergent language (Auersperger and Pecina 2022).

The differences between the *posdis* and *bosdis* distributions for each network type are not statistically signifiant. Therefore, the differences in the *posdis* and *bosdis* metrics across network types could be due to random fluctuations in the score distribution.

Analysing the development of temporal references, we observe that agents trained with a temporal module learn to use messages that refer consistently to prior object occurrences.

As an example, in one training run, using the *TemporalR* configuration, a specific message, $\boldsymbol{m_t} = [25, 6, 9, 3, 2]$, consistently emerged as a temporal reference, or a $\ominus^1$ operator. During evaluation in the Always Same environment, this message was used exclusively in cases where the target object had appeared in previous observations. Notably, it was used across twelve distinct objects, but only after their initial appearance.

For each of these twelve objects, which were repeated ten times each during evaluation, this message was used in nine out of ten occurrences. A different message was used only when the object appeared for the first

time. For instance, when the object [4, 2, 3, 6, 5, 8, 8, 4] first appeared, the agents produced a different message, [25, 6, 17, 9, 9]. On subsequent occurrences, however, the temporal message $\boldsymbol{m_t}$ was used instead.

This consistent pattern demonstrates that the agents learn to generalise temporal references developed during training. Importantly, the same temporal message learned in the TRG environment was successfully transferred and reused in the Always Same environment, despite the object sets being different. This suggests that the agents do not rely on object identity alone but indeed use temporal references to generalise their behaviour across tasks.

6 Discussion

The results in Section 5.1 indicate that no explicit pressures are required for temporal messages to emerge, unlike increasing linguistic parsimony where additional losses are needed (Kalinowska et al. 2022; Rita, Chaabouni, et al. 2020). We show that the incentives are already present in datasets that are *not* altered to increase the number of repetitions occurring. Temporal references therefore emerge naturally, as long as the agents are able to build a temporal understanding of the data, such as with the sequential LSTM in the *Temporal* and *TemporalR* agents. This allows temporal references to emerge in any communication settings if a suitable architecture is used. This could provide greater bandwidth efficiency by allowing agents to use shorter messages for events that happen often, when combined with other linguistic parsimony approaches (Chaabouni, Kharitonov, Dupoux, et al. 2019; Rita, Chaabouni, et al. 2020).

The emergence of temporal references only through architectural changes could also point towards additional insights in terms of modelling human language evolution using emergent communication (Galke et al. 2022). These architectural approaches to the emergence of temporal references could be viewed as analogous to sequential learning in natural language (Christiansen and Kirby 2003), as we learn to encode and represent elements in temporal sequences.

By focusing only on the three main factors for temporal reference emergence — architectural capacity, dataset repetition, and a temporal loss — we can confidently attribute emergence to the sequential understanding built by the temporal module. Future work can build on this foundation by exploring more complex features and aspects of temporal references in emergent communication. Such cases could include how often the agents use temporal references, and what combination of factors influence the agent's choices to use temporal references or to instead describe the object directly.

6.1 Accuracy

As expected, both the RG Hard and the TRG Hard environments posed a challenge, presenting a significant accuracy drop. In the case of the *Temporal*, *TemporalR* agents, we hypothesised the emergence of temporal references to provide an advantage, increasing the agents' accuracy (Section 5). While there indeed is a small increase in accuracy in the TRG Hard environment, it is not attributable to temporal references (*cf.*, Section D). This is because we observe the same increase for the *Base* agents, which do not develop temporal references. We conclude that the increase is due to increased object repetition, making the task slightly easier.

A possible cause for the lack of accuracy increase for agents which develop temporal references might be the perceptual similarities between the highly similar distractor objects. This makes the task of discerning the difference between these objects becoming too difficult for the receiver. Alternatively, if the receiver struggled to correctly identify an object the first time it has observed it, the additional information that temporal references would offer would not be enough. The receiver would not know what the correct choice was for the previous timesteps.

Additionally, networks that have been trained with the temporal loss *and* on the temporally focused dataset, perform slightly worse, by about 1%. The reason for the accuracy drop could lie in too much pressure on the temporal aspects of the dataset. Because of the additional loss, agents can increase their rewards by only focusing

on creating temporal messages, without learning a general communication protocol. This then leads to overfitting the training dataset, where they can rely on both their mostly temporal language and their memory of the object sequences, instead of communicating about the object attributes. Consequently, we observe a decline in performance on the evaluation dataset.

6.2 Compositionality

We hypothesise the effect of the temporal loss on the topographic similarity scores (we do not discuss the posdis and bosdis scores, as there are no statistically significant differences) is similar to that of the task accuracy. Since the agents focus on the temporal aspects of the task and dataset more, they develop less general languages, leading to lower compositionality scores. We do not believe the low compositionality scores are related to the simplicity of the dataset, being composed of integer vectors, since other works in similar settings achieve higher topographic similarity scores (Chaabouni, Kharitonov, Bouchacourt, et al. 2020; Rita, Tallec, et al. 2022).

7 Limitations

All reported compositionality scores could be negatively affected by the presence of temporal references. Temporal messages can be compositional, but they would not refer to a specific object, and so topographic similarity would not be able to identify them correctly. Similarly, since posdis and bosdis also rely on the mappings between the messages and the dataset, instead of the temporal relationships captured by the temporal references, they could also be inaccurately lowered by their presence. Since temporal references, even if they were compositional, do not map directly to object properties in the dataset, they would be counted as non-compositional messages, therefore lowering the values of the evaluated metrics. This could be the reason for the lower values observed in our experiments when compared to previous research (Rita, Tallec, et al. 2022).

8 Conclusion

Emergent communication has been studied extensively, considering many aspects of emergent languages, such as efficiency (Chaabouni, Kharitonov, Dupoux, et al. 2019; Rita, Chaabouni, et al. 2020), compositionality (Auersperger and Pecina 2022), generalisation (Chaabouni, Kharitonov, Bouchacourt, et al. 2020) and population dynamics (Chaabouni, Strub, et al. 2022; Michel et al. 2022; Rita, Strub, et al. 2022). Yet, there has been no evidence of temporal relationships emerging. Discussing past observations is vital to communication, saving bandwidth by avoiding repeating information and allowing for easier experience sharing in multiagent communication.

This paper provides a first exploration of such emergent languages, including addressing the fundamental questions of when they could develop and what is required for their emergence. To achieve this, we confine our investigation to three main factors, enabling a targeted analysis of temporal reference emergence without potential confounding variables from more complex setups. We present a set of environments that are designed to facilitate investigation into how agents might create such references. We use the conventional agent architecture for emergent communication (Kharitonov et al. 2019) as a baseline and explore both temporal loss and alternative architectures that may endow agents with the ability to learn temporal relationships. We show that architectural change is necessary for temporal references to emerge, and demonstrate that temporal prediction loss is neither sufficient for their emergence, nor does it improve the emergent language. Our findings highlight the importance of analysing architectural designs in the study of multiagent communication, providing valuable insights for future research using environments where temporal relationships are important or into how other linguistic aspects might emerge.

Acknowledgments

This work was supported by the UK Research and Innovation Centre for Doctoral Training in Machine Intelligence for Nano-electronic Devices and Systems [EP/S024298/1]. Adam Sobey was supported by The Alan Turing Institute under the Lloyd's Register Foundation grant ATI/100004. The authors would like to thank Lloyd's Register Foundation for their support, and acknowledge the use of the IRIDIS High-Performance Computing Facility, and associated support services at the University of Southampton, in the completion of this work.

References

M. Auersperger and P. Pecina. 2022. "Defending Compositionality in Emergent Languages." In: *Proceedings of the 2022 Conference of the NAACL: Human Language Technologies: Student Research Workshop*, 285–291.

L. Biewald. 2020. *Experiment Tracking with Weights and Biases*. en-US. (2020).

B. Boldt and D. R. Mortensen. 2024. "A Review of the Applications of Deep Learning-Based Emergent Communication." *Transactions on Machine Learning Research*.

N. Brandizzi. 2023. "Toward More Human-Like AI Communication: A Review of Emergent Communication Research." *IEEE Access*, 11, 142317–142340.

N. Brandizzi, D. Grossi, and L. Iocchi. 2021. "RLupus: Cooperation through emergent communication in The Werewolf social deduction game." *Intelligenza Artificiale*, 15, 2, 55–70.

H. Brighton and S. Kirby. 2006. "Understanding Linguistic Evolution by Visualizing the Emergence of Topographic Mappings." *Artificial Life*, 12, 2, 229–242.

R. Chaabouni, E. Kharitonov, D. Bouchacourt, E. Dupoux, and M. Baroni. 2020. "Compositionality and Generalization In Emergent Languages." In: *Proceedings of ACL*, 4427–4442.

R. Chaabouni, E. Kharitonov, E. Dupoux, and M. Baroni. 2019. "Anti-efficient encoding in emergent communication." In: *Proceedings of NeurIPS*.

R. Chaabouni, F. Strub, et al.. 2022. "Emergent Communication at Scale." In: *Proceedings of ICLR*.

P. D. Chocron and M. Schorlemmer. 2020. "Vocabulary Alignment in Openly Specified Interactions." en. *Journal of Artificial Intelligence Research*, 68, 69–107. doi:10.1613/jair.1.11497.

M. H. Christiansen and S. Kirby. 2003. "Language evolution: consensus and controversies." en. *Trends in Cognitive Sciences*, 7, 7, 300–307.

W. Conover and R. Iman. 1979. *On multiple-comparisons procedures*. Tech. rep. Los Alamos National Lab.

W. Falcon and The PyTorch Lightning Team. 2019. *PyTorch Lightning*. (2019).

L. Galke, Y. Ram, and L. Raviv. 2022. "Emergent Communication for Understanding Human Language Evolution: What's Missing?" en. In: *Emergent Communication Workshop at ICLR 2022*.

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

S. Holm. 1979. "A Simple Sequentially Rejective Multiple Test Procedure." *Scandinavian Journal of Statistics*, 6, 2, 65–70.

D. A. Huffman. 1952. "A Method for the Construction of Minimum-Redundancy Codes." *Proceedings of the IRE*, 40, 9, 1098–1101.

E. Jang, S. Gu, and B. Poole. 2017. "Categorical Reparameterization with Gumbel-Softmax." In: *Proceedings of ICLR*.

A. Kalinowska, E. Davoodi, F. Strub, K. Mathewson, T. Murphey, and P. Pilarski. 2022. "Situated Communication: A Solution to Over-communication between Artificial Agents." en. In: *Emergent Communication Workshop at ICLR 2022*.

Y. Kang, T. Wang, and G. de Melo. 2020. "Incorporating Pragmatic Reasoning Communication into Emergent Language." In: *Proceedings of NeurIPS*.

E. Kharitonov, R. Chaabouni, D. Bouchacourt, and M. Baroni. 2019. "EGG: a toolkit for research on Emergence of lanGuage in Games." In: *Proceedings of EMNLP*.

D. P. Kingma and J. Ba. 2015. "Adam: A Method for Stochastic Optimization." In: *Proceedings of ICLR*.

D. E. Knuth. 1985. "Dynamic huffman coding." *Journal of Algorithms*, 6, 2, 163–180.

K. Kopparapu et al.. 2022. "Hidden Agenda: a Social Deduction Game with Diverse Learned Equilibria." *ArXiv preprint*.

W. H. Kruskal and W. A. Wallis. 1952. "Use of Ranks in One-Criterion Variance Analysis." *Journal of the American Statistical Association*, 47, 260, 583–621.

A. Lazaridou and M. Baroni. 2020. "Emergent Multi-Agent Communication in the Deep Learning Era." *ArXiv preprint*, abs/2006.02419.

A. Lazaridou, K. M. Hermann, K. Tuyls, and S. Clark. 2018. "Emergence of Linguistic Communication from Referential Games with Symbolic and Pixel Input." In: *Proceedings of ICLR*.

A. Lazaridou, A. Peysakhovich, and M. Baroni. 2017. "Multi-Agent Cooperation and the Emergence of (Natural) Language." In: *Proceedings of ICLR*.

D. K. Lewis. 1969. *Convention: A Philosophical Study*. Cambridge, MA, USA: Wiley-Blackwell.

O. Lichtenstein, A. Pnueli, and L. Zuck. 1985. "The glory of the past." In: *Workshop on Logic of Programs*. Springer, 196–218.

O. Lipinski, A. Sobey, F. Cerutti, and T. J. Norman. 2022. "Emergent Password Signalling in the Game of Werewolf." en. In: *Emergent Communication Workshop at ICLR 2022*.
O. Maler, D. Nickovic, and A. Pnueli. 2008. "Checking Temporal Properties of Discrete, Timed and Continuous Behaviors." en. In: *Pillars of Computer Science: Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday*. Lecture Notes in Computer Science. Springer, 475–505.
P. Michel, M. Rita, K. W. Mathewson, O. Tieleman, and A. Lazaridou. 2022. "Revisiting Populations in multi-agent Communication." en. In.
A. Pnueli. 1977. "The temporal logic of programs." In: *18th Annual Symposium on Foundations of Computer Science*, 46–57.
M. Rita, R. Chaabouni, and E. Dupoux. 2020. ""LazImpa": Lazy and Impatient neural agents learn to communicate efficiently." In: *Proceedings of the CoNLL*.
M. Rita, P. Michel, R. Chaabouni, O. Pietquin, E. Dupoux, and F. Strub. 2024. *Language Evolution with Deep Learning*. arXiv:2403.11958 [cs]. (2024). doi:10.48550/arXiv.2403.11958.
M. Rita, F. Strub, J. Grill, O. Pietquin, and E. Dupoux. 2022. "On the role of population heterogeneity in emergent communication." In: *Proceedings of ICLR*.
M. Rita, C. Tallec, P. Michel, J.-B. Grill, O. Pietquin, E. Dupoux, and F. Strub. 2022. "Emergent Communication: Generalization and Overfitting in Lewis Games." en. In: *Proceedings of NeurIPS*.
I. Rochlin and D. Sarne. 2015. "Constraining Information Sharing to Improve Cooperative Information Gathering." en. *Journal of Artificial Intelligence Research*, 54, 437–469. doi:10.1613/jair.4613.
C. Spearman. 1904. "The Proof and Measurement of Association between Two Things." *The American Journal of Psychology*, 15, 1, 72–101.
R. Ueda, T. Ishii, K. Washio, and Y. Miyao. 2022. "Categorial Grammar Induction as a Compositionality Measure for Emergent Languages in Signaling Games." en. In: *Emergent Communication Workshop at ICLR 2022*.
R. Vieira, A. F. Moreira, M. Wooldridge, and R. H. Bordini. 2007. "On the Formal Semantics of Speech-Act Based Communication in an Agent-Oriented Programming Language." en. *Journal of Artificial Intelligence Research*, 29, 221–267. doi:10.1613/jair.2221.
J. S. Vitter. 1987. "Design and analysis of dynamic Huffman codes." *Journal of the ACM*, 34, 4, 825–845.
J. Wang, M. Guo, and Z. Li. Sept. 2023. *Multi-agent Coordination Under Temporal Logic Tasks and Team-Wise Intermittent Communication*. arXiv:2308.14042 [cs]. (Sept. 2023). doi:10.48550/arXiv.2308.14042.
G. K. Zipf. 1949. *Human behavior and the principle of least effort*.

A Experimental Combinations

To complement the main experiment description, we include Table 3, which summarises the different training configurations, evaluated environments, and relevant result sections.

Table 3. Summary of Experimental Combinations. Each architecture is trained with and without the temporal prediction loss, across all values of repetition chance $p \in \{0.25, 0.5, 0.75\}$, in both the Referential Games (RG) and Temporal Referential Games (TRG) training environments. Evaluations are conducted in all six environments. All configurations are run with 10 random seeds. Relevant results are discussed in the sections listed.

Architecture	Training Env	Temporal Loss	Repetition Chance p	Eval Envs	Results
Base	RG / TRG	True / False	0.25 / 0.5 / 0.75	All 6	Section 5.1
Temporal	RG / TRG	True / False	0.25 / 0.5 / 0.75	All 6	Sections 5.1 and 6
TemporalR	RG / TRG	True / False	0.25 / 0.5 / 0.75	All 6	Sections 5.1 and 6

B Training Details

Our agents were trained using PyTorch Lightning (Falcon and The PyTorch Lightning Team 2019) using the Adam optimizer (Kingma and Ba 2015), with experiment tracking done via Weights & Biases (Biewald 2020). We provide our grid search parameters per network and per training environment in Table 4. We ran a manual grid search over these parameters for each network and training dataset combination, where the networks were *Base*, *Temporal*, *TemporalR* and the training datasets were Referential Games or Temporal Referential Games. Each trained network was then evaluated on the six available environments: Always Same, Never Same, Referential

Games, Temporal Referential Games, Hard Referential Games, and Hard Temporal Referential Games. Running the grid search for one iteration, with the value of repetition chance fixed, took approximately 28 hours, using the compute resources in Table 5.

Table 4. Grid Search Parameters

Parameter	Value
Epochs	[600]
Optimizer	Adam
Learning Rate α	0.001
Number of Objects in Dataset	[20 000]
Number of Distractors	[10]
Number of Attributes N_{att}	[8]
Number of Values N_{val}	[8]
Temporal Prediction Loss Present	[True, False]
Length Penalty	[0]
Maximum Message Length L	[5]
Vocabulary Size N_{vocab}	[26]
Repetition Chance (p)	[0.25, 0.5, 0.75]
Previous Horizon h	[8]
Sender Embedding Size	[128]
Sender Meaning LSTM Hidden Size	[128]
Sender Temporal LSTM Hidden Size	[128]
Sender Message LSTM Hidden Size	[128]
Receiver LSTM+Linear Hidden Size	[128]
Gumbel-Softmax Temperature	[1.0]

Table 5. Compute Resources

Resource	Quantity
CPU Cores (Intel(R) Xeon(R) Silver 4216 × 2)	20
GPUs (NVIDIA Quadro RTX8000)	1
Wall Time	28hrs

C Datasets Details

In Figure 4, we analyse our datasets, using the parameters as specified in Section B, for the number of repetitions that occur. When the temporal dataset repetition chance is set to 50%, the datasets, predictably, oscillate around 50% of repeating targets. Generating the targets randomly yields a miniscule fraction of repetitions of less than 1%, as we can see in Figure 4, for the Classic and Hard referential games.

C.1 Test Environments

Both Always Same and Never Same environments act as sanity checks for our results.

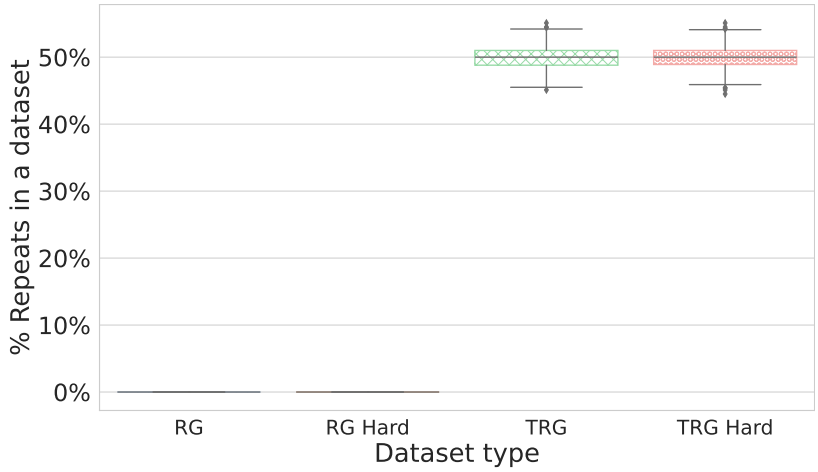


Fig. 4. Number of target repetitions per dataset. Regular referential games datasets very rarely encounter target repetitions. This data is an average over 1000 seeds per environment.

We provide example inputs and outputs for both environments in Table 6 and Table 7. We use single-attribute objects and messages for clarity.

For the Always Same environment, in the case of the agent using temporal references, we may also see other messages instead of the message $\boldsymbol{m}_4$, as we have observed that there are more than one message used as previously. We always expect to see at most 90% of usage as previously for this environment, unless the agents learn temporal referencing strategies, when we would expect the usage to reach 100%.

For the Never Same environment, we expect to see no temporal references being identified. Any identification of temporal references in the Never Same environment would indicate an issue with our metric.

Table 6. Example Inputs and Outputs for Always Same.

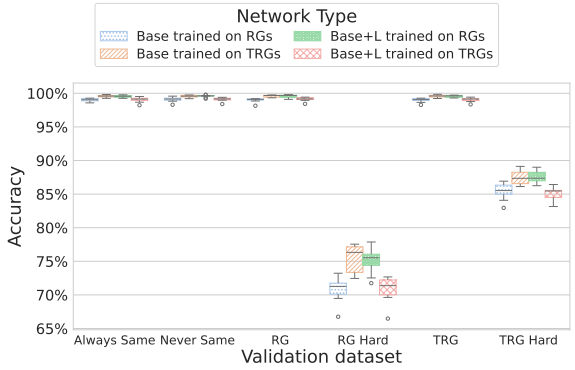
Environment	Always Same
Input	$[\boldsymbol{x}, \boldsymbol{x}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{y}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{z}, \boldsymbol{z}]$
Temporal Referencing	$[\boldsymbol{m}_1, \boldsymbol{m}_4, \boldsymbol{m}_4, \boldsymbol{m}_2, \boldsymbol{m}_4, \boldsymbol{m}_4, \boldsymbol{m}_3, \boldsymbol{m}_4, \boldsymbol{m}_4]$
No Temporal Referencing	$[\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_4, \boldsymbol{m}_5, \boldsymbol{m}_6, \boldsymbol{m}_7, \boldsymbol{m}_8]$

Table 7. Example Inputs and Outputs for Never Same.

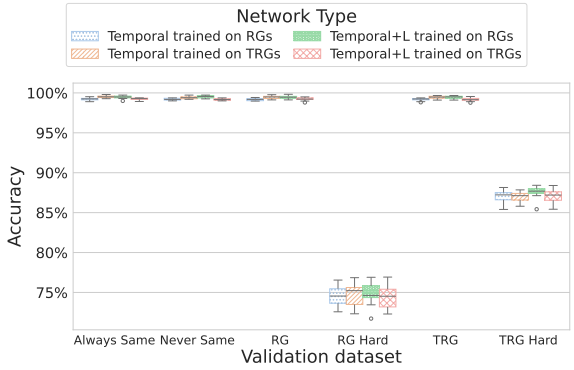
Environment	Never Same
Input	$[\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}]$
Temporal Referencing	$[\boldsymbol{m}_1, \boldsymbol{m}_1, \boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_2, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_3, \boldsymbol{m}_3]$
No Temporal Referencing	$[\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_4, \boldsymbol{m}_5, \boldsymbol{m}_6, \boldsymbol{m}_7, \boldsymbol{m}_8]$

D Accuracy Distributions

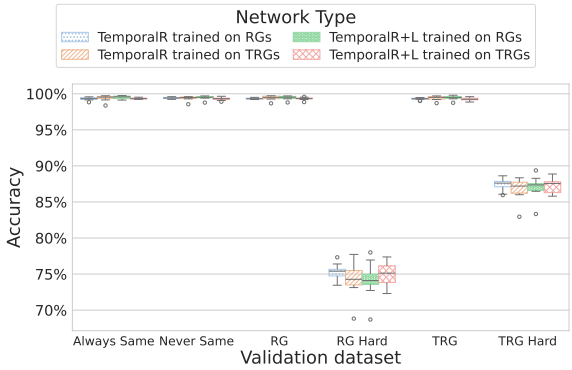
The accuracy distributions for all agent types across all evaluation environments are shown in Figures 5a to 5c. All agents converge to very similar levels of accuracy.



(a) The evaluation accuracy for the *Base* agents across all environments.



(b) The evaluation accuracy for the *Temporal* agents across all environments.

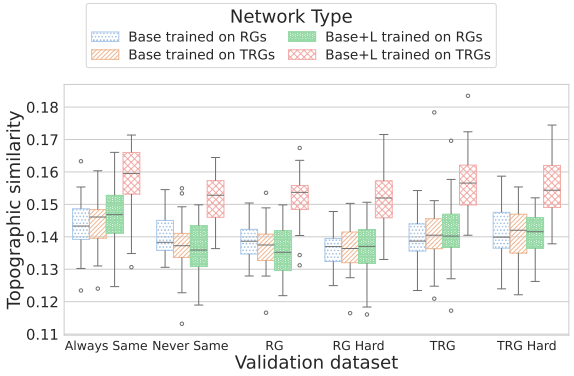


(c) The evaluation accuracy for the *TemporalR* agents across all environments.

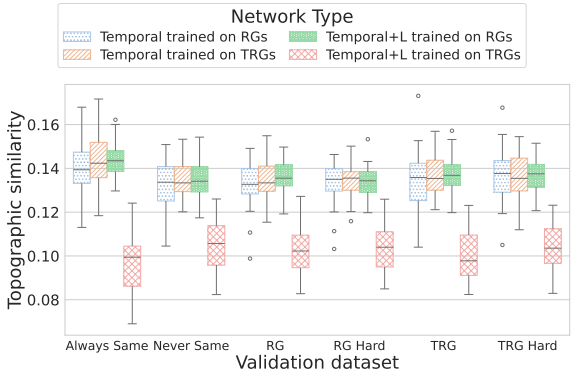
Fig. 5. Accuracies for each network variant on all evaluation environments.

E Topographic Similarity Distributions

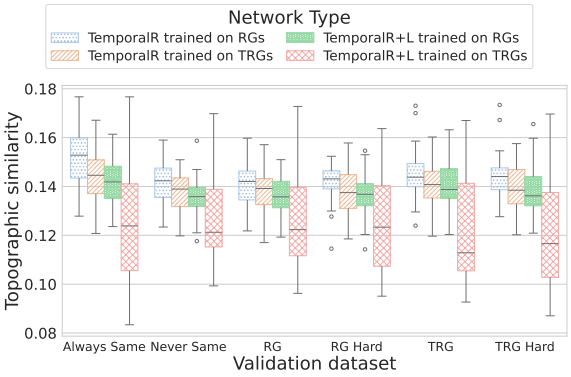
The topographic similarity distributions for all agent types across all evaluation environments are shown in Figures 6a to 6c. All agents converge to very similar values of topographic similarity.



(a) The topographic similarity scores for the *Base* agents across all environments.



(b) The topographic similarity scores for the *Temporal* agents across all environments.



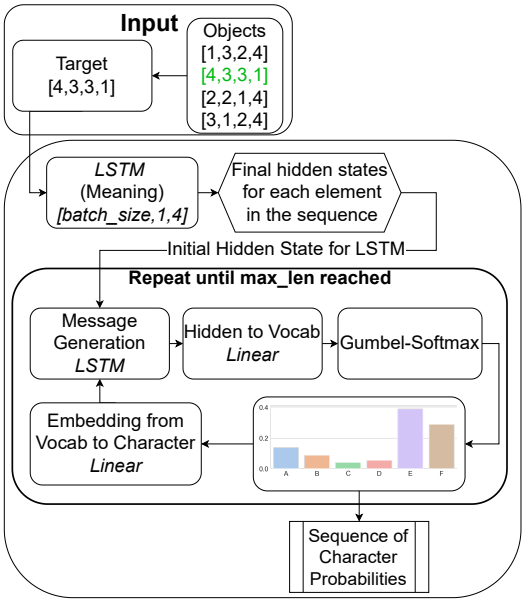
(c) The topographic similarity scores for the *TemporalR* agents across all environments.

Fig. 6. Topographic similarity scores for each network variant on all evaluation environments.

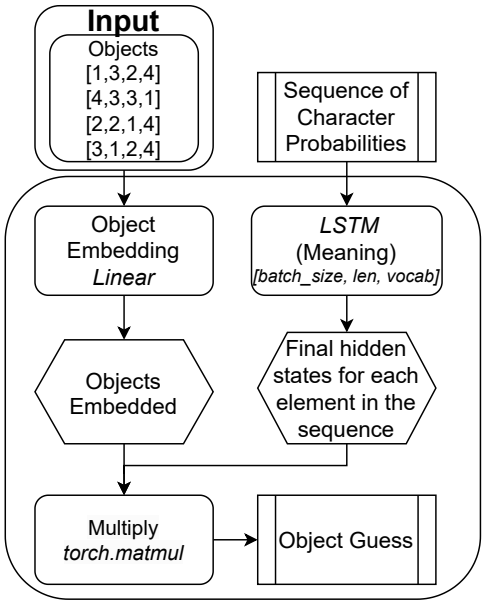
F Architectural Details

In Figure 7 and Figure 8 we present the detailed overview of the *Base* and *TemporalR* agents respectively. Compared to the *Temporal* agent, the *Base* agent lacks the sequential LSTM, while the *TemporalR* agent has the regularly batched LSTM, in addition to the sequential LSTM.

In Figure 9, we present an overview of our whole experimental setup. We can see the sender and receiver architectures, together with their inputs, as described in Section 3. We also show our loss calculations.

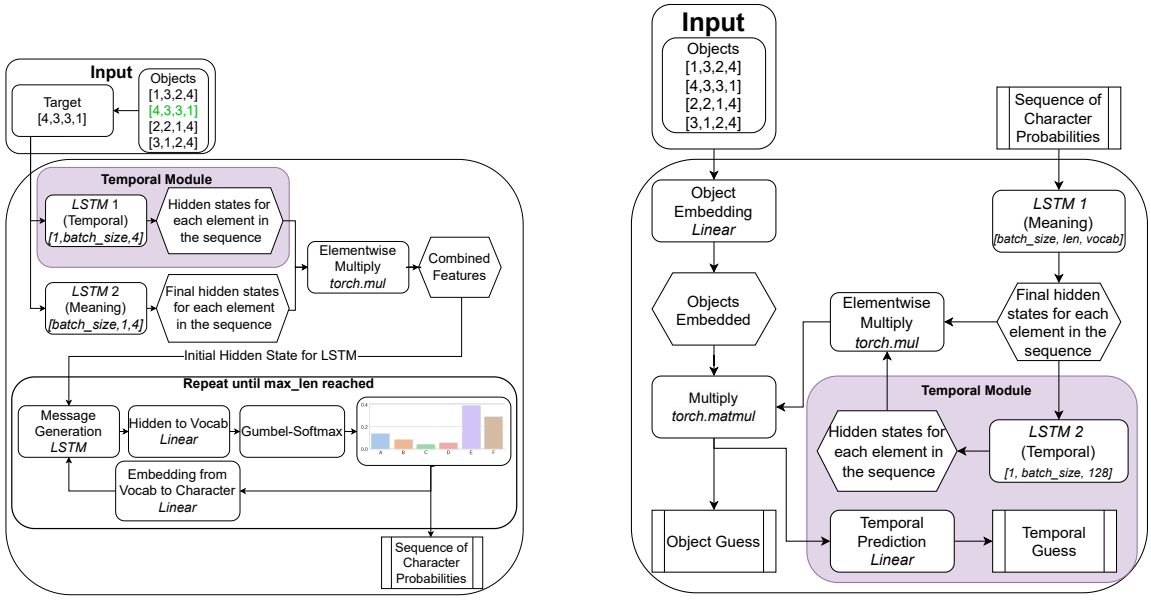


(a) The *Base* LSTM sender architecture.



(b) The *Base* LSTM receiver architecture.

Fig. 7. The *Base* LSTM sender and receiver architectures.



(a) The *TemporalR* LSTM sender architecture.

(b) The *TemporalR* LSTM receiver architecture.

Fig. 8. The *TemporalR* LSTM sender and receiver architectures, with the temporal modules highlighted in purple.

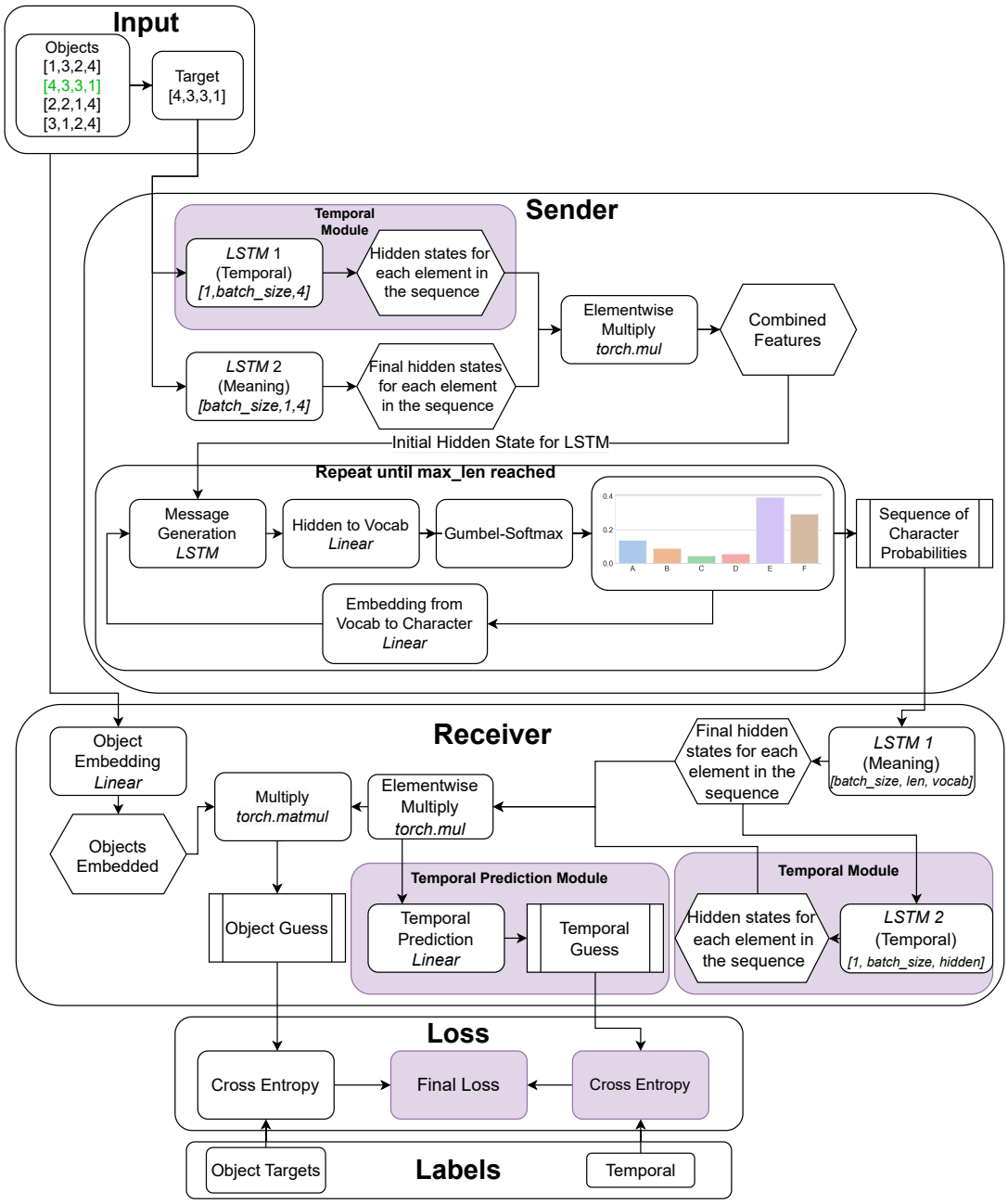


Fig. 9. Full overview of our Temporal Referential Games setup. Together with the sender and receiver we have described in Section 3, we also include the working of the loss.

Received 09 July 2025; accepted 19 March 2026