UNIVERSITY OF SOUTHAMPTON

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Temporal Dynamics in Emergent Communication

by

Olaf Lipinski

BSc

ORCiD: 0000-0002-2023-7617

A thesis for the degree of
Doctor of Philosophy

February 2025

University of Southampton

Abstract

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Doctor of Philosophy

Temporal Dynamics in Emergent Communication

by Olaf Lipinski

Emergent communication is an approach designed to enhance the communicative capabilities of agents in multiagent systems. Unlike traditional multiagent communication fields, emergent communication allows agents to learn both the structure and content of their communication protocol with minimal constraints on character sets or vocabularies. This flexibility enables the development of more efficient, adaptable, and environment-specific languages compared to hand-crafted protocols. In this thesis, we identify a significant gap in the existing literature on emergent communication, specifically the lack of exploration of temporal dynamics of emergent languages. To address this, we investigate three dimensions of temporality in emergent communication. First, we examine the influence of communication duration on agent behaviour in a social deduction game called Werewolf, by allowing agents to communicate for varying lengths of time. Our findings reveal that agents develop unexpected strategies and that our modifications enhance their ability to converge on a common language. Second, we study the emergence of temporal references, or words indicating relative positions in time. We introduce a novel environment where agents must communicate about temporal relationships within a dataset. The results demonstrate that agents can learn to reference different time steps to solve the environment successfully. Third, we explore how agents communicate about local spatio-temporal relationships within a single observation. The results not only show that the agents learn to communicate about such relationships, but also that this language can be human interpretable. The contributions presented in this thesis pave the way for more efficient and adaptable protocols in multiagent communicative settings.

Contents

List of Figures **ix**

List of Tables **xi**

Declaration of Authorship **xiii**

Acknowledgements **xv**

Abbreviations **xvii**

1 Introduction **1**
1.1 Emergent Communication 1
1.2 Time in Emergent Communication 3
1.3 Research Aims 4
1.4 Contributions and Novelty 5
1.5 Thesis Outline 6

2 Literature Review **7**
2.1 Linguistics Background for Emergent Communication 7
2.2 Language Emergence with Deep Learning 9
2.3 Emergent Language Properties 9
2.3.1 Compositionality 9
2.3.2 Language Efficiency 10
2.3.3 Generalisation 11
2.3.4 Understudied Properties 12
2.4 Quantifying the Properties of Emergent Languages 13
2.4.1 Measuring Compositionality 13
2.4.2 Analysing Semantics 14
2.5 Emergent Communication Environments 15
2.5.1 Referential Games 15
2.5.2 Werewolf 16
2.6 Agent Architectures 18
2.7 Agent Optimisation 19
2.8 Conclusions 20

3 Interaction Time in Dialogue **23**
3.1 Emergent Communication in Werewolf 23
3.2 Werewolf Environment 24

3.3 Architecture . . . 26
3.4 Interaction Time Experiments . . . 27
3.4.1 Hypotheses . . . 27
3.4.2 Convergence Speed . . . 27
3.4.3 Win Rate . . . 27
3.4.4 Comparison to the Original Environment . . . 29
3.4.5 Language Analysis . . . 30
3.5 Discussion . . . 31
3.5.1 Strategy Analysis . . . 32
3.5.2 Convergence Speed . . . 32
3.5.3 Win Rate . . . 33
3.5.4 Failure Modes . . . 33
3.6 Limitations . . . 34
3.7 Conclusions . . . 35

4 Temporal References **37**
4.1 Temporal References in Emergent Communication . . . 37
4.2 Temporal Referential Games . . . 39
4.2.1 Definitions . . . 39
4.2.2 Temporal Logic . . . 40
4.2.3 Temporal Referential Games . . . 41
4.3 Agent Architectures . . . 43
4.3.1 *Base* Agent . . . 44
4.3.2 *Temporal* Agent . . . 44
4.3.3 *TemporalR* Agent . . . 46
4.4 Measuring Temporality and Compositionality . . . 47
4.4.1 Temporality Metric . . . 47
4.4.2 Compositionality Metrics . . . 49
4.5 Temporal Referencing Experiments . . . 49
4.5.1 Hypotheses . . . 49
4.5.2 Agent Training . . . 50
4.5.3 Significance Analysis . . . 51
4.5.4 Task Accuracy . . . 51
4.5.5 Temporality Sanity Check . . . 51
4.5.6 Temporality Analysis . . . 51
4.5.7 Compositionality Analysis . . . 54
4.5.8 Generalisation Analysis . . . 54
4.6 Discussion . . . 55
4.6.1 Accuracy . . . 55
4.6.2 Compositionality . . . 56
4.7 Limitations . . . 56
4.8 Conclusion . . . 56

5 Spatio-temporal References **59**
5.1 Spatio-temporal Referencing in Emergent Communication . . . 59
5.2 Spatio-temporal Referential Game . . . 60
5.2.1 Referential Game Environment . . . 60

5.2.2 Spatio-temporal Reference Formalisation 62
5.3 Agent Architecture 64
5.4 Message Interpretability and Analysis using NPMI 65
5.5 Spatio-temporal Referencing Experiments 69
5.5.1 Emergence of non-compositional spatio-temporal references . . . 70
5.5.2 Emergence of compositional spatio-temporal references 70
5.5.3 Generalisation 71
5.5.4 Evaluating interpretation validity and accuracy 72
5.6 Discussion 74
5.7 Limitations 75
5.8 Conclusion 75

6 Discussion **77**
6.1 Compositionality 77
6.2 Efficiency 78
6.3 Scalability 79
6.4 Temporality 80
6.5 Broader Impact 81

7 Conclusion **83**

Appendix A Werewolf **85**
Appendix A.1 Training Details 85
Appendix A.2 Statistical Significance Analysis 86

Appendix B Temporal Referential Games **89**
Appendix B.1 Training Details 89
Appendix B.2 Datasets Details 91
Appendix B.2.1 Test Environments 92
Appendix B.3 Accuracy Distributions 93
Appendix B.4 Topographic Similarity Distributions 94

Appendix C Temporal Progression Games **95**
Appendix C.1 Training Details 95
Appendix C.2 Dataset Details 95
Appendix C.3 NPMI Algorithm Descriptions 96

Glossary **103**

References **105**

List of Figures

2.1 An illustration of the referential game environment. 16
2.2 Visual representation of the flow of the game of Werewolf. 17
2.3 The base architecture of EGG agents (Kharitonov et al., 2019). 18

3.1 Werewolf agent architecture. 26
3.2 Impact of the number of rounds on the convergence speed. 28
3.3 Impact of the voting plurality threshold on the convergence speed. . . . 28
3.4 Impact of the voting plurality threshold on the win rate. 29
3.5 Impact of the number of communication rounds on the win rate. 29
3.6 Most used unique message **and** its top ten distance-one adjacents versus the villager win percentage. 31
3.7 Most used unique message versus the villager win percentage. 31
3.8 Impact of the number of communication rounds on the number of training episodes. 34

4.1 Attribute-Value Object Representation 40
4.2 Structure of the referential game and temporal referential game. 42
4.3 The *Base* GRU sender and receiver architectures. 44
4.4 The *Temporal* GRU sender and receiver architectures, with the temporal modules highlighted in purple. 45
4.5 Examples of regular and temporal batching strategies. 46
4.6 The *TemporalR* GRU sender and receiver architectures, with the temporal modules highlighted in purple. 47

5.1 Spatio-temporal referencing example. 63
5.2 The sender and receiver architectures. Adapted from (Lipinski et al., 2023). 65
5.3 Examples of the different types of message compositionality that are possible to identify using the PMI algorithms. 67

Appendix A.1 Spearman correlation strength and its significance. 87

Appendix B.1 Number of target repetitions per dataset. 91
Appendix B.2 Accuracies for each network variant on all evaluation environments. 93
Appendix B.3 Topographic similarity scores for each network variant on all evaluation environments. 94

List of Tables

3.1 Results for both our and the original Brandizzi et al. (2021) environment. 30

4.1 Maximum value of the $M_{\ominus^4}$ metric for each network/loss/training environment combination. 52
4.2 Percentage of networks that develop temporal messages. 53

5.1 Average emergence and vocabulary coverage of all message types. . . . 71
5.2 Evaluation accuracy differences for shorter sequence lengths compared to the training sequence lengths. 72
5.3 Accuracy improvements using the NPMI-based dictionary. 73
5.4 Example dictionary of the agents' messages and their meanings 75

Appendix A.1 Training and Grid Search Parameters 85
Appendix A.2 Compute Resources 86
Appendix A.3 Linear regression analysis. 86

Appendix B.1 Training and Grid Search Parameters 90
Appendix B.2 Compute Resources 90
Appendix B.3 Example Inputs and Outputs for Always Same. 92
Appendix B.4 Example Inputs and Outputs for Never Same. 92

Appendix C.1 Compute resources 95
Appendix C.2 Training and Grid Search Parameters 96
Appendix C.3 PMI Grid Search Parameters 96

Declaration of Authorship

I declare that this thesis and the work presented in it is my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Parts of this work have been published as: Lipinski et al. (2022); Lipinski (2023); Lipinski et al. (2023, 2024)

Signed:..... Date:.....

Acknowledgements

First, I would like to thank my supervisors, Tim Norman, Adam Sobey and Federico Cerutti. Their guidance and support have been invaluable and have made me the researcher I am today. The countless drafts that I have sent, and the many comments, culminated in an incredible improvement in how I can understand and communicate complex topics more clearly. I am truly grateful for their time, patience, and trust, especially when the deadlines always seemed just around the corner.

I would also like to thank all my friends at the MINDS CDT — for the pub escapes from deadlines, the deep discussions over drinks, and most importantly, for the true friendships formed that, I hope, will last a lifetime.

I would also like to thank my family and friends in Poland. Their emotional support has been invaluable, from reading my drafts despite being outside their field, to making each return visit feel like I had never left.

Last but definitely not least, I would like to thank my partner, Elinor, whose unwavering support and patience have been a blessing throughout my PhD. Her willingness to listen to my research talks, despite working in a completely different field, and her encouragement, during both victories and challenges, have made this achievement possible.

It is through all your support that this work was made possible.

Abbreviations

AI Artificial Intelligence.

DRL Deep Reinforcement Learning.

EC Emergent Communication.

HAS Harris' Articulation Scheme.

IoT Internet of Things.

LLM Large Language Model.

LLMs Large Language Models.

LTL Linear Temporal Logic.

ML Machine Learning.

NN Neural Networks.

NPMI Normalised Pointwise Mutual Information.

PLTL Past Linear Temporal Logic.

RG Referential Game.

RGs Referential Games.

RL Reinforcement Learning.

RLAIF Reinforcement Learning from AI Feedback.

RLHF Reinforcement Learning from Human Feedback.

TPG Temporal Progression Game.

TPGs Temporal Progression Games.

TRG Temporal Referential Game.

TRGs Temporal Referential Games.

VAE Variational Autoencoder.

VAEs Variational Autoencoders.

Chapter 1

Introduction

Machine Learning (ML), and more specifically, Neural Networks (NN) and Deep Reinforcement Learning (DRL) have made strides of progress since the increased interest in the techniques from 2012 (Maslej et al., 2024). They have since been used in a variety of domains, from transcribing ancient texts (Assael et al., 2022) or control of tokamak plasmas for improvements in fusion energy (Degrave et al., 2022) to optimising the algorithms that are crucial components of our operating systems (Mankowitz et al., 2023).

A subfield of ML recently experiencing unprecedented leaps in both neural network abilities and funding, with over $25.2 billion invested in 2022 (Maslej et al., 2024), is research into Large Language Models (LLMs). While mostly known for the GPT chatbot models (Brown et al., 2020; Radford et al., 2018; OpenAI et al., 2023), LLMs are also used to study aspects of human cognition and linguistic ability (e.g., Cheng et al., 2024; Yin et al., 2024; Kuribayashi et al., 2024). While the study of LLMs bases the insights on languages **taught** to the agents, our work also lies in a field at the intersection of ML and linguistics that takes the opposite approach: Emergent Communication.

1.1 Emergent Communication

Emergent Communication (EC) is a subfield of multiagent communication research, where instead of specifying the protocol, or language, that agents would use to communicate, they are allowed to develop it from scratch. The agents control both the structure and content of this protocol, developing a language through repeated interaction in given tasks. In common with other subfields of Machine Learning, Emergent Communication is experiencing a surge in interest in different emergent language properties, such as the impact of different pressures and biases on the emergent language (e.g., Tucker et al., 2022), the effect of agent and population structure

and homogeneity (e.g., Rita et al., 2022a), or the evolution of grammars and natural language-like structures (e.g., Ueda et al., 2023).

The goal of such experiments is two-fold. Firstly, the field of EC aims to improve the communication efficiency and interpretability of multiagent systems (Boldt and Mortensen, 2024b; Rita et al., 2024). The emergent protocol is naturally optimised through the way it is developed. As the language emerges in a specific environment, it is naturally tailored to its observation space. Given that the agents can only perceive their task-specific environment, they do not need to learn words that would refer to irrelevant objects. Instead, the protocol can be more task specific, while still allowing for generalisation to unseen observations within the task context (Rita et al., 2022b; Auersperger and Pecina, 2022; Xu et al., 2022).

With advances in EC, we could expect improvements in terms of ad-hoc connectivity in multiagent systems (Cope and McBurney, 2022), given that the language developed by the agents is easily transferable (Li and Bowling, 2019). Ad-hoc connectivity would then directly translate to many applications, such as communication in ad-hoc teams of agents for autonomous vehicles or IoT (Cope and McBurney, 2022; Abudu and Markham, 2020). Agents that can more easily teach others their language, will also benefit from more effective communication themselves, as the language would be more general and structured (Cogswell et al., 2019). A generalisable protocol will enable them to adapt to new observations and environments (Silver et al., 2021), allowing the agents to interact autonomously more easily. As more agents are equipped with the ability to adapt and learn new emergent languages, the quality and efficiency of these interactions would also improve (Baroni, 2020a).

With the appropriate pressures and reward structures, the observations that agents refer to most will also have the shortest utterances (Rita et al., 2020), increasing their bandwidth efficiency. Both of these properties mean that there are no redundant phrases in the language, and that the agents can use their vocabulary as efficiently as possible. Therefore, there is less need for a human to hand-code any optimised protocols, also removing the need to account for all possible observations and their corresponding descriptions for a given task (Silver et al., 2021).

Through analysing the emergence of different properties of language among agents, we can attempt to understand the evolution and origins of human languages (Warstadt and Bowman, 2022; Boldt and Mortensen, 2024b; Rita et al., 2024), learning what influences may have shaped their structure, thanks to the relative ease of analysing and observing the behaviour of neural networks as compared to human brains (Boldt and Mortensen, 2024b).

However, the applicability of insights from the field of EC is not limited to multiagent communication or linguistics. Another aspect that Emergent Communication could potentially enable is easier acquisition of human language, for either human-agent

interaction (Baroni, 2020a), or for natural language processing (Yao et al., 2022; Steinert-Threlkeld et al., 2022). EC has been shown to offer improvements in the latter (Steinert-Threlkeld et al., 2022; Yao et al., 2022), where pre-training a natural language model on an emergent communication dataset improves its performance on NLP tasks. Emergent Communication has been shown to perform better than other systems in terms of their linguistic ability (Baroni, 2020a), better learning of compositional representations (Xu et al., 2022) and even better generalisation in recognising objects in images (Feng et al., 2023).

EC has been applied to various domains, varying from Augmented Reality (Chen and Guo, 2023), controlling flying base stations (Naoumi et al., 2023), task offloading in IoT (Mostafa et al., 2024b), intent profiling (Mostafa et al., 2024a), and has even been applied to help interpret fMRI data (Latheef et al., 2024). These advances demonstrate that EC has a large potential impact on fields not related to multiagent communication, and could improve other aspects of ML (Boldt and Mortensen, 2024b).

1.2 Time in Emergent Communication

A key feature of human communication is the ability to correctly reference the temporal sequence of events. We use words that can refer to the past or future, specify when certain actions have occurred, and take turns to achieve our goals. This ability fundamentally shapes how we communicate and interact with others.

For artificial agents, the capacity to discuss temporal relationships is equally crucial. However, in the current emergent communication literature, the temporal aspect of language has been ignored. There are no examples of agents developing or understanding temporal references or using temporal aspects of their environment to their advantage.

Agents deployed in autonomous vehicles or monitoring financial transactions would benefit from being able to refer to temporal relationships, as they could inform other agents about their experiences and observations. In the case of autonomous vehicles, agents could describe the conditions on different parts of the roads that the cars have driven through, aiding traffic control and enhancing safety. In financial monitoring, agents could relay information about past suspicious transactions and when they occurred, helping to prevent fraud by recognizing patterns over time. In smart warehouses, robots and sensors could optimize inventory management by discussing the arrival and departure times of goods. In smart homes, devices could synchronize actions based on user routines and historical data. In manufacturing, machines could adjust operations based on the timing of production stages.

As environmental complexity is being scaled in emergent communication research (Chaabouni et al., 2022; Rita et al., 2024), temporal references will also benefit agents in settings where temporal relationships are embedded. One example is social deduction games, where referencing past events is key to winning strategies. These games, which often involve multiple players attempting to uncover hidden roles or information, rely heavily on the players' abilities to recall and reference past events accurately. The strategic use of temporal references, such as indicating when a particular event occurred or referencing a sequence of past actions, can be crucial for devising winning strategies. Temporal referencing will also allow agents to develop more efficient methods of communication by assigning shorter messages using temporal references. For instance, instead of elaborating on a series of past events in detail, an agent might simply reference a specific time point or a known sequence of actions, thus conveying the necessary information succinctly.

1.3 Research Aims

This thesis aims to bring the advantages of using temporal dynamics in natural language to the field of Emergent Communication. To do so, we address three key research questions.

How does the amount of interaction time impact the emergent language? We examine the relationship between communication duration and the emergent language in a social deduction game of Werewolf, where teams of agents need to collaborate to eliminate the opposing team, through agreeing on a common voting strategy. By varying the time agents have to communicate, we analyse how the length of interaction time affects both the agents' strategies and the properties of the emergent language. The agents must develop effective communication protocols to establish voting majorities and coordinate their strategies, making this an ideal environment to study how different interaction times influence the emergent language. We explore this in detail in Chapter 3.

What is needed for agents to communicate about temporal relationships across their dataset? We focus on the emergence of temporal references in agent communication, or the ability to refer to past events. We investigate the minimal conditions and architectural requirements that enable agents to develop temporal references in their language. Additionally, we analyse the compositional properties of the emergent languages, comparing those with and without temporal references to understand how the development of temporal references affects language compositionality. We present this work in Chapter 4.

How do agents communicate about spatio-temporal relationships within their observations? We study how agents can communicate spatio-temporal relationships between parts of their observations. We explore whether complex environments are truly required for such language properties to emerge (Rita et al., 2024), or are simple referential games (Section 2.5) enough. We analyse the resulting language using normalised pointwise mutual information, a linguistic collocation measure, to investigate its structure and interpretability. We provide the details of this study in Chapter 5.

1.4 Contributions and Novelty

Our work analysing the impact of the amount of time to communicate in the game of Werewolf (Lipinski et al., 2022) shows that, even for simple strategies, increasing the number of turns the agents can take to communicate improves their convergence efficiency and their game performance. We also show that the agents develop a highly efficient and successful strategy of using passwords to communicate their identities.

To analyse the emergence of languages capable of expressing temporal relationships between different observations (Lipinski et al., 2023), we develop a novel environment, called Temporal Referential Games (TRGs). This variant of the commonly used referential game environment allows us to repeat targets that the agents observe, allowing them to exploit the temporal relationships. By using a simple environment to study the emergence of temporal references, we explore what building blocks are required for them to emerge. We develop a novel metric to analyse the emergence of such references. We show that additional losses are unnecessary, and that the key to the emergence of such languages are simple changes to the agent architecture, paving the way for more ubiquitous use of temporal references in other EC settings.

To analyse the development of languages capable of discussing spatio-temporal relationships (Lipinski et al., 2024) within data points we develop a new environment, based on the referential games, called Temporal Progression Games (TPGs). Using this simple environment, we show that the usual agent architecture requires only a small modification to be able to understand and communicate about such relationships. We also provide an analysis of the emergent language, showing how the agents compose their messages, by using a novel NPMI measure. We also show that the resulting language is interpretable using this measure.

1.5 Thesis Outline

This thesis first analyses the available literature from both the field of EC, and related linguistics concepts in Chapter 2, providing the theoretical foundation for temporal dynamics in artificial languages. Chapters 3 to 5 are our contribution chapters. Chapter 3 investigates the relationship between the length of communication time and emergent language in the social deduction game of Werewolf. Chapter 4 explores the conditions required for temporal references to emerge in agent communication. Chapter 5 examines the emergence of spatio-temporal communication in a referential game environment. In Chapter 6, we present a critical analysis of the work discussed in the previous chapters, together with future directions that this line of research could take. Chapter 7 concludes the work presented in this thesis.

Chapter 2

Literature Review

Since the field has begun, the evolution of language has been of interest to linguists, with many theories and investigations into the origin of human language (e.g. Hockett, 1960; Pinker and Bloom, 1990; Mesoudi et al., 2011). With the increase in computational power, this question started to be studied through simulation (Steels and Kaplan, 1999; Kirby, 1999). This has led to research in Emergent Communication (EC), and its application to multiagent communication.

2.1 Linguistics Background for Emergent Communication

From a linguistic perspective, the current work in EC pertains to the pragmatics and semantics of emergent language. **Pragmatics**, as defined by Korta and Perry (2020), concerns the context around *words* rather than the properties of them. Analysing the pragmatics of an emergent language would be studying the context and its effect on meaning behind the phrases used by agents. Pragmatics in the context of EC are much simpler than that considered by linguistics, as the languages studied in EC are analogically simpler. For an overview of pragmatics in linguistics, we refer to Kempson (2003) and Korta and Perry (2020).

One pragmatic property of human languages relevant to this work is the existence of deixis (Lyons, 1977; Stapleton, 2017). Deixis has been described as a way of pointing through language, with words such as “here” or “now”. These two words are examples of *spatial* and *temporal* deixis, respectively. Deixis can also be categorized into other types such as *personal*, *discourse*, and *social* deixis (Stapleton, 2017). Personal deixis refers to the use of pronouns like “I”, “you”, and “we”, which indicate the participants in a conversation. Discourse deixis involves expressions like “this” or “that” when referring to parts of the discourse itself, aiding in the navigation of conversation. Social deixis pertains to words that encode social information about the relationships between

speakers, such as honorifics or titles. The effective use of deixis requires a shared context and mutual understanding between interlocutors, making it a fundamental aspect of communication that bridges the gap between language and the physical, temporal, and social world.

Another pragmatic aspect of human language is the existence of anaphora. Anaphora enhances textual cohesion by referring back to something previously mentioned in discourse. Anaphoric references can take several forms, including *pronominal*, *nominal*, and *adverbial* anaphora (Halliday and Hasan, 1976). Pronominal anaphora uses pronouns like "he," "she," or "it" to refer to earlier, or future, entities. Nominal anaphora employs nouns or noun phrases, such as "the teacher" or "the book," while adverbial anaphora uses adverbs or adverbial phrases like "there" or "then." Anaphora is crucial for managing information flow and maintaining cohesion in text, making it vital for tasks in natural language processing and computational linguistics, such as machine translation and text summarization.

In contrast to pragmatics, the study of **semantics** focuses on the analysis of meaning (Lappin, 2003). In linguistics, semantics refers to the meaning behind words, sentences, or longer form utterances. The two main subfields of semantics are that of *compositional semantics* and *lexical semantics*. The former studies the structure of smaller parts of speech, and how new meanings can be created by combining these components. The latter focuses on the meaning of the smaller parts itself, how they acquire their meaning, and how new meanings of words can be created.

From within the field of semantics, comes the *principle of compositionality* (Szabó, 2020), a term often used in emergent communication. **Compositionality**, as defined by Szabó (2020), states that meaningful expressions in natural language, can be built using smaller forms of other meaningful expressions. For example, combining the words “green” and “banana” to create the phrase “green banana” conveys a different meaning than the two words individually. More formally, as stated by Szabó (2020), “For every complex expression e in L, the meaning of e in L is determined by the structure of e in L and the meanings of the constituents of e in L", where L is the language under consideration. This more formal definition is known as the aforementioned *principle of compositionality* in linguistics.

Compositionality has been argued as a way to increase natural language productivity (Szabó, 2020), the ability to describe a virtually limitless number of things. Given the knowledge of just a few words, such as “black, white, gray, cat, stripes” the English language makes it possible to describe tens of combinations of a cat’s fur, using its compositional qualities. Once a natural language is learned, the understanding of it is also productive and allows for processing and producing of almost any conveyable sentence in that language (Szabó, 2020). These properties made compositionality a

desideratum in emergent communication (Brighton, 2002; Smith et al., 2003; Vogt, 2005), as it has the potential to increase the efficiency and generality of emergent languages.

Another linguistics analysis that has been recently used in EC is text **segmentation**. As a linguistic term, segmentation refers to dividing sentences into their core components. One such technique is Harris' Articulation Scheme (HAS), which can segment a sentence into its constituent words by using the changes in the probability of appearance of consequent phonemes (Harris, 1955).

2.2 Language Emergence with Deep Learning

We break down the progress in five specific areas of Emergent Communication research: the relevant properties of the emergent language, the ways used to measure such properties, the environments used to analyse the emergence of such properties, the agent architectures and the optimisation techniques used to train EC agents.

2.3 Emergent Language Properties

The main properties that have been investigated in EC are compositionality (e.g. Lazaridou et al., 2018; Chaabouni et al., 2020; Auersperger and Pecina, 2022), transmission efficiency and linguistic parsimony (e.g. Chaabouni et al., 2019; Rita et al., 2020), generalisation to novel input and different modalities (e.g. Harding Graesser et al., 2019; Chaabouni et al., 2020), ease of learning and transmission (e.g. Li and Bowling, 2019; Ueda et al., 2023), influence of competition (e.g. Noukhovitch et al., 2021; Liang et al., 2020), population dynamics (e.g. Rita et al., 2022b; Mahaut et al., 2023) and their effect on the language, and interpretability of the emergent protocol (e.g. Andreas et al., 2017; Mihai and Hare, 2021; Cope and McBurney, 2024).

In this thesis we focus our review on the properties of compositionality, transmission efficiency and generalisation as the most relevant to our work. We also analyse some of the understudied properties of emergent languages.

2.3.1 Compositionality

Compositionality has been the most researched property in EC. The main aspects that have been analysed are the way to incentivise compositionality (e.g. Kirby, 1999; Vogt, 2005; Chaabouni et al., 2020), generalisation of compositional languages (e.g. Harding Graesser et al., 2019; Baroni, 2020b; Chaabouni et al., 2020), and ways to measure compositionality (e.g. Lowe et al., 2019; Korbak et al., 2020; Perkins, 2021).

Some research has also been dedicated to the non-trivial aspects of compositionality, which are much harder to measure using conventional and well-established methods (Bogin et al., 2018; Perkins, 2021; Bosc, 2022). Compositional languages are a desideratum for EC research, as they may enable better generalisation and facilitate better human understanding (Michel et al., 2022). We discuss the ways to measure compositionality, as well as other properties of emergent languages, in Section 2.4

Compositionality, while at first may seem a trivial aspect, is not a given in emergent protocols. Numerous results indicate that compositionality does not emerge naturally, and instead agents develop *degenerate* or *holistic* languages (Kottur et al., 2017; Chaabouni et al., 2019; Lipinski et al., 2022). A degenerate language indicates that all meaning is associated with a single, ambiguous signal, while holistic languages do not contain any structure, and just map each distinct observation to a separate signal (Kirby et al., 2015). The main ways for compositionality to emerge that have been identified so far are information bottlenecks (Kharitonov et al., 2020), linguistic parsimony or Zipf's law pressure (Zipf, 1949; Chaabouni et al., 2019; Rita et al., 2020), the ease of transmission pressure (Li and Bowling, 2019; Ren et al., 2020), and noisy channels (Kucinski et al., 2021). However, little research has been done into the compound environmental pressures that may incentivise compositionality (Kucinski et al., 2021; Chaabouni et al., 2022). These could include the ability to refer to complex observations, where compositionality, in our view, would be a requirement for successful and efficient communication. For example, referring to a blue object seen two time steps ago is easier if the meanings of "two", "time step" and "blue" can be composed, instead of creating a unique word for the combination of these three properties.

Whether compositionality is a positive aspect of the emergent languages, and should be a goal when training agents, is debated. Some results point to compositionality being not necessary for generalisation (Chaabouni et al., 2020; Andreas, 2019; Kharitonov and Baroni, 2020), some that it is negatively correlated (Nikolaus, 2023), and some that it is positively correlated (Ohmer et al., 2022a). It may also be that due to linguistic variation in the languages, ones that are classified as non-compositional could nevertheless be compositional (Conklin and Smith, 2022). Together with the questions posed around the validity of the metrics used to measure compositionality (Section 2.4), this adds to the growing number of questions about the impact and importance of compositionality in emergent languages.

2.3.2 Language Efficiency

Language efficiency has also been of interest in emergent communication (Chaabouni et al., 2021). Chaabouni et al. (2019) have shown that without an advantageous pressure to do otherwise, agents will develop "anti-efficient" protocols. "Anti-efficiency" refers to the words that the agents assign to objects, usually using the maximum length

available. Unless a penalty is applied to reduce the utterance length, the agents will use all the available message length to convey their messages. This phenomenon differs notably from human languages, where articulatory effort often shapes language evolution, with many human languages showing a tendency toward efficiency through linguistic parsimony (Chaabouni et al., 2019). This pattern, however, is not constant across cultures and languages, with some maintaining longer forms in formal contexts while allowing significant contractions in casual speech, while others show different patterns of such behaviour altogether. In contrast, in artificial agents, the only pressure that exists on the networks is the perceptual pressure, where longer messages are easier to distinguish as they can be more distinct from each other (Lazaridou and Baroni, 2020). Rita et al. (2020) and Rodríguez Luna et al. (2020) explore the possible ways of applying this and other pressures to force the agents to create more succinct languages. Rita et al. (2020) suggest that both the speaker *and* the listener need to have a pressure towards efficiency. Once the pressure on both types of agents is in place, the emergent language follows the Zipf's Law [1] (Zipf, 1949), where the most used utterances are the shortest. However, the work of Rodríguez Luna et al. (2020) shows that transmission efficiency may be accomplishable through a different pressure, where the speaker is incentivised to end its messages as soon as possible.

2.3.3 Generalisation

Generalisation requirements for compositional, or structured, languages have also been disputed (Kharitonov and Baroni, 2020). It has been argued that compositional languages are not needed for generality because the pressures that exist in an environment do not necessarily incentivise a compositional language. Instead, they may incentivise only a general language, which may be holistic or degenerate, depending on the agent's tasks (Kharitonov and Baroni, 2020). Kharitonov and Baroni (2020) even argue that compositionality may not be a good target if we want to focus only on the generality of the protocol because it may be too human-centric.

This view has been challenged in recent works, with compositional languages leading to better generalisation abilities (Auersperger and Pecina, 2022). Auersperger and Pecina (2022) show that generalisation is only successful if the emergent languages are compositional. Similarly, Ohmer et al. (2022a) show that agents develop good generalisation through using a compositional hierarchical language. This has again been called into question by Nikolaus (2023), where they provide evidence that compositionality and generalisation could be negatively correlated.

Some degree of generality has already been shown in the early work of (Havrylov and Titov, 2017), where the agent's language exhibited some categorisation (Baroni, 2020c).

[1] Zipf's Law can also be viewed as an ordering according to the Kolmogorov complexity for each phrase (Manin, 2014)

Generalisation of emergent languages has also been shown through multiple tests of zero-shot evaluations (e.g., Choi et al., 2018; Tucker et al., 2021). It was observed that agents with a structured language, developed through environmental pressures, can afterwards perform better in other types of games and environments (Mu and Goodman, 2021).

In contrast to the works we have described so far, others have found that agents do not develop a sufficiently general language, but rather an *idiolect* (Steels and Kaplan, 1999). An idiolect is, in the case of EC, when the emergent language is specific to the speaking agent, and so the agents have to each learn each other's language instead of a general one, as was also noted by Bouchacourt and Baroni (2019).

2.3.4 Understudied Properties

Another aspect of interest is that of the influence of time on the emergent language. Research has been done on turn taking (Taillandier et al., 2023) and multistep interactions (Kalinowska et al., 2022) in emergent communication. However, in the current literature, there are few investigations concerning other aspects of the influence of time, such as how the amount of time to communicate affects the emergent language and the agent strategies. Similarly, while spatial and temporal deixis have been studied extensively in linguistics (Stapleton, 2017), they have been an underexplored topic in EC. We consider that these aspects will be significantly important in the future, as the environment and task complexity increases, requiring the use of more varied language properties.

Close to the investigation of deixis is the research into anaphoric structure in emergent communication (Edwards et al., 2023). Edwards et al. (2023) show that neural agents can easily acquire languages with anaphoric structure, and that such structures may be able to naturally emerge. The authors argue that, given emergent languages use unique n-grams that refer to redundancy and increase ambiguity (especially under the pressure of brevity), anaphoric structures emerge naturally.

The ability to refer to numerical concepts has also only recently been shown in EC by Zhou et al. (2024). Zhou et al. (2024) present a task where agents have to learn to refer to the number of objects present in their observations. These numerical concepts must then be used with a prescribed mathematical operator, such as addition or subtraction, to arrive at the correct answer to the task. While the authors show that the agents can learn to count, the agents do not learn about the mathematical operators, as these are prescribed. In essence, the sender agent learns to count, and the receiver agent learns to perform addition and subtraction given two numbers.

2.4 Quantifying the Properties of Emergent Languages

To measure all the properties that an emergent language could have, several metrics have been proposed. Most metrics concentrate on measuring compositionality (e.g., Andreas, 2019; Korbak et al., 2020; Bogin et al., 2018; Brighton and Kirby, 2006; Chaabouni et al., 2020), the influence of communication on the behaviour of other agents (e.g., Lowe et al., 2019; Eccles et al., 2019), or how to measure communication in competitive environments (e.g., Noukhovitch et al., 2021). In this thesis, we focus on the ways to measure compositionality, and the semantics of emergent languages, as the most relevant to this work.

2.4.1 Measuring Compositionality

Measuring compositionality is a non-trivial task. Most metrics developed so far suffer from the inability to measure non-trivial compositionality (Perkins, 2021; Korbak et al., 2020; Xu et al., 2022; Carmeli et al., 2024). This is because these metrics fail to account for the complex nature of compositional structures that differ from traditional human language patterns. Most use simple heuristics to define and find compositional languages. Metrics such as topographic similarity (Brighton and Kirby, 2006) would, for example, qualify the English language as non-compositional, as individual letters hold no meaning (Bosc and Vincent, 2022).

Topographic similarity can be viewed as a correlation between pairwise input distances and corresponding representation, or message, distances (Chaabouni et al., 2020). In recent work, topographic similarity has been shown to have a high correlation with generalisability of the emergent language (Rita et al., 2022b). However, the use of this metric is limited, as it would not be able to measure non-trivial compositionality (Perkins, 2021; Korbak et al., 2020) or when the input-representation pairs are unavailable (Ossenkopf et al., 2022).

As shown by Perkins (2021), intelligent agents can use unusual methods to create compositional structures. These would lead to negative results on most compositionality metrics, even if the emergent language is trivially compositional. That is why Perkins (2021) proposes a benchmark for future metrics to check for more complex types of compositionality. They create transformations that can be applied to a language to create compositional systems, which current metrics cannot measure. These can then be used to test any new proposed metrics to verify if they could still perform well against such languages.

Based on these shortcomings of analysing compositionality, Chaabouni et al. (2022) suggest that these attempts should be abandoned in favour of measuring more quantitative outcomes of language performance, such as task success. Rita et al. (2024)

suggest instead that measures and metrics should focus on using linguistic insights to be able to measure emergent languages. By using metrics developed for human language, or at least inspired by such techniques, emergent languages can be more closely compared to natural language.

2.4.2 Analysing Semantics

Analysing the semantics of the emergent language is challenging. By the nature of emergent communication settings, the language developed by the agents is adapted and optimised for their task, making it harder to understand how agents compose their messages (Perkins, 2021), and the meanings behind them. While most work in emergent communication focuses on discrete token-based languages, research has shown that agents can learn to communicate through visual channels using drawings (Mihai and Hare, 2021), similar to how visual communication preceded written language in human history. However, regardless of the communication modality, the fundamental challenge of semantic interpretation persists.

Attempts to interpret emergent messages often rely on the presence of dataset labels. One such method is using normalised mutual information (nMI) to express the closeness of dataset labels, or meanings, to the emergent language (Dessì et al., 2021). An extension to nMI has been proposed in the same work called WNsim which further measures the similarity based on the shortest path between two categories using WordNet (Miller, 1992), thus creating less penalty for using the same utterance for similar categories (Dessì et al., 2021). However, these methods require extensive dataset labelling.

A first step to interpreting an emergent language is to be able to segment messages into the atomic parts which carry different meanings. A new metric to quantify compositionality has also been recently proposed by Bosc (2022), called concatenability. This metric measures whether two atomic symbols or expressions can be concatenated together, and still retain their separate meanings. For example, if we assign the number *1* to the meaning of *red* and the number *3* to the meaning of *car*, then this metric would check if the symbol *13* meant *red car*, or if it had another meaning. This technique could be a first step towards interpreting each symbol of the emergent language, however Bosc (2022) still rely on dataset labels.

Another way of segmenting emergent language messages is Harris' Articulation Scheme (HAS) (Harris, 1955). Initially devised to segment natural language sentences based on phoneme probabilities, HAS has been adapted to employ entropy measures after each character to segment the words in the emergent language (Ueda et al., 2023). This method detects a separate word by the entropy falling after one character, while increasing on the next character. For example, in the English language, if we write

"lang", then it would be quite likely that the next letter is "u", and so the entropy would be lower. However, after a full word "language" the entropy is high, as we have many other characters we could use to start a new word. While the emergent languages have been shown to be segmentable, no meaningful segmentation has been found so far (Ueda et al., 2023).

An approach to both segment and create a meaningful mapping of the concepts from emergent messages to natural language has been developed by Carmeli et al. (2024). Carmeli et al. (2024) use adjusted mutual information to create a bipartite graph from words present in the messages to concepts present in the dataset. They show that agents can create interpretable mappings, and that these can be represented by a graph. However, the authors treat each character in a message as a separate word, meaning that any words which are longer than a single character would be ignored by this proposed method.

2.5 Emergent Communication Environments

An important factor in the evolution of the emergent protocol is the environment. The properties of a language depend on the domain it needs to cover, which makes the design of the environment an important aspect in determining how the agents create their languages. The environment design is essential to allow for better generalisation (Hill et al., 2020). Even how the environment is perceived can affect how and what the agents learn, with the perceptual biases passed onto the language (Ohmer et al., 2022b). The design of the agents' perception can even significantly affect their performance in a given task, with pretrained vision networks performing worse than vision networks trained in a given environment (Ohmer et al., 2022b). In this section, we will focus on two environments: Werewolf and Referential Games.

2.5.1 Referential Games

Referential games, also known as discrimination games, are a popular choice of environment for testing various aspects of emergent communication. First introduced by Lewis (1969), and adapted for the specific purpose of emergent communication by Lazaridou et al. (2017), the referential games are a relatively simple setting. There are two agents, a sender and a receiver. The sender observes a vector and transmits its compressed representation through a discrete channel to the receiver. The receiver observes a set of vectors together with the sender's message. One of these vectors is the same as the one the sender has observed. The receiver's goal is to correctly identify the vector the sender has described, among other vectors referred to as distractors. The

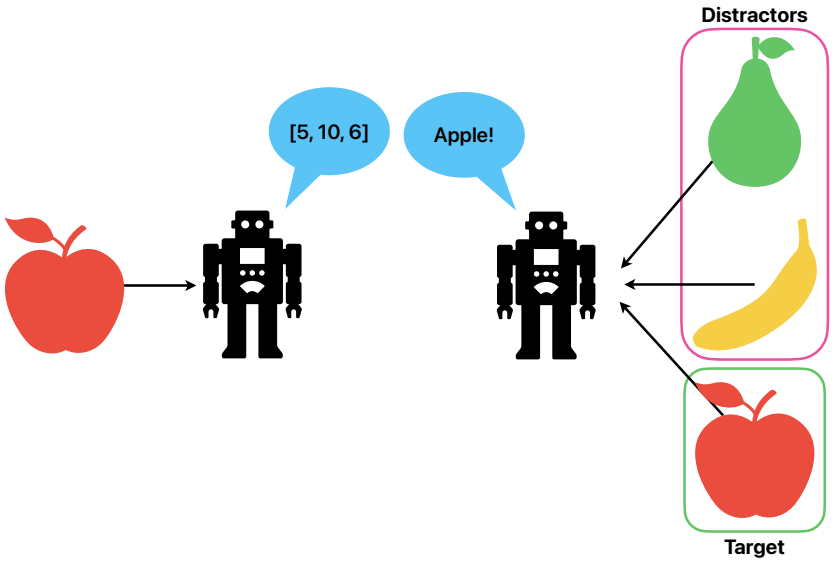


FIGURE 2.1: An illustration of the referential game environment.

simplicity of the referential games enables the reduction of extraneous factors which could impact the emergence of different language properties.

For illustration purposes, a single round of the referential games is provided in Figure 2.1. The sender observes a red apple, and transmits the vector [5,10,6] to the receiver. This vector could be composed of parts representing both red and apple, or it could be non-compositional, with the whole vector representing the concept of a red apple. Alternatively, the vector could encode other concepts typically ignored by people in their descriptions of such an object (Perkins, 2021), such as the presence of a curved stem. The receiver processes this vector, and has to decide which object is being described from the three objects it can observe. Assuming that the concept of a red apple is encoded in the vector [5,10,6], and the receiver can recover this information from this compressed representation, the receiver correctly identifies the red apple as the target. The referential game would then continue to a new round, with different target and distractor objects.

2.5.2 Werewolf

In Werewolf, players are divided into two teams: werewolves and villagers, typically with a ratio of one werewolf to three villagers. The game alternates between two phases: nighttime and daytime. During the nighttime, werewolves secretly communicate and decide on a villager to eliminate. When daytime arrives, all players, including the werewolves, discuss and vote to eliminate someone they suspect to be a werewolf. The objective for the werewolves is to eliminate villagers without being discovered, using

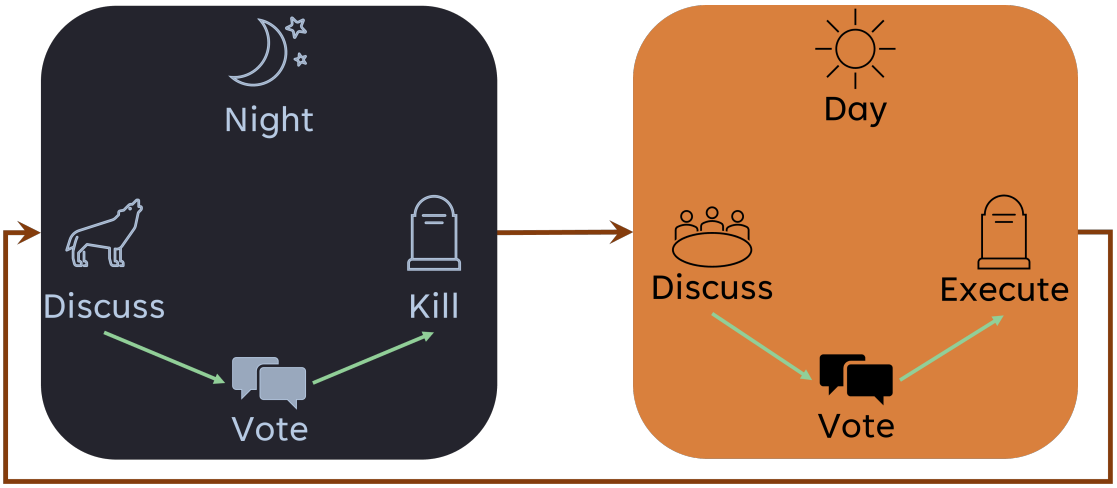
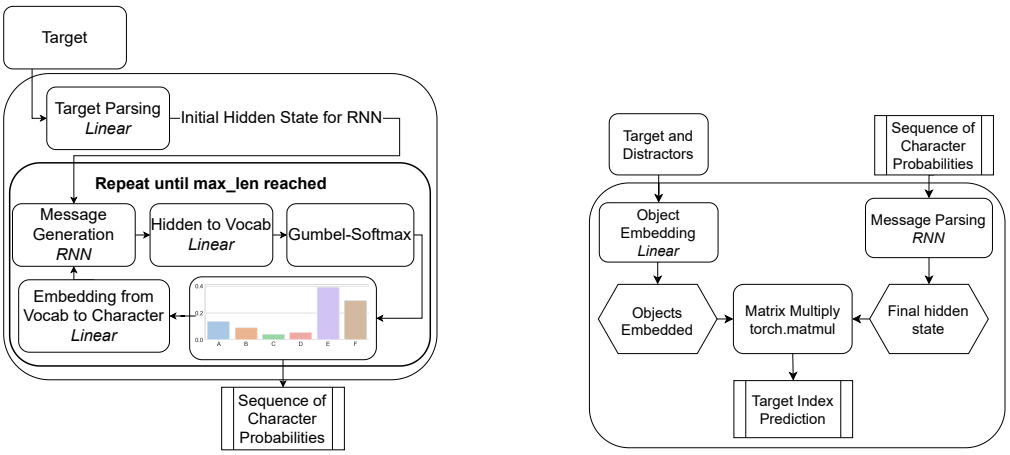


FIGURE 2.2: Visual representation of the flow of the game of Werewolf.

deception and manipulation during the day discussions. Villagers aim to identify and vote out the werewolves based on the behaviours and clues observed during the discussions. The game continues through these cycles of night and day until either all werewolves are eliminated or the werewolves outnumber the villagers, resulting in a victory for the respective team. We provide a visual representation of the game phases in Figure 2.2.

For example, in a four-player game of Werewolf, there are three villagers (players 1, 2, and 3) and one werewolf (player 4). During the first nighttime phase, player 4 (the werewolf) decides to eliminate player 2. In the following daytime phase, the players discover that player 2 has been eliminated, leaving three players (1,3,4). During the daytime phase, players 1 and 3 discuss who they suspect might be the werewolf. Player 4 tries to blend in and deflect suspicion. After deliberation, players 1 and 4 vote to eliminate player 3, believing them to be the werewolf. With only players 1 and 4 remaining, the game continues to nighttime. Player 4 eliminates player 1, resulting in a victory for the werewolf.

The game of Werewolf has garnered some interest in the ML community recently, with the launch of the AI Wolf Competition (Toriumi et al., 2017). The competition, however, relies on pre-defined structures for agent communication, and does not allow for the emergence of other protocols. With only one work published (Brandizzi et al., 2021) on emergent protocols in Werewolf, this presents an important research gap. Achieving any significant win rates Werewolf could require complex strategies, coordination, and communication protocols, making it a useful environment for EC research.



(A) EGG Sender Architecture

(B) EGG Receiver Architecture

FIGURE 2.3: The base architecture of EGG agents (Kharitonov et al., 2019).

2.6 Agent Architectures

In EC the most used agent architecture is that of EGG (Kharitonov et al., 2019). This basic architecture consists of sender and receiver agents, constructed around a single RNN, such as a GRU (Cho et al., 2014) or an LSTM (Hochreiter and Schmidhuber, 1997). First, the sender (Figure 2.3a) processes the target object, using either an RNN or a Linear layer. The result is the initial hidden state for the message generation RNN. The messages are generated character by character, using the Gumbel-Softmax trick (Jang et al., 2017). These messages are then passed to the receiver, an overview of which is shown in Figure 2.3b. The receiver's architecture contains an object embedding linear layer and a message processing RNN. The output of both the embedding layer and the message RNN is combined to create the referential game object prediction. We present a visual representation of the EGG architecture for both the sender and the receiver in Figure 2.3.

Other architectures have also been proposed, including Variational Autoencoders (Ueda and Taniguchi, 2023), and attention-based architectures (Vaswani et al., 2017; Ri et al., 2023). Initial results show significant promise, with the attention architecture creating more compositional languages (Ri et al., 2023). Interestingly, combining an attention mechanism with the more common RNN performs even better than a purely attention-based approach (Ri et al., 2023). While transformers show great promise in many areas (OpenAI et al., 2023), they still struggle with compositional generalisation (Yin et al., 2024; Kim and Smolensky, 2024), which humans can do easily (Kim and Smolensky, 2024). This could potentially point to the best architecture being a combination of attention-based mechanisms together with RNNs, especially with the modified LSTM architectures even outperforming transformers (Merrill et al., 2024; Merrill and Sabharwal, 2022; Deletang et al., 2022), which have been shown to be less

powerful than LSTMs (Beck et al., 2024). With new RNN architectures such as the xLSTM, the RNN-based models could be competitive with LLMs (Beck et al., 2024).

2.7 Agent Optimisation

The three main techniques that are used in emergent communication are REINFORCE (Williams, 1992), Gumbel-Softmax (Jang et al., 2017), and classic backpropagation (Schulman et al., 2015). Some approaches also use a combination of these optimisation techniques to achieve better performance or faster convergence rates.

The main problem encountered when implementing EC systems is that direct backpropagation and differentiation is usually impossible. As the messages passed between agents are often composed of discrete tokens, the gradients for backpropagation cannot be computed for those discrete token choices. The simplest solution to this problem comes with REINFORCE (Williams, 1992). The REINFORCE approach introduces rewards to agents so that they can learn from the positive or negative results of their message choices, instead of back propagating the referential game error directly.

In combination, REINFORCE (Williams, 1992) is often used for the sender, and backpropagation for the listener. The objective of the listener can often be viewed as a simple classification task, as it only needs to map the discrete input it receives to the correct output. For example, if we assume that the sender is a static dictionary, then the listener can be viewed as learning in a supervised way a mapping of all items in a dictionary to all objects that have that meaning. The combined approach means that the more computationally expensive technique (REINFORCE) can be limited to only one agent, therefore increasing the training performance of the whole model (Chaabouni et al., 2019).

The **Gumbel-Softmax** technique (Jang et al., 2017) can be viewed as both continuous and discrete, as it "converts" the emergent language from a discrete to a probabilistic continuous representation so that it can be differentiated and then backpropagated. At training time, the discrete symbols are approximated to a continuous vector, with a concentrated peak around the value of the discrete character (Chaabouni et al., 2021; Jang et al., 2017). This discretisation trick allows for the training speed to come close to the continuous communication, where the gradients can flow freely, while the communication being discrete at test time allows for linguistic analysis.

While the Gumbel-Softmax technique (Jang et al., 2017) has gained significant popularity in the field of EC, there exists a significant limitation to its possible applications. It allows the gradients to flow through both the sender and the receiver, and so the two agents can be treated as one neural network, similar to autoencoder

architectures (Kramer, 1992). This makes it harder to scale this technique to multiagent settings. Converting multiple agents to virtually a single network is challenging, and becomes impossible when the agents can act at different time steps. This is where pure reinforcement learning techniques, bypassing the issue of direct gradient flow, will gain an advantage, as scaling is being pursued in EC (Chaabouni et al., 2022).

2.8 Conclusions

This literature review has explored the evolution and current state of research in Emergent Communication, emphasizing the foundational linguistic concepts such as pragmatics and semantics, which inform the study of emergent languages. Pragmatic elements like deixis and anaphora, alongside semantic principles, especially compositionality, are crucial for understanding how agents can develop and use language, while providing new directions for future research in EC.

While progress has been made, especially in understanding properties like compositionality and efficiency (Chaabouni et al., 2019, 2020; Auersperger and Pecina, 2022), there is still much to explore within simple environments. Referential games (Lazaridou et al., 2018), despite their simplicity, continue to offer a controlled and effective setting for studying various aspects of EC. These environments allow isolation and analysis of specific language properties without the confounding factors present in more complex settings. As such, they remain valuable for foundational research and for testing new hypotheses about language emergence and agent communication strategies.

The review also highlights the challenges in quantifying the properties of emergent languages, pointing out that current metrics often fall short in capturing their full complexity. There is a need for metrics that are more linguistically inspired and less reliant on extensive dataset labelling (Rita et al., 2024). Techniques such as concatenability (Bosc, 2022) and entropy-based segmentation (Ueda et al., 2023) show promise but also underscore the need for further research. Future studies should focus on developing metrics that draw from linguistic theories and methods, ensuring a closer alignment with how natural languages are studied and understood (Boldt and Mortensen, 2024b; Rita et al., 2024). The insight we may gather from such developments in EC may also lead to new questions being asked in linguistics (Galke and Raviv, 2024). Metrics inspired or borrowed from the study of human language may also help bridge the gap between human and emergent languages (Rita et al., 2024).

In addition to refining metrics, the interpretation of emergent languages should move towards more automated methods rather than relying heavily on manual dataset labelling. Automated interpretation techniques, such as those involving mutual information and bipartite graph mappings (Carmeli et al., 2024), offer a way forward. These methods can facilitate a more scalable and efficient analysis of emergent

languages, enabling researchers to decode and understand agent communication with less manual intervention.

Finally, the temporal dynamics in emergent communication could play an important role in the future. Fundamental aspects of natural languages like spatial or temporal deixis serve a crucial purpose of bridging language with the physical, temporal, and social world (Stapleton, 2017). We consider that this avenue deserves close attention to investigate how deixis emerges and functions within EC, as it holds the potential to significantly enhance the richness and applicability of agent communication strategies.

Overall, the field of emergent communication offers a wealth of opportunities for continued interdisciplinary research. Combining insights from computational linguistics and machine learning will be essential for advancing our understanding and capabilities in this field, leading to more intelligent and effective communicative multiagent systems.

Chapter 3

Interaction Time in Dialogue

Building on the key concepts discussed in Chapter 2, in this chapter we examine the impact of interaction time on the development of emergent communication in multi-agent systems. We extend the framework of the Werewolf game, previously explored by Brandizzi et al. (2021), to examine how varying the amount of interaction time influences agent strategies and language efficiency.

3.1 Emergent Communication in Werewolf

Emergent communication has been used to improve the performance of intelligent agents in the game of Werewolf (Brandizzi et al., 2021), a well-known party game, also sometimes referred to as Mafia. The authors show that agent win rate significantly improves when the agents create their own language, however the analysis of the language and strategy of the villagers is left for future work. In the original environment (Brandizzi et al., 2021), the villagers only had a single round of communication to agree on a player to vote out as an alleged werewolf, which may not allow them to fully explore other, possibly better, strategies.

Investigating the amount of time allocated for communication allows us to analyse how extended discussions influence the effectiveness of the agents' strategies, potentially leading to more refined and successful approaches. By observing how agents interact over different timeframes, we can gain a deeper understanding of how time constraints impact the development of shared languages and cooperative behaviour. This work can provide generalizable knowledge applicable to enhancing teamwork in automated systems, and optimizing communication protocols in complex, multi-agent environments.

We extend the Werewolf environment by introducing multiple rounds of communication, to determine the effect of a longer period of discussion on the language

developed and the success rate in identifying the werewolf. Our work is motivated by the assumption that if agents are allowed more time to communicate, they will develop successful strategies faster. We introduce a voting threshold to encourage villagers to work together and to vote uniformly. Both of these changes tell us how much the time needed to communicate influences the agents' performance.

We show that the agents develop a highly successful strategy of password signalling to win the game, where their communication behaves like a Turing test (Turing, 1950) analogue for identifying the werewolves. The multiround aspect increases convergence speed and allows the agents to establish this identity test more rapidly. Finally, we describe the effects of the voting threshold regarding the win rate of the villagers, as well as convergence speed.

3.2 Werewolf Environment

The agent environment is based on the work of Brandizzi et al. (2021). The environment game flow follows that of the game of Werewolf (Section 2.5.2), with alternating nighttime and daytime phases with some minor changes. Firstly, the "daytime" phase consists of one round of communication, where agents are allowed to exchange a single message (Brandizzi et al., 2021). Secondly, the game in these environments ends when all werewolves are voted out, or when the number of villagers and werewolves are equal. This condition is included because when the numbers are equal on each team, the villagers can no longer win.

The observations the agents receive consist of the day count, the status map, the game phase, the voting targets, agent ID, and messages. The day count allows the agents to understand the number of days that have passed in the game. The status map shows which agents are still in the game, and which have been eliminated. The game phase indicates which of the four game phases the agents are currently in: the nighttime communication phase; the nighttime elimination phase; the daytime communication phase; or the daytime elimination phase. The voting targets allows each agent to see the final vote of all other agents. The agent ID represents the ID of each agent, which is the ID is used during voting. Finally, the messages represent all messages sent by all agents during a given phase.

The action that each agent can take is to produce a message and a vote for each round of communication. The message is represented by an array of integers, which can be selected by the agents from a given vocabulary. The length of this array is determined by the message length parameter. No other actions can be taken by the agents. After a successful vote, be it by the werewolves during the nighttime, or all players in the daytime phase, the player voted out is removed from the game, and may take no further action. The status map is updated to reflect this.

Messages are passed during all phases as an observation to the agents. Depending on the agent's role, it can receive both the nighttime and daytime observations, if it is a werewolf, or purely the daytime observations, if it is a villager.

The reward scheme for the villagers seeks to incentivise specific behaviours (Brandizzi et al., 2021). The reward values are: -5 for death, to incentivise avoiding being voted out/eaten by a werewolf; $+25$ for winning the game, to incentivise winning strategies; -25 for losing the game so that the agents avoid any losing strategies; -1 for picking a target besides the one that was voted out, to reinforce uniform voting.

To test our hypotheses (Section 3.1), we extend the Werewolf environment to a multi-round communication domain and introduce changes to how the voting works to incentivise quicker convergence on communication between the villagers[1].

Our multi-round approach is implemented during the daytime phase when all agents communicate. We choose only the daytime phase, as the werewolves have a static policy (Brandizzi et al., 2021) (Section 3.3), meaning communication during the nighttime phase has no effect on the werewolves' vote. This modified game can be parameterised with the number of communication rounds, n_r, varying the amount of time agents have to converse. More precisely, it is the number of times that the agents can exchange a single message between all of them. The number of rounds is distinct from the number of episodes which relates to the number of *full games* played by the agents, where full games refer to a game which has reached a win condition, and the environment was then reset.

We incentivise our agents to decide quickly on the target of their vote, while also maintaining a high consensus rate among them as to who their target will be. This is done through the agreement loss, as presented by Brandizzi et al. (2021), as well as our additional loss based on the number of rounds taken to reach a conclusion. We introduce a negative reward of -2 for wasting a round, or not reaching the voting threshold, t_v, to further incentivise voting in unison. The agreement loss penalises agents who voted for a target that was not voted out, while t_v represents the minimum percentage of agents that must agree on a target for the vote to be considered valid. This means that, when less than t_v% of agents agree on a target, then the vote is considered invalid, and no agents are eliminated. This modification can be viewed as adding an independent judge to count all agent votes, where only a significant plurality is permitted to decide whether to vote out an agent or not. This contrasts with the original environment, which only required a simple plurality of agents to choose a target, even if that plurality amounted to just two out of 21 agents, and when no plurality could be reached, an agent would be removed at random.

[1]Our code is available on GitHub at https://github.com/olipinski/rl_werewolf

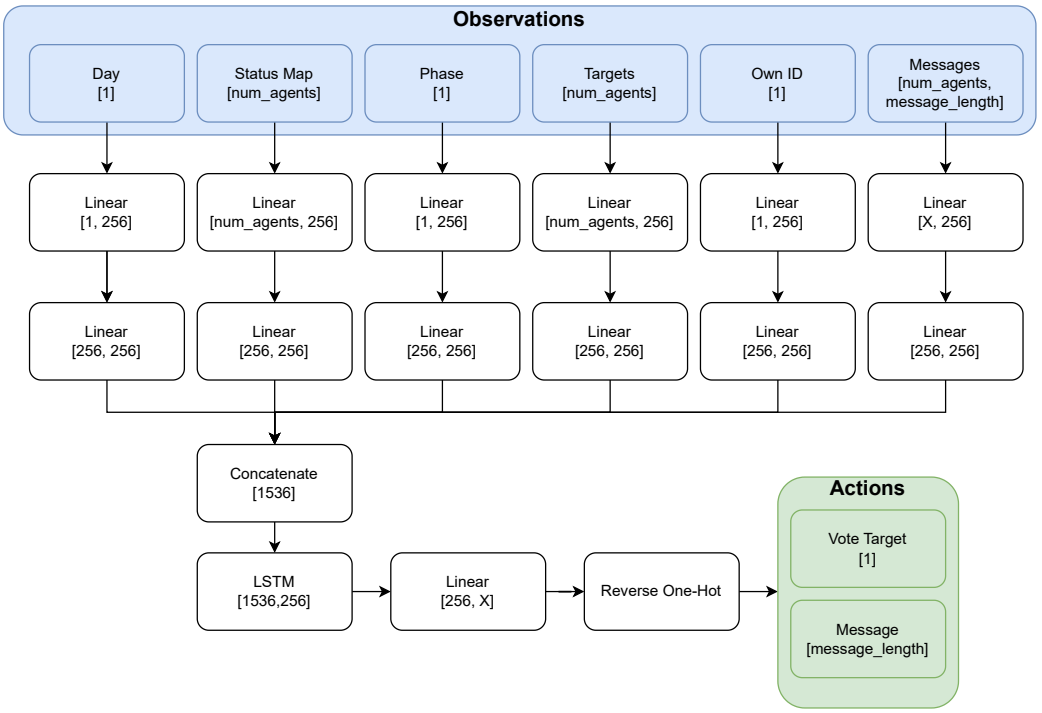


FIGURE 3.1: Werewolf agent architecture.

3.3 Architecture

The architecture of Brandizzi et al. (2021), and by extension of this work, follows the default agent architecture of Ray RLLib (Liang et al., 2018; Moritz et al., 2017). We present an overview of the agent architecture in Figure 3.1. As the number of agents and message length are variables, we do not present the sizes of the layers which depend on their values, instead using the placeholder value *X*. These values are automatically calculated based on the runtime parameters.

The presented architecture, following Brandizzi et al. (2021), is used only for the villager agents. The werewolf agents instead follow a static werewolf policy (Brandizzi et al., 2021). This policy picks an agent at random to be voted out by the werewolves. All werewolves follow the same static policy.

Each agent observation (*i.e.*, Day Number, Status Map, Game Phase, Targets, Own ID and Messages (Section 3.2)) is first processed by two linear layers. The output of all linear layers is concatenated to a single vector, which is then passed to an LSTM. After the LSTM processes this input, it is passed to a final linear layer, which outputs a one-hot encoding of the chosen agent actions. This one-hot representation is then processed into actions which can be passed into the environment.

3.4 Interaction Time Experiments

The modified version of the environment exposes two additional parameters to explore: the number of rounds n_r and the voting threshold t_v. We explore them using a grid search, maintaining the other parameters from the original paper (Brandizzi et al., 2021). We provide details of the training and the grid search parameters in Appendix A.1.

3.4.1 Hypotheses

To investigate the influence the amount of time and voting plurality has on the agents' strategies, we pose two hypotheses that we test in this work:

Hypothesis 1 (H1) The longer we allow the villagers to communicate, the faster they will converge to a common communication strategy, and the higher their win rate will be.

Hypothesis 2 (H2) The more the agents are encouraged to vote in unison, the faster they will adopt an accurate voting pattern.

3.4.2 Convergence Speed

Increasing both n_r and t_v appears to decrease the average convergence episode, as shown in Figure 3.2 and Figure 3.3, with shaded areas representing the 95% confidence interval. We define the convergence episode as the episode number where our agents reach over 75% win rate, which we choose as an arbitrary threshold for a successful strategy. The convergence speed is defined as the average number of training episodes required to reach this 75% win rate.

To confirm these observations, statistical analysis on the impact that both the number of rounds and the voting threshold have on the convergence speed is performed, shown in detail in Appendix A.2. The results indicate that the number of rounds has a statistically significant effect on both win rate and convergence speed. We find that the higher the number of communication rounds, the quicker the agents converge, partially confirming our **H1**. The voting threshold, however, does not have a statistically significant effect, falsifying our **H2**, and so its impact on the convergence speed is not discussed further.

3.4.3 Win Rate

We analyse the impact that both of our additional parameters have on the average win rate of the villagers. Figure 3.4 and Figure 3.5 illustrate the interaction between the

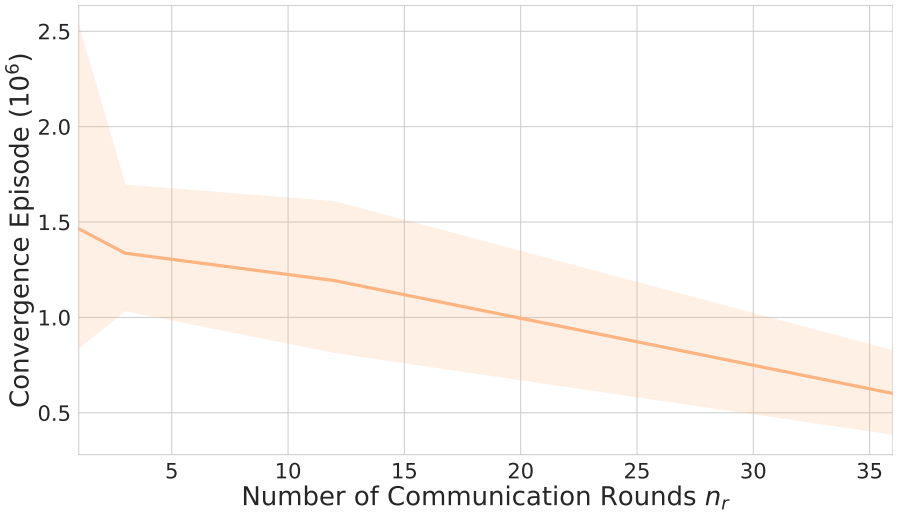


FIGURE 3.2: Impact of the number of rounds on the convergence speed.

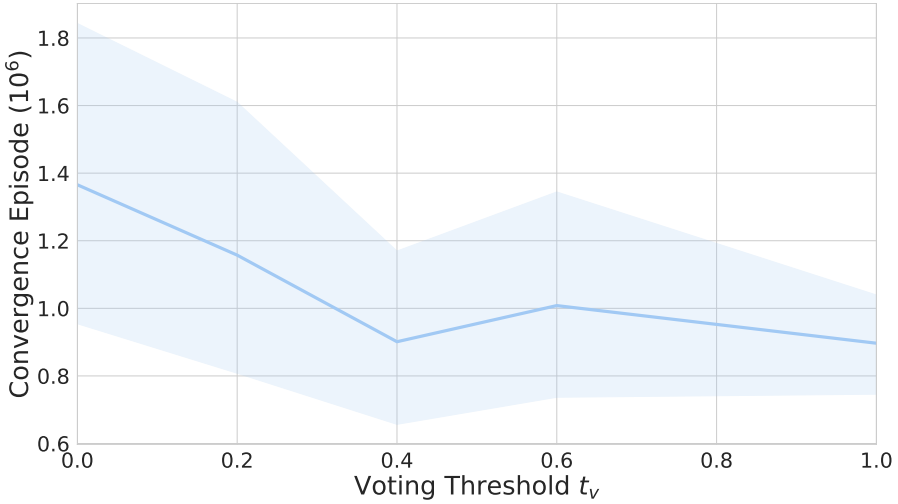


FIGURE 3.3: Impact of the voting plurality threshold on the convergence speed.

parameters n_r and t_v in terms of the win rate, with the shaded areas representing the 95% confidence interval. The win rate is defined as the percentage of games that the villagers win in a single training run. For certain configurations of the communication round count and voting threshold, our agents achieve over 95% win rate.

Performing the statistical significance analysis, outlined in Appendix A.2, we find that only the number of communication rounds has a statistically significant effect on the win rate. With a larger n_r, the win rates tend to be lower, as shown in Figure 3.5 indicating that increasing the number of communication rounds is negatively correlated with the average win rate. This partially falsifies our **H1**. However, the number of rounds together with the voting threshold increases the average win rate past the previously reported values by Brandizzi et al. (2021), shown in Table 3.1, and results in an overall positive trend.

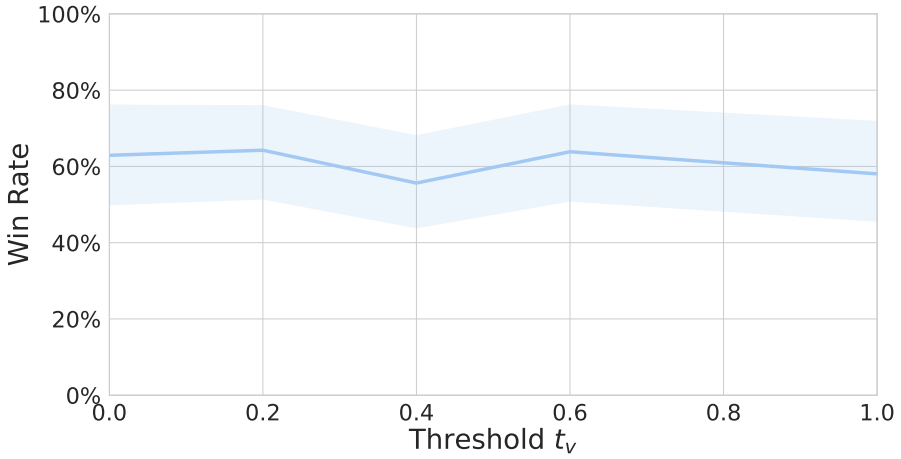


FIGURE 3.4: Impact of the voting plurality threshold on the win rate.

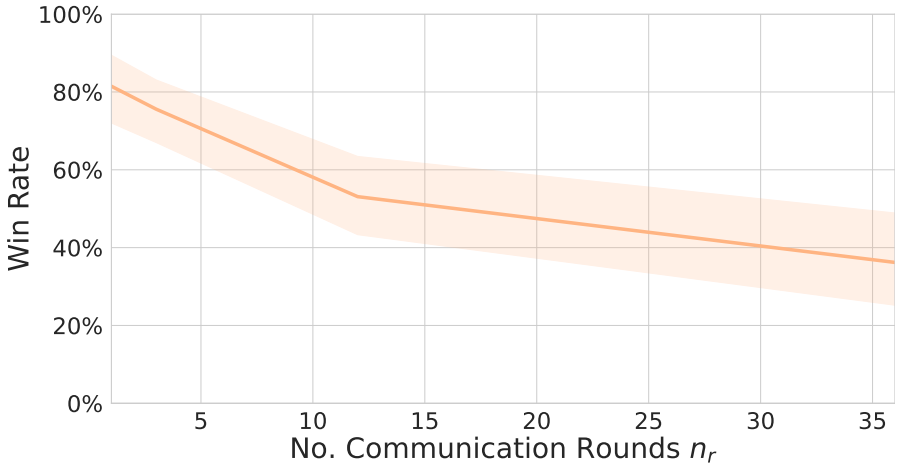


FIGURE 3.5: Impact of the number of communication rounds on the win rate.

3.4.4 Comparison to the Original Environment

In Table 3.1 the results from this work are compared to the original experiments by Brandizzi et al. (2021). Our configuration notation is consistent with Brandizzi et al. (2021), where "SR" is vocabulary size, or the number of characters available for each position in the message (originally called Signal Range), "SL" is message length (originally called Signal Length), "PL" is the number of players, "WR" is "Win Rate", and "TH" and "RS" being the threshold value and number of rounds respectively. Lastly, the "Convergence" column refers to the episode number, or convergence point, that the agents with that configuration achieved. All values are reported for the best run of each configuration, with better ones (as compared between the original environment and our modified version) displayed in **bold**. The mean win rate together with the 1-sigma standard deviation, $\pm$, are reported in the parentheses. The convergence episode for the original configurations is reported as obtained by our reproduction.

Table 3.1 shows that the introduced modifications result in a lower total episode count before convergence for two out of three compared configurations, while also achieving

TABLE 3.1: Results for both our and the original Brandizzi et al. (2021) environment.

Configuration	TH	RS	Our WR (%)	Original WR (%)	Convergence (10^6)
SL9-SR2-PL9	1	36	**100** (60 ± 31)	N/A	**0.70** (0.94 ± 0.84)
SL9-SR2-PL9	0	1	99 (78 ± 32)	45	1.29 (1.57 ± 0.41)
SL9-SR2-PL21	1	3	96 (83 ± 13)	N/A	**0.70** (1.47 ± 0.42)
SL9-SR2-PL21	0	1	95 (78 ± 27)	**98**	1.19 (1.28 ± 0.59)
SL21-SR2-PL21	0.4	3	**100** (80 ± 17)	N/A	0.62 (0.93 ± 0.19)
SL21-SR2-PL21	0	1	98 (92 ± 13)	94	**0.58** (0.72 ± 0.12)

a higher win rate for two out of the three configurations. This demonstrates that allowing the agents to communicate for longer results in a possible improved performance for both metrics of the game. Reproductions of the configurations that were presented by Brandizzi et al. (2021) are also included, with our modifications to the code for a better comparison. We note that our reproductions have higher win rates than reported, which could mean that the original runs were not fully converged, possibly owing to the time or compute power available to the original study. Nevertheless, our results still improve over the reproductions.

3.4.5 Language Analysis

To analyse the emergent language, we gather the information about word usage from the latter parts of the training of our agents, focusing on agents who have completed the exploration phase, and developed a stable language. We consider this point when the win rate improvements plateau and there is minimal variation in the agent policies.

Analysing the language that agents have developed, we find a focus on a sparse word vocabulary for the successful strategies, while most unsuccessful strategies have multiple words in their vocabulary. Moreover, almost all successful agent populations use a single word at least 90% of the time, with a minor number of outliers.

We can see the usage of the most common message in Figure 3.7. We observe the same correlation when including the most common message and its distance-one adjacents, defined as any message that has a single character difference from the most common message, in Figure 3.6. Both figures are produced using a rolling average of the usage percentages, to reduce the noise in the plot, and increase the visibility of the common trends. However, the rolling average does remove the outliers at the higher win rate

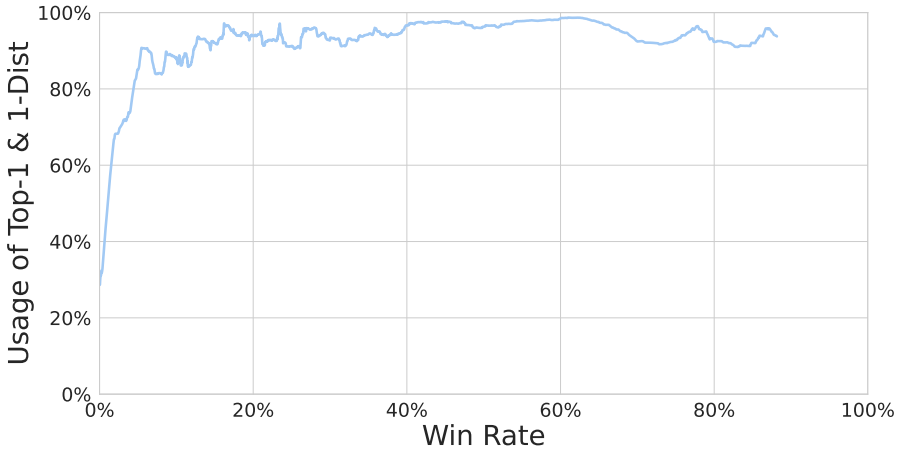


FIGURE 3.6: Most used unique message **and** its top ten distance-one adjacents versus the villager win percentage.

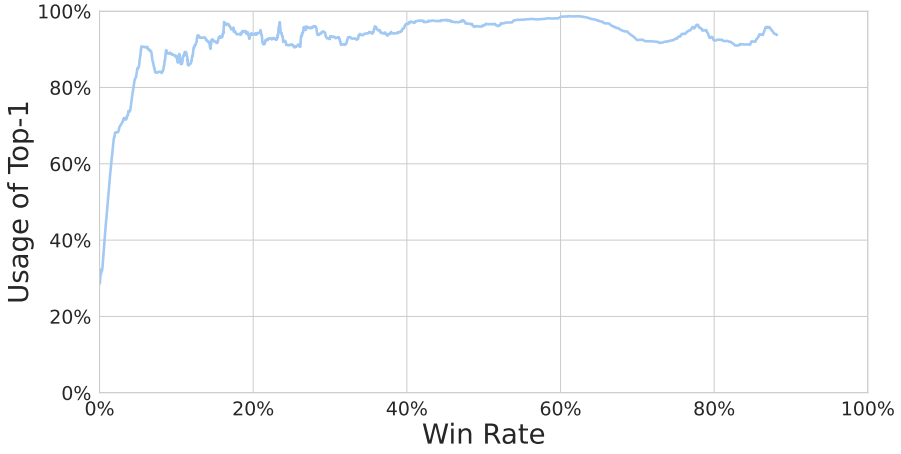


FIGURE 3.7: Most used unique message versus the villager win percentage.

part of the plot, therefore limiting the plot to about 90%. The positive correlation observed in both figures shows winning strategies mostly use a single unique message, with some exploration of adjacent messages (i.e., using $[1, 1, 1, 0]$ instead of previously used $[1, 1, 0, 0]$).

3.5 Discussion

Brandizzi et al. (2021) demonstrate that agents can develop communication strategies that improve upon the calculated theoretical baseline win rates for random policies using a single round of communication. We investigated the impact of permitting agents to engage in multiple rounds of communication during each episode, and constraints on level of voting "agreement". Our agent and game configuration significantly improves over both the theoretical baseline win rates and the previously demonstrated performance (Brandizzi et al., 2021) (Section 3.4).

3.5.1 Strategy Analysis

During the game, villagers are observed to send the same message every round of communication and vote off those who do not comply with this strategy. As the agents mostly use very sparse, almost single-word vocabularies, we posit that our agents do not develop a compositional language, but rather a kind of password, similar to a degenerate language (Section 2.3.1). We theorise that this is because expanding their vocabulary would not bring an improvement to their performance. With such sparse vocabularies, they can already achieve a high success rate, without needing to develop more complex languages. As the password strategy relies on very simple communication, we assume that it is straightforward to reach in the global loss landscape, which is why it is the most adopted strategy. We additionally observe no difference in performance or adoption of this strategy, when accounting for different message lengths or ranges (Table 3.1).

Following the original game setup of Brandizzi et al. (2021), the werewolves use a static policy, allowing villagers to easily distinguish the alternative agent type, as the werewolves can never adapt their communication to their interlocutors. For a single communication round, the agents at the start of the game have no information about who the werewolves may be, and so would need to vote for a random player. Instead, with multiple rounds, they can establish this distinction without having to cast the final vote. We consider this strategy to be of particular interest, as we believe it resembles that of the Turing Test (Turing, 1950), with it being performed by multiple separate intelligent agents, with no involvement of humans. We discuss the limitations of the static policy approach, together with experiments where werewolves are allowed to adapt, in Section 3.6.

3.5.2 Convergence Speed

The number of communication rounds has a statistically significant (Appendix A.2) effect on the convergence speed. The more rounds the agents can converse for, the quicker they can converge on a common strategy. We hypothesise this is due to the increased potential for information exchange and strategy alignment. When agents can communicate more, they can better refine their understanding of the game dynamics and the behaviour of other agents. This iterative process of communication and adjustment leads to a faster alignment of strategies, reducing the time needed for convergence.

The voting threshold, however, does not have a statistically significant effect on the convergence speed. The voting threshold was intended to incentivise faster convergence in the agents, by penalising agents that vote for different agents each round. The negative result for the voting threshold may be explained by the voting threshold being

too close in function to the agreement reward, as implemented by Brandizzi et al. (2021). Essentially, both mechanisms might serve similar purposes in guiding agent behaviour, leading to redundancy and thus diminishing the distinct impact of the voting threshold.

Another possible reason for the lack of impact from the voting threshold is that agents may not require such a penalty to converge effectively. If the communication rounds themselves provide sufficient incentive, additional mechanisms like the voting threshold might not contribute significantly to the convergence process.

3.5.3 Win Rate

The number of communication rounds also affects the win rate, which is confirmed to be statistically significant (Appendix A.2). Increasing the number of communication rounds impacts the win rate negatively. We believe this decrease is due to the longer training times, as the larger number of rounds requires more computation, and therefore does not achieve convergence within the same time as our other configurations. We can see that the number of total training episodes decreases as the number of rounds increases, which is due to the limited amount of time that we could run the simulations in Figure 3.8. We hypothesise that this is the reason behind lower performance as the number of rounds increases. As the agents need to exchange more messages in configurations with higher number of rounds, they do not have enough time to converge to a winning strategy; hence they perform worse. This observation suggests that while additional communication rounds can facilitate strategy refinement and convergence, practical constraints such as computational resources and training duration limit must be considered.

The voting threshold does not have a statistically significant effect on the win rate of the villagers, for reasons we discuss in Section 3.5.2. This further supports the notion that the increase in the number of communication rounds is sufficient in guiding agent behaviour towards convergence and effective strategy formation.

3.5.4 Failure Modes

We observe that, in one case, our agents perform worse than those of the original study (Brandizzi et al., 2021). Our results show a higher convergence speed for the 9-SL settings, while a lower, but similar, convergence speed for the 21-SL setting. We hypothesise that this is due to the larger message lengths in that setting. As agents can communicate more information per round, the advantage of a larger number of rounds may decrease. This also points to an interesting finding, where the convergence speed may be tied to the total amount of information exchanged in a single round, be it through more interaction time or longer messages. However, we consider our approach

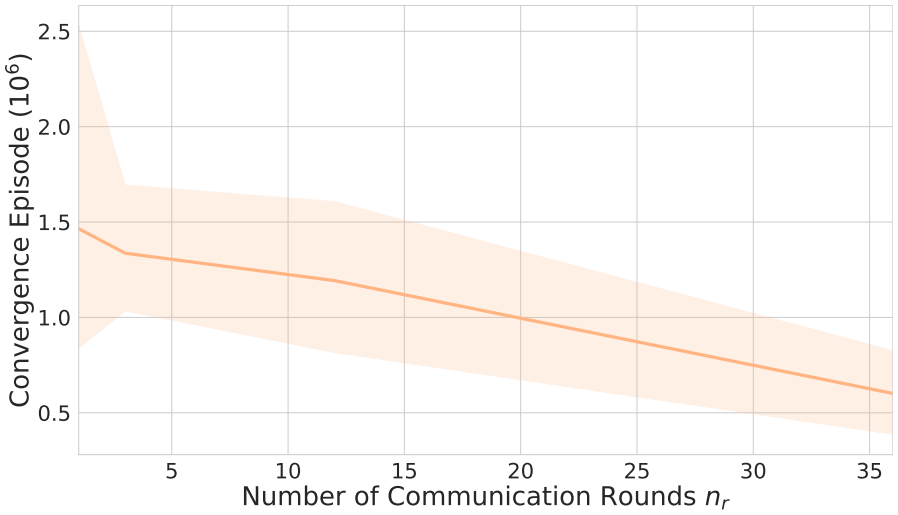


FIGURE 3.8: Impact of the number of communication rounds on the number of training episodes.

of increasing the amount of time, rather than message length, more robust. As message length increases, any corruption in the message transmission could potentially significantly affect the message content (Kucinski et al., 2021; Ueda and Washio, 2021), especially if the messages are information-dense and do not have built in error correction.

3.6 Limitations

The main limitation of this work is the use of static policies for the werewolf agents. We use that static policies to provide a faithful reproduction and comparison to the work of Brandizzi et al. (2021), which also used static policies. However, it does preclude the werewolves from adapting to the villager strategies. While the results presented hold with this static policy approach, we do consider that if the werewolves were able to learn, they would be able to counter the identified password signalling strategy.

Brandizzi et al. (2021) also explore the idea of werewolves having a trainable policy. They identify that this would lead to an unnecessary increase in complexity and decrease in stability of the learning, clouding the goal of studying the emergent language between cooperative agents. We ran experiments with both villagers and the werewolves being able to learn, finding an equilibrium of 50% win rate for either team. This may be due to the lack of stability in the learning, as we find no signs of an explicit strategy emerging from the villagers. These experiments are therefore excluded from this work.

3.7 Conclusions

We introduce multi-round communication to the originally single round environment of Werewolf (Brandizzi et al., 2021). We show that the number of communication rounds decreases the convergence time of the agents, with statistical analysis showing that this correlation is significant. We investigate the strategies that the agents develop to achieve the high win rates, and demonstrate that our agents are using password signalling to allow villagers to efficiently identify each other. Our results confirm that allowing agents to communicate for longer offers improvements to the way that the agents can play the game. With more time, agents can create more efficient strategies, while also having an efficient communication protocol. Our findings underscore the significance of extended communication periods in enhancing agent performance and language development. The strategies that emerge from these extended interactions, such as password signalling, illustrate the potential for more sophisticated and efficient communication protocols stemming from investigating temporal dynamics in EC.

Chapter 4

Temporal References

Building on the exploration of communication time in Chapter 3, in this chapter we investigate another crucial aspect of emergent communication: *temporal deixis* or temporal references (Section 2.1). *Deixis* (Section 2.1), has been described as a way of pointing through language. Examples of *temporal deixis* include words such as "yesterday" or "before" (Lyons, 1977). In this chapter, we examine how agents develop the ability to use deixis to refer to past events, and the architectural modifications necessary to facilitate such references.

4.1 Temporal References in Emergent Communication

Many aspects of emergent language have been explored (Lazaridou and Baroni, 2020; Boldt and Mortensen, 2024b), with a particular focus on improving communication efficiency (Rita et al., 2020; Chaabouni et al., 2019; Kang 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 offered 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).

Despite these advances, no existing research has investigated or reported on the emergence of temporal referencing strategies, where agents can communicate relationships between different time steps. Including temporal references, alongside the general characteristics of emergent languages, promises to enhance agents' bandwidth efficiency and task performance across a variety of scenarios.

As research in emergent communication continues to advance, increasing the complexity of environmental settings is becoming a crucial step to better simulate real-world scenarios (Chaabouni et al., 2022). This scaling often involves introducing

more variables, dynamic elements, and multi-agent interactions, which make communication between agents more challenging, while also becoming more reflective of the communication strategies observed in humans. In these complex environments, the ability of agents to reference temporal relationships, such as identifying the sequence or timing of events, becomes particularly valuable.

Temporal references are essential because they allow for more sophisticated strategies and better coordination. For instance, in environments where actions need to be planned based on previous outcomes or where understanding the order of events is crucial, agents equipped with the ability to use temporal references can significantly improve their performance. One example is social deduction games (Brandizzi et al., 2021; Lipinski et al., 2022; Kopparapu et al., 2022), where players must infer others' intentions or past actions to succeed. In such games, the ability to accurately refer to past events is a key component of a winning strategy, as it allows agents to deduce hidden information, predict opponents' behaviour, and coordinate with allies more effectively.

Temporal referencing also facilitates the development of more efficient communication protocols among agents. By assigning shorter, more efficient messages to frequently occurring events, agents can streamline their communication, reducing the cognitive load and bandwidth required for successful interactions. This concept is analogous to Zipf's Law in human languages (Zipf, 1949), which observes that the most commonly used words tend to be shorter (such as "is" or "am" versus "neuroscience" or "embroidery"), allowing for quicker and more efficient communication. In emergent language systems, similar patterns can evolve, where agents naturally develop a shorthand for common events, leading to optimized communication that is both concise and effective. Such efficiency gains are critical in complex environments where rapid and accurate communication can mean the difference between success and failure.

Temporal references 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 the emergent language, optimizing communication.

Our contribution lies in examining when temporal references emerge between agents. Three potential prerequisites are explored: environmental pressures, external pressures and architectural changes. The agents are trained in both the regular referential game (Lazaridou et al., 2017) and on an environment which encourages the development of

temporal references through embedded environmental pressures (Section 4.2.3). The effect of an external pressure to develop temporal referencing is explored via an additional loss applied to the agents (Section 4.5). Three types of architecture are evaluated, (Section 4.3), analysing two novel architectures together with a reference architecture based on the commonly used EGG (Kharitonov et al., 2019) agents. The baseline *Base* (Section 4.3.1) agent, 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 4.3.2) agent, which features a sequentially batched GRU, instead of the parallel batching used in EGG, which allows the agents to build an understanding of the sequence of target objects. Additionally, the *TemporalR* (Section 4.3.3) agent combines the information from the sequential GRU and the parallel batched GRU from the *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.

4.2 Temporal Referential Games

4.2.1 Definitions

In referential games (Section 2.5.1), agents need to identify objects from an *object space* V, which appear to them as attribute-value vectors $\boldsymbol{x} \in V$. To define the *object space* V, the *value space* of all possible attributes is defined as $S = \{0, 1, 2 \dots N_{val}\}$ where N_{val} is the *number of values*. The value space represents the variations each object *attribute* can have. 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* of an object.

To give an intuition of the notion of attributes and values, consider that the object shown to the sender is an abstraction of an image of a 2-D shape, shown in Figure 4.1. The attributes of the shape could include the colour of the background or what shape it represents. The values are the variations of these attributes. In this example, possible values of the background colour are red, green, or blue, which are indicated by integers $[0, 1, 2]$ respectively. The shape is represented by integers $[0, 1, 2]$, indicating a triangle, square, and circle. The colour of the shape is in the second position of the vector, while the shape is in the first. To represent a blue circle, a vector [circle, blue] is used, which would be represented as an integer vector $[3, 2]$, where 3 represents a circle, and 2 represents the colour blue.

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

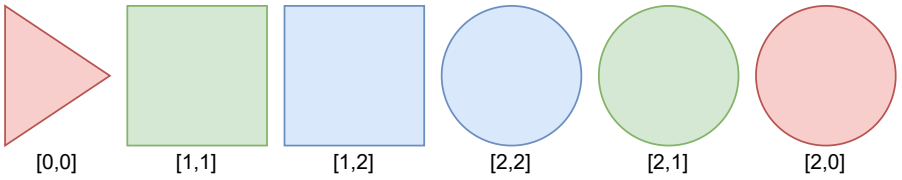


FIGURE 4.1: Attribute-Value Object Representation

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.

Combining the message and object space, the agents' language is defined as a mapping from the objects in V to messages in ξ. Finally, the exchange history, representing all messages and objects that the agents have sent/seen so far, is defined as 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 signifying the episode of the last exchange.

4.2.2 Temporal Logic

Temporal logic is used to formally define the behaviour of our temporal referential games. To achieve this, a form of Linear Temporal Logic (LTL) (Pnueli, 1977) called Past Linear Temporal Logic (PLTL) (Lichtenstein et al., 1985) is employed.

LTL focuses on the connection between future and present propositions, defining operators such as "next" $\bigcirc$, indicating that a given predicate or event will be true in the next step. The LTL operators can then be extended to include the temporal relationship with propositions in the past, creating PLTL. PLTL defines the operator "previously" $\ominus$, corresponding to the LTL operator of "next" $\bigcirc$.

The "previously" PLTL operator must satisfy Equation (4.1), using the definitions from Maler et al. (2008), where σ refers to a behaviour of a system (the message sent by an agent) at time t, and ϕ signifies a property (the object seen by the agent).

$$(\sigma, t) \models \ominus\phi \leftrightarrow (\sigma, t-1) \models \phi \tag{4.1}$$

Additionally, the shorthand notation of $\ominus^n$ is used, signifying that the $\ominus$ operator is applied n game steps back. For instance, $\ominus^4\phi \leftrightarrow \ominus\ominus\ominus\ominus\phi$.

4.2.3 Temporal Referential Games

The temporal version of the referential games (Lewis, 1969; Lazaridou et al., 2017) 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 *history range* value, h_v, and the current episode, t.

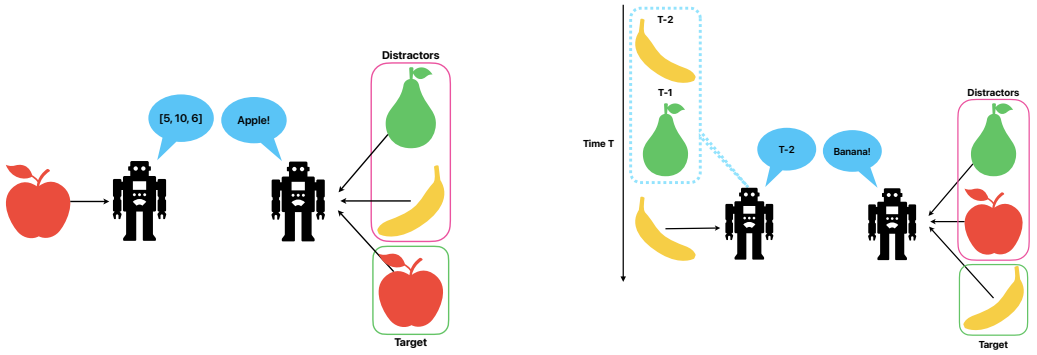
$$X(t, c, h_v) = \begin{cases} \boldsymbol{x} & c = 0 \\ \ominus^{h_v}\boldsymbol{x} = \tau_{t-h_v} & c = 1 \end{cases} \tag{4.2}$$

The *history range* value, h_v, is uniformly sampled, taking the value of any integer in the range $[1, h]$, where h is the *history length* hyperparameter. The *history range* value is randomly sampled to allow agents to develop temporal references of varying temporal lengths, instead of the parameter being fixed each run. The function $X(t, c, h_v)$ selects a target object to be presented to the sender using Equation (4.2), 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, with $p = 0.5$. 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 (4.2), 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 (4.2), a random target $\boldsymbol{a}$ is generated from $\boldsymbol{a} \in V$. The target sequence then becomes $[\boldsymbol{j}, \boldsymbol{k}, \boldsymbol{l}, \boldsymbol{a}]$.

This behaviour describes the environment *TRG*, which represents the base variant of temporal referential games, where targets are randomly generated with a 50% chance of repetition of a target from the *history length* $[1, h]$. The *TRG Hard* variant is also used, which is a temporal referential game with the same 50% chance of a repetition, but where targets only differ in a single attribute when compared to the distractors. TRG Hard tests whether temporal referencing improves performance in environments where highly similar target repetitions are common. We expect that this environment will prove challenging for the agents, and so we expect an accuracy drop. However, we expect the accuracy to improve with the addition of temporal references, as the agents should be able to overcome the target similarities by referring to the temporal relationships between the targets instead. A visual representation of the TRG environment is shown in Figure 4.2b, with the original referential game presented in Figure 4.2a.

[1] Our code is available at `https://github.com/olipinski/TRG`



(A) Referential game (B) Temporal referential game

FIGURE 4.2: Structure of the referential game and temporal referential game.

Additionally, two more environments are used: *Always Same* and *Never Same*. Their purpose is to verify whether the messages that are identified as temporal references are correctly labelled. The *Always Same* environment sequentially repeats each target from a uniformly sampled subset of all possible targets ten times[2]. Repeating the target ten times allows verification that the messages are used consistently; *i.e.*, if the agents use temporal messaging. The *Never Same* never repeats a target and goes through a subset of all possible targets in order. The *Never Same* environment is used to verify if the same messages are used for other purposes than to purely indicate that the targets are the same. In both environments, the dataset only repeats the target object, while the distractor objects are randomly generated for each object set. Sample inputs and expected outputs for these environments are provided in Appendix B.2.1.

The agents are also trained and evaluated in the *RG* environment, which represents the classic referential game (Lewis, 1969; Lazaridou et al., 2017), where targets are randomly generated, and *RG Hard*, which is a more difficult version of the referential games, where the target and distractors only differ in a single attribute. The RG environment establishes a reference performance for the agents, while RG Hard determines whether temporal references enhance performance in an environment where targets are harder to differentiate. In common with the TRG Hard, we expect an accuracy drop in the RG Hard environment.

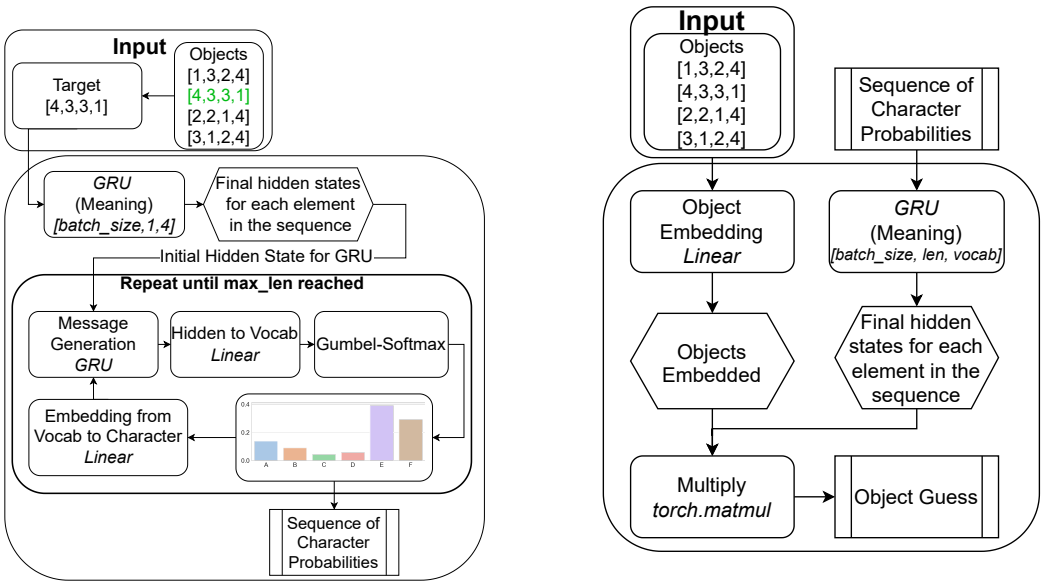
Since the RG, RG Hard, and Never Same environments have almost no target repetitions, (*cf.*, Appendix B.2, Figure B.1) we do not expect to observe the emergence of temporal references in these environments. We use RG, RG Hard, and Never Same to validate our results, and to provide a baseline performance on the environments usually used in emergent communication research (Boldt and Mortensen, 2024b).

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

4.3 Agent Architectures

As discussed in Section 2.6, both sender and receiver agents are typically built around a single recurrent neural network (Kharitonov et al., 2019). In this work, these standard agent architectures, based on a GRU (Cho et al., 2014), are compared to temporal GRU architectures, which use a different batching strategy. The *Base* agent, used as a baseline, is the commonly used GRU-based EGG agent (Cho et al., 2014; Kharitonov et al., 2019). Our two novel architectures, the *Temporal* and *TemporalR* agents, feature a sequentially batched GRU (Cho et al., 2014). This additional module allows the agents to gather information about the sequence of objects itself, instead of the regular GRU, as used in EGG (Kharitonov et al., 2019), which processes all objects in parallel. While the *Temporal* agents use just the sequential GRU to process their input, the *TemporalR* agents combine both the *Base* and *Temporal* approaches, combining the outputs of a regular GRU, together with the sequentially batched GRU. 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 GRU 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 requires minimal 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 GRU-based networks. Consequently, our focus in this study remains on GRU-based agents.



(A) The *Base* GRU sender architecture.

(B) The *Base* GRU receiver architecture.

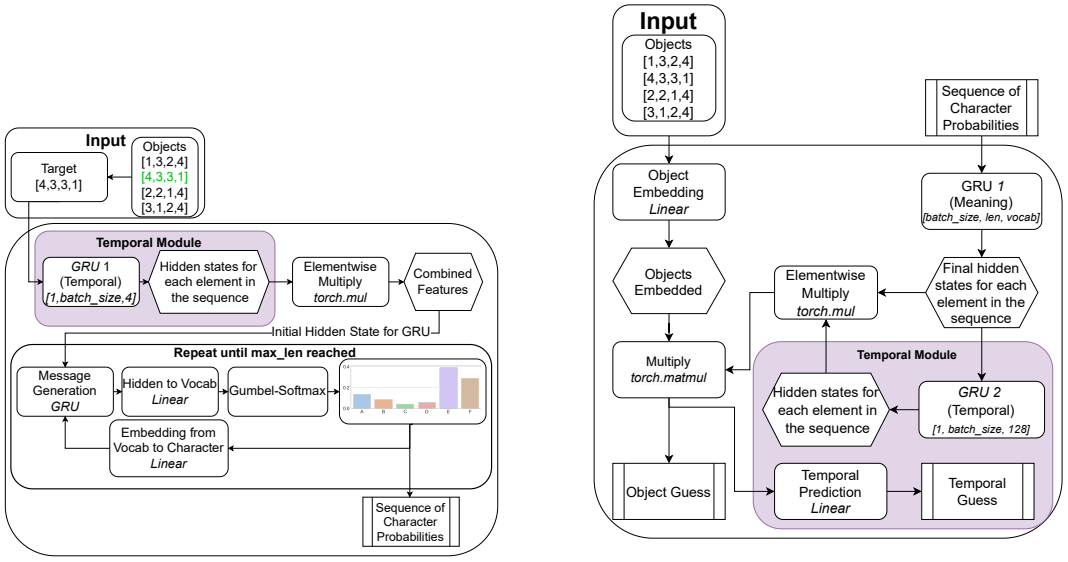
FIGURE 4.3: The *Base* GRU sender and receiver architectures.

4.3.1 *Base* Agent

In common with other approaches (Kharitonov et al., 2019; Chaabouni et al., 2019; Auersperger and Pecina, 2022), each of the *Base* sender and receiver agents are constructed around a single GRU (Cho et al., 2014), as described in Section 2.6. The sender processes the target object through the GRU to produce the initial hidden state for generating messages. These messages are then generated character by character using the Gumbel-Softmax technique (Jang et al., 2017) and passed to the receiver. The receiver's architecture combines the output from an object embedding linear layer and a message processing GRU to predict the target object in the referential game. We present the sender (Figure 4.3a) and receiver (Figure 4.3b) architectures for comparison with the *Temporal* and *TemporalR* agents.

4.3.2 *Temporal* Agent

For the *Temporal* agent, a sequential GRU module is introduced in both the sender and receiver networks (Figure 4.4). This additional GRU is batched with a sequence over the whole training input, similar to the sequential learning of language in humans (Christiansen and Kirby, 2003). Assume the regular sender GRU (Section 4.3.1) expects an input of the form $[batch\_size, seq\_len, N_{att}]$. Let $batch\_size = 128$, and $N_{att} = 6$. A batch of shape $[128, 1, 6]$ is then created, obtaining 128 objects of size 6, with sequence length 1 (Kharitonov et al., 2019). The sequential GRU (*cf.*, Figure 4.4a) instead receives



(A) The *Temporal* GRU sender architecture.

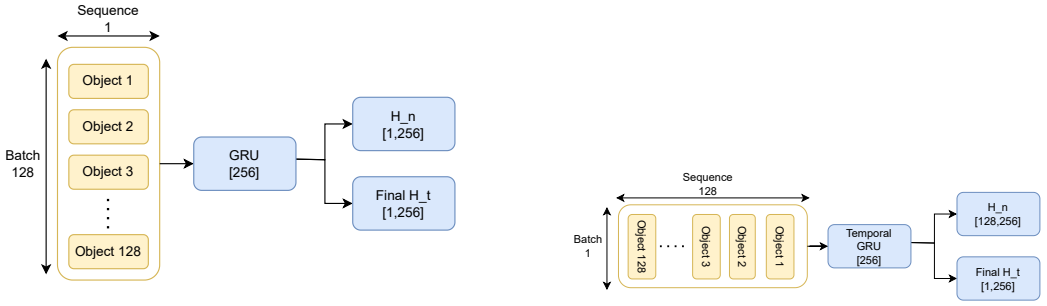
(B) The *Temporal* GRU receiver architecture.

FIGURE 4.4: The *Temporal* GRU sender and receiver architectures, with the temporal modules highlighted in purple.

a batch of shape $[1, 128, 6]$, or a sequence of 128 objects of size 6. This allows the sequential GRU to process all objects one after another to create temporal understanding. A visual representation of the two batching strategies is shown in Figure 4.5. In Figure 4.5a, the GRU takes as input all the objects at once (sequence length is 1), whereas in Figure 4.5b the GRU takes as input all objects in a sequence (sequence length is 128). This then allows the GRU's final hidden state (final h_t) in Figure 4.5b to contain the information about the temporal relationships between the objects, as compared to only the information about each object separately, as is the case in Figure 4.5a.

By including this sequential GRU, the sender and the receiver are able to develop a more temporally focused understanding. This ability to process temporal relationships is proposed to allow the agents to represent the entire object sequence within the GRU hidden state. Since it does not require reward shaping approaches or architectures specifically designed for referential games, this addition is also a scalable and general approach to allowing temporal references to develop. A different batching strategy can be applied to any environment and agent.

Additionally, the temporal prediction layer and the sequential GRU are also used in the receiver agent (Figure 4.4b). First, a hidden state is computed for each message using the regularly batched GRU. Then, the sequential GRU processes each of the regularly batched GRU's hidden states to build a temporal understanding of the sender's messages. The combined information from both GRUs and the object is also used in the temporal prediction layer, which allows the agent to signify whether an object is the



(A) Regular batching of a GRU.

(B) Temporal batching of a GRU.

FIGURE 4.5: Examples of regular and temporal batching strategies.

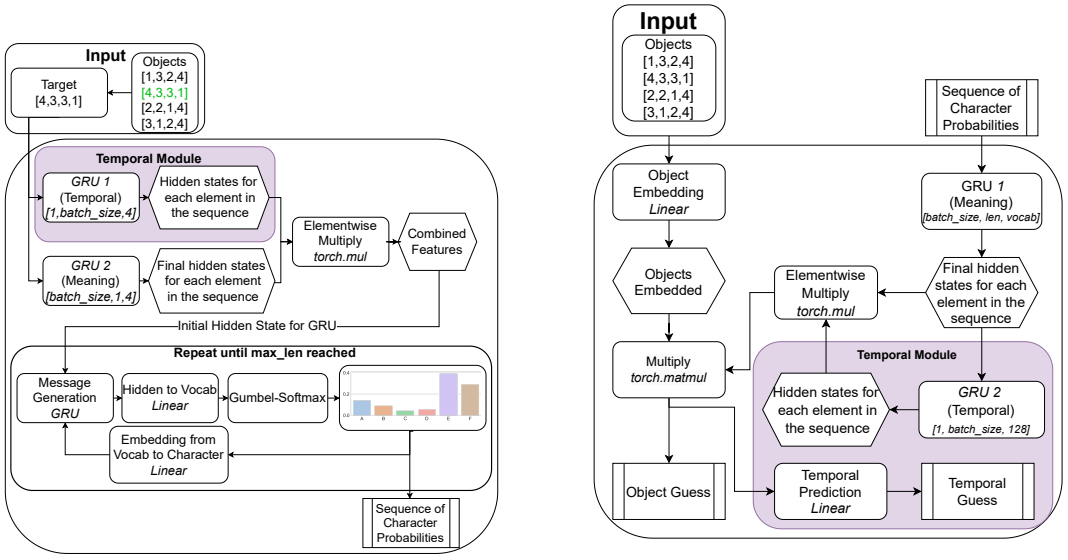
same as a previously seen object within the history length h. This is implemented as a single linear layer, which outputs the temporal label prediction.

The temporal label used in this prediction only considers the past of history length h; otherwise, it defaults to 0. For example, assume an object has been repeated in the current episode and last appeared 5 episodes ago. If the history length, h, is 8, the label assigned to this object would be 5, as 5 past episodes are still within the history, *i.e.*, $5 \leq h$. However, if h is 4, the label would be 0, as the episode lies outside the history length, *i.e.*, $5 \geq h$.

This predictive ability is combined with an additional term in the loss function, which together form a **temporal prediction loss**. The agent's loss function can be formulated as $L_t = L_{rg} + L_{tp}$. The L_{rg} component is the referential game loss between the receiver guess and the sender target label, using cross entropy. L_{tp} is the temporal prediction loss, which is implemented using cross entropy between the labels of when an object has last appeared, and the receiver's prediction of that label. Agents that include this loss perform an additional task, which corresponds to correctly identifying which two outputs are the same. The goal of this loss is to improve the likelihood of an agent developing temporal references by increasing the focus on these relationships. Analysis of how the presence of this explicit loss impacts the development of temporal references is provided in Section 4.5.

4.3.3 *TemporalR* Agent

The *TemporalR* (*cf.*, Figure 4.6) agent combines the *Base* and the *Temporal* architectures. The sequential GRU 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 GRU. The hidden states of both GRUs are combined through an elementwise multiplication and fed into the message generation GRU. The receiver agent is the same as the *Temporal* receiver, and includes the temporal prediction module.



(A) The *TemporalR* GRU sender architecture.

(B) The *TemporalR* GRU receiver architecture.

FIGURE 4.6: The *TemporalR* GRU sender and receiver architectures, with the temporal modules highlighted in purple.

4.4 Measuring Temporality and Compositionality

To be able to confirm the emergence of temporal references we need a way to analyse their presence. As current literature has not investigated temporal references, we develop a correlation metric, $M_{\ominus^n}$, to achieve this task. We also present the compositionality metrics that we use to analyse the interaction between the development of temporal references and the development of compositional languages.

4.4.1 Temporality Metric

To evaluate the development of temporal references, we propose a new metric, $M_{\ominus^n}$, which measures how often a given message has been used as the "previous" operator in prior communication. 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 history range h_v, and records the corresponding message sent to describe that object.

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

$$C_{\boldsymbol{m}\ominus^n} = \sum_{j=1}^{t} \mathbb{I}(\boldsymbol{m}_j = \boldsymbol{m} \wedge \text{objectSame}(\boldsymbol{x}_j, n)) \tag{4.3}$$

where $\mathbb{I}(\cdot)$ is the indicator function that returns 1 if the condition is true and 0 otherwise, and 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

$$\mathcal{T}_{\boldsymbol{m}} = \sum_{j=1}^{t} \mathbb{I}(\boldsymbol{m}_j = \boldsymbol{m}) \tag{4.4}$$

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 $M_{\ominus^n}(\boldsymbol{m})$:

$$M_{\ominus^n}(\boldsymbol{m}) = \frac{C_{\boldsymbol{m}}\ominus^n}{\mathcal{T}_{\boldsymbol{m}}} \times 100. \tag{4.5}$$

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". More formally, assume a target object sequence of $[\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{y}, \boldsymbol{y}, \boldsymbol{y}, \boldsymbol{x}]$. Each vector, $\boldsymbol{x}$,$\boldsymbol{y}$,$\boldsymbol{z}$, represents an object belonging to the same arbitrary V. In this example, there is only one object repeating: $\boldsymbol{y}$. We can then consider three message sequences: $[\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_2, \boldsymbol{m}_4, \boldsymbol{m}_4, \boldsymbol{m}_1]$, $[\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_4, \boldsymbol{m}_4, \boldsymbol{m}_4, \boldsymbol{m}_1]$ and $[\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3, \boldsymbol{m}_2, \boldsymbol{m}_2, \boldsymbol{m}_2, \boldsymbol{m}_1]$, with each $\boldsymbol{m}_n$ belonging to the same arbitrary ξ and calculate the metric $\ominus^1$.

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 the first example message sequence, the message $\boldsymbol{m}_4$ has been sent and so $C_{\boldsymbol{m}_4}\ominus^1 = 2$, for both of the repetitions. The total use of $\boldsymbol{m}_4$ is $\mathcal{T}_{\boldsymbol{m}_4} = 2$. Calculating the metric $M_{\ominus^1}(\boldsymbol{m}_4) = 2/2 \times 100 = 100\%$ gives 100% for the use of $\boldsymbol{m}_4$ as a $\ominus^1$ operator in this sequence: this message is used exclusively as a $\ominus^1$ operator.

In the second message sequence, $\boldsymbol{m}_4$ has also been used for the initial observation of the object. This means that $\mathcal{T}_{\boldsymbol{m}_4} = 3$, while $C_{\boldsymbol{m}_4}\ominus^1 = 2$, $M_{\ominus^1}(\boldsymbol{m}_4) = 2/3 \times 100 = 66\%$, which could mean that the message is being used as $\ominus^1$ 66% of the time.

The third message sequence illustrates the case where an agent may be using messages describing objects using their features. Following the previous examples, $\mathcal{T}_{\boldsymbol{m}_2} = 4$, with $C_{\boldsymbol{m}_2}\ominus^1 = 2$. This message would then be classed as 50% $\ominus^1$ use,
$M_{\ominus^1}(\boldsymbol{m}_2) = 2/4 \times 100 = 50\%$.

In general, if $M_{\ominus^n}(\boldsymbol{m}) < 100\%$, we cannot conclude that message $\boldsymbol{m}$ is used exclusively as a $\ominus^1$ operator. We therefore define the emergence of temporal references as the appearance of messages that are used exclusively in situations in which an agent would be expected to use a temporal reference in communication; *i.e.*, $M_{\ominus^n} = 100\%$. We expect the $M_{\ominus^n}$ metric to reach 100% for all agents that successfully incorporate temporal

references into their communication, thus providing a clear and unequivocal indication of temporal reference emergence. We employ the 100% threshold to ensure that only messages consistently used as temporal references are considered, thereby mitigating the influence of chance repetitions on our results. Chance repetitions could potentially skew the values provided by the $M_{\ominus^n}$, since if the same objects appeared multiple times in a row, our metric would classify the given message as high $M_{\ominus^n}$ value, as the message could also be seen as being used in a temporal context (see Section 4.4). However, the $M_{\ominus^n}$ metric is extremely unlikely to reach a value as high as 100% by chance alone.

4.4.2 Compositionality Metrics

Emergent languages are often analysed in terms of their compositionality scores, using the **topographic similarity** metric (Brighton and Kirby, 2006) (Section 2.4.1). 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" if the language is compositional.

Languages are also evaluated using metrics that 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 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 et al. (2020), "dogs and cats" and "cats and dogs", where both constructions are valid and convey the same information.

4.5 Temporal Referencing Experiments

Having defined our metrics for measuring temporality ($M_{\ominus^n}$) and compositionality in emergent languages, we can now analyse the language developed within the temporal referential game. Using the metrics described in Section 4.4, we can evaluate how different agent architectures and training environments influence the development of temporal references.

4.5.1 Hypotheses

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 3 (H3) All agents can develop some form of temporal references, with agents that include the temporal prediction loss more likely to do so.

Hypothesis 4 (H4) Temporal references will increase the agent's performance in environments that include temporal relationships.

Hypothesis 5 (H5) No temporal references will be detected with the $M_{\ominus^n}$ metric in environments where there are no temporal relationships.

Hypothesis 3 is investigated by using multiple agent types and applying a temporal prediction loss. Hypothesis 4 is analysed by using two environments, TRG and TRG Hard, where temporal relationships are explicitly introduced. We then compare the performance of agents, for which we have good evidence that they develop temporal references, to those that do not, to determine the impact of temporal references on task accuracy. We include all evaluation environments, including those where there are no target repetitions when evaluating the $M_{\ominus^n}$ to validate Hypothesis 5, which acts as a sanity check.

4.5.2 Agent Training

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 4.3.1);

Temporal The base learner with the sequential GRU **instead** of the regularly batched GRU (Section 4.3.2); and

TemporalR The base learner with the regularly batched GRU **and** the sequential GRU (Section 4.3.3)

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.

All agent types 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 of the agents is performed after the training has finished. Each agent pair is assessed in six different environments: Always Same, Never Same, RG, RG Hard, TRG and TRG Hard. The target objects are uniformly sampled from the object space V in all environments. The training dataset contains 20k objects, with N_{val} and

N_{att} both equal to 8. Each possible configuration was run ten times, with randomised seeds between runs for both the agents and the datasets. Appendix B.1 provides further details.

4.5.3 Significance Analysis

The underlying 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 level of significance $p < 0.05$.

4.5.4 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 that develop temporal references, and so we reject hypothesis H4. We provide a discussion of the possible reasons behind this in Section 4.6. The detailed accuracy distributions are provided in Appendix B.3.

4.5.5 Temporality Sanity Check

In line with H5, 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.

4.5.6 Temporality Analysis

Table 4.1 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 4.4 and 4.5.2), where $M_{\ominus^4} > 0\%$.[3]

[3]The value of 4 is chosen arbitrarily, to lie in the middle of the explored range of h.

TABLE 4.1: Maximum value of the $M_{\ominus^4}$ metric for each network/loss/training environment combination.

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%**

Table 4.1 indicates that the temporally focused processing of the input data makes the agents predetermined to develop temporal references. "AS" is Always Same, "RG" is the regular Referential Game environment, "TRG" is the Temporal Referential Game, and "TRG Hard" is TRG with the target differing in a single attribute with respect to the distractors.

The behaviour of sequential GRU (*Temporal* and *TemporalR*) agents, provide good evidence for the emergence of temporal references. There is strong evidence for the emergence of temporal references 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. We, therefore, conclude that temporal prediction loss is insufficient for temporal reference emergence, and that the ability to process observations temporally is the key factor. These results partially confirm hypothesis H3, 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 neither sufficient nor necessary.

Additionally, messages that are used for $\ominus^4$ have a high chance of being correct, with most averaging above 90% **correctness**. **Correctness** refers to whether the receiver agent correctly guessed the target object after receiving the message.

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

TABLE 4.2: 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%

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 temporal references, they can be used more frequently. This message specialisation, combined with a linguistic parsimony pressure (Rita 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.

The percentage of networks that develop temporal messaging is shown in Table 4.2. 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 4.2.

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, in the case of the *TemporalR* network, some runs do not converge to a temporal strategy within the training time. 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.

When increasing the number of target repetitions in a dataset, the use of temporal messages increases. As the repetition chance p increases, the percentage of messages that are used for $\ominus^n$ increases for all agent variants. 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. If a dataset contains 75% repetitions, on average, each message will be used as an accidental $\ominus^n$ 75% of the time. For example, if the language does not have temporal references and uses a given message to describe an object, this message will be repeated every time this object appears. This means that for every repetition, the message could be considered a message indicating a *previous* episode, whereas, in

reality, it is just a description of the object. 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.

4.5.7 Compositionality Analysis

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 et al., 2022b), where a score of 0.4 has been considered high in previous research (Rita et al., 2022b). We provide a visual representation of the topographic similarity scores in Appendix B.4. 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.

4.5.8 Generalisation Analysis

Analysing the development of temporal references, we observe the emergence of messages being used by the agents to describe the previous $h_v = 4$ episodes. As an example of such behaviour, in one of the runs where the agents were trained in the *TemporalR* configuration, the message $[25, 6, 9, 3, 2]$ was consistently used as a $\ominus^1$ operator. When the agents were evaluated in the Always Same environment, they used this message only when the target objects were repeating, while also being used exclusively for twelve distinct objects. For a total of 10 repetitions of each object, this message was used nine times, indicating that the only time a different message was sent was when the object appeared for the first time. For example, when the object $[4, 2, 3, 6, 5, 8, 8, 4]$ appeared for the first time, a message $[25, 6, 17, 9, 9]$ was sent, and subsequently the temporal message was used. This shows that temporal messages aid generalisation. A message that has been developed in a different training environment, in this case TRG, can be subsequently used during evaluation, even if the targets are not shared between the two environments.

4.6 Discussion

The results presented in this chapter indicate that no explicit pressures are required for temporal messages to emerge, unlike increasing linguistic parsimony where additional losses are needed (Rita et al., 2020; Kalinowska et al., 2022). 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 GRU in the *Temporal* and *TemporalR* agents. This allows temporal references to emerge in any communication setting 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 (Rita et al., 2020; Chaabouni et al., 2019).

The emergence of temporal references only through architectural changes could also point towards additional insights in terms of modelling human language evolution using EC (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.

4.6.1 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 4.5.1). While there indeed is a small increase in accuracy in the TRG Hard environment, it is not attributable to temporal references (*cf.*, Appendix B.3). 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 reason for no increase in accuracy being observed for agents that develop temporal references might be the perceptual similarities between the highly similar distractor objects. This may make the task of discerning the difference between these objects 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 be insufficient. 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 this

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 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.

4.6.2 Compositionality

We hypothesise that 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 its effect on task accuracy. Agents focus more on the temporal aspects of the task and dataset, and so 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 research in similar settings achieve higher topographic similarity scores (Chaabouni et al., 2020; Rita et al., 2022b).

4.7 Limitations

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 et al., 2022b).

4.8 Conclusion

The investigation of learning and communicating temporal relationships has remained largely unexplored, despite extensive research on various aspects of emergent languages, such as efficiency (Rita et al., 2020; Chaabouni et al., 2019), compositionality

(Auersperger and Pecina, 2022), generalization (Chaabouni et al., 2020), and population dynamics (Chaabouni et al., 2022; Rita et al., 2022a). Understanding and discussing past events is crucial for effective communication, as it conserves bandwidth by reducing redundancy and facilitates easier sharing of experiences.

This work is the first exploration of such emergent languages, including addressing the fundamental questions of when they could develop and what is required for their emergence. 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.

The results presented in this chapter demonstrate the emergence of temporal references with minimal architectural changes, highlighting the adaptability of EC systems. These findings pave the way for our subsequent analysis of spatio-temporal references in Chapter 5, where we extend the concept of temporal deixis to include spatial relationships.

Chapter 5

Spatio-temporal References

In Chapter 4, we investigated how agents may refer to repeated observations, which could also be viewed from the linguistic perspective as investigating *temporal deixis*, or how agents can refer to different moments in time (Section 2.1). Although scientists have advocated for investigations into how key concepts from natural language such as this can emerge (Rita et al., 2024), no work has demonstrated the emergence of relative references to specific locations *within* an observation. Such references could be using either *spatial* or *temporal* deixis, by, for example, referring to a part of the observation that comes after another, or which parts are next to each other.

5.1 Spatio-temporal Referencing in Emergent Communication

In linguistics, *deixis* (Section 2.1) serves as a referential pointing mechanism within language, with *temporal deixis* utilising terms such as "yesterday" or "before," while *spatial deixis* employs expressions like "here" or "next to" (Lyons, 1977).

In an emergent communication scenario, such references would be valuable in establishing shared context between agents, increasing communication efficiency by reducing the need for detailed descriptions, and adaptability, by removing the need for unique references per object. Spatio-temporal referencing streamlines communication by leveraging the shared environment as a reference point. In dynamic environments where objects might change positions, spatial references enable agents to easily track and refer to objects without having to update their descriptions. This enhances communication efficiency and improves interaction and collaboration between agents. These elements may also help the evolved language become human interpretable, allowing the development of trustworthy emergent communication (Lazaridou and Baroni, 2020; Mu and Goodman, 2021).

This work therefore explores how agents can develop communication with relative spatio-temporal references. While Rita et al. (2024) posit that the emergence of these references might require complex settings, we show that even agents trained in a modified version of the simple referential game (Lazaridou et al., 2018; Lewis, 1969) can develop them.[1]

The resulting language is analysed using a collocation measure, Normalised Pointwise Mutual Information (Section 5.4) adapted from computational linguistics. Normalised Pointwise Mutual Information allows us to measure the strength of associations between message parts and their context, allowing for message segmentation, not usually performed due to its perceived complexity (Bosc and Vincent, 2022). Using Normalised Pointwise Mutual Information, we show how the agents compose such references, providing the first evidence of a syntactic structure, usually assumed not to be present (Bosc and Vincent, 2022).

We find that the segmented language is interpretable by humans, a significant step toward developing trustworthy and transparent communication systems between agents and humans (Lazaridou and Baroni, 2020; Mu and Goodman, 2021). This human interpretability could facilitate better collaboration and trust in human-agent interactions.

5.2 Spatio-temporal Referential Game

Current emergent communication environments have not produced languages incorporating spatio-temporal references. To address this, we present a referential game (Section 2.5.1) environment where an effective language requires communication about spatio-temporal relationships.

5.2.1 Referential Game Environment

In this work, the sender's input is an observation in the form of a vector $\boldsymbol{o} = [o_1, o_2, o_3, o_4, o_5]$, where $\forall o \in \{-1, 0, 1 \dots 59\}$. The vector $\boldsymbol{o}$ is always composed of 5 integers. The observation includes a -1 in only one position, *e.g.*, $\boldsymbol{o}_3 = -1$ for $\boldsymbol{o} = [x, x, -1, x, x]$, to indicate the target integer for the receiver to identify. $\boldsymbol{o}$ represents a window into a longer sequence $\boldsymbol{s}$, which is randomly generated using the integers $\{0 \dots 59\}$ without repetitions. This sequence is visible to the receiver, but **not** to the sender. As the target's position in the sequence is unknown to the sender, it has to rely on the relative positional information present in its observation, necessitating the use of *spatio-temporal referencing*.

[1] Our code is available on GitHub at https://github.com/olipinski/TPG

Due to the window into the sequence being of length 5, it is necessary to shift the window when it approaches either extent of the sequence. The window is then shifted to the other side, maintaining the size of 5. For example, given a short sequence $\boldsymbol{s} = [7, 5, 2, 12, 10, 4, 3, 15, 16, 13, 14, 6, 9, 8, 11, 1]$, if the selected target is 1, since there are no integers to the right of 1 the vector $\boldsymbol{o}$ would be $\boldsymbol{o} = [6, 9, 8, 11, -1]$ where it is shifted to the left as it approaches this rightmost extent of the sequence.

Due to the necessity of maintaining the window size, some observations provide additional positional information to the sender agent. Given the same example sequence $\boldsymbol{s}$, we can categorise all observations into 5 types. The first two types are *begin* and *begin+1*, where the target integer is either at, or one after, the beginning of the sequence, *i.e.*, $\boldsymbol{o} = [-1, 5, 2, 12, 10]$ or $\boldsymbol{o} = [7, -1, 2, 12, 10]$. The *end* and *end-1*, where the target integer is either at, or one before, the end of the sequence, *i.e.*, $\boldsymbol{o} = [6, 9, 8, 11, -1]$ or $\boldsymbol{o} = [6, 9, 8, -1, 1]$. The most common case is the *middle* observation, where the target integer is anywhere in the sequence, excluding the first, second, second to last, and last positions, *e.g.*, $\boldsymbol{o} = [12, 10, -1, 3, 15]$. Given a window of length 5, only 4 specific target integer positions per sequence can result in the other observations (*begin*, *begin+1*, *end-1*, and *end*). All other target integer positions within the sequence fall into the *middle* category, as they do not occupy the first, second, second to last, or last positions. Consequently, the majority of the target integer positions result in a *middle* type observation, as for the window size 5, only 4 integers per sequence can be the **non**-*middle* observations, *i.e.*, the first two and the last two integers. This means that as the sequence length grows, the probability of a **non**-*middle* observation decreases.

The sender's output is a message defined as a vector $\boldsymbol{m} = [m_1, m_2, m_3]$, where $m \in \{1 \dots 26\}$. 26 is chosen to allow for a high degree of expressivity, with the agents being able to use over 17k different messages, while also matching the size of the Latin alphabet. Since such a vocabulary size is enough to convey any information in natural languages like English, we consider that this should also apply to the agents. The vector $\boldsymbol{m}$ is always composed of 3 integers.

The receiver's input is an observation consisting of three vectors: the sender's message $\boldsymbol{m}$, the sequence $\boldsymbol{s}$, and the set of distractor integers together with the target integer $\boldsymbol{td}$. The distractor integers are randomly generated, without repetitions, given the same range of integers as the original sequence $\boldsymbol{s}$, *i.e.*, $\{0 \dots 59\}$, excluding the target object itself. Given an environment with 3 distractors, $\boldsymbol{td}$ could be $[d_1, t, d_2, d_3]$, where t is the target object and d_1, d_2, d_3 are distractor objects. The position of the target object in $\boldsymbol{td}$ is randomised.

For example, given the sequence $\boldsymbol{s} = [7, 5, 2, 12, 10, 4, 3, 15, 16, 13, 14, 6, 9, 8, 11, 1]$, and the sender's observation $\boldsymbol{o} = [4, 3, -1, 16, 13]$, the vector $\boldsymbol{td}$ could be $\boldsymbol{td} = [7, 15, 11, 9]$, with 15 being the target that the receiver needs to identify. The sender could produce a message $\boldsymbol{m} = [3, 1, 1]$, which would mean that the target integer is one after the integer

3. This message would then be passed to the receiver, together with $\boldsymbol{s}$ and $\boldsymbol{td}$. The receiver would then have to correctly understand the message $\boldsymbol{m}$ (*i.e.*, that the target is one after 3) and find the integer 3 together with the following integer in the sequence $\boldsymbol{s}$. Having identified the target, 15, given the message $\boldsymbol{m}$ and the sequence $\boldsymbol{s}$, it would output the correct position of this target in the $\boldsymbol{td}$ vector, *i.e.*, 2, since $\boldsymbol{td}_2 = 15$.

5.2.2 Spatio-temporal Reference Formalisation

To provide a foundation for understanding how spatio-temporal references might extend beyond our current setting, we formalise what we refer to by spatio-temporal references.

Let O represent an abstract observation that an agent perceives from its environment, $O \in \mathbb{R}^m$, where m represents the dimensions of the observation. These dimensions can represent any structure of the observation space. For example, in a 2D grid, m could be $m = j \times k$ or all the grid cells; in an RGB image m could be $j \times k \times 3$ (pixels $\times$ colour channels); and in a video sequence m could be $j \times k \times t \times 3$ (pixels $\times$ timesteps $\times$ colour channels). The m dimensions are able to represent the spatial, temporal, or other positions.

Let O_p and O_t be the coordinates of some elements in O, represented by an m-tuple of natural numbers $(x_1, x_2 ... x_m)$ and $(y_1, y_2 ... y_m)$, respectively. Each coordinate x_i or y_i represents the position along the i-th dimension of the observation space. O_p represents the reference point and O_t represents a target point. These coordinates could refer to some objects in the observation O.

Then, the relative distance function $d(O_p, O_t)$ returns an m-tuple of integers $(z_1, z_2 ... z_m)$, such that $z_i = x_i - y_i$. This relative distance function allows for unambiguous identification of the target object O_t, given that the position of O_p is known.

We define the spatio-temporally referent expression as a mapping of the value of $d(O_p, O_t)$, the reference point O_p, and their context O, to a specific linguistic or symbolic phrase that describes the relationship between O_p and O_t. This mapping can be represented as:

$$(O, d(O_p, O_t), O_p) \rightarrow \text{Phrase}(O, d(O_p, O_t), O_p)$$

where the resulting expression $\text{Phrase}(O, d(O_p, O_t), O_p)$ is a description of the reference point O_p and its relative distance to the target point O_t, given the context O.

To give intuition to this formalisation consider an example, shown in Figure 5.1. An agent observes a 2D grid $O \in \mathbb{R}^{4 \times 4 \times 2}$ representing a 4×4 grid, where each element contains two values: one for the existence of a box and one for the box colour. The agent's reference point is $O_p = (2, 2)$, which could represent the agent itself. In this grid,

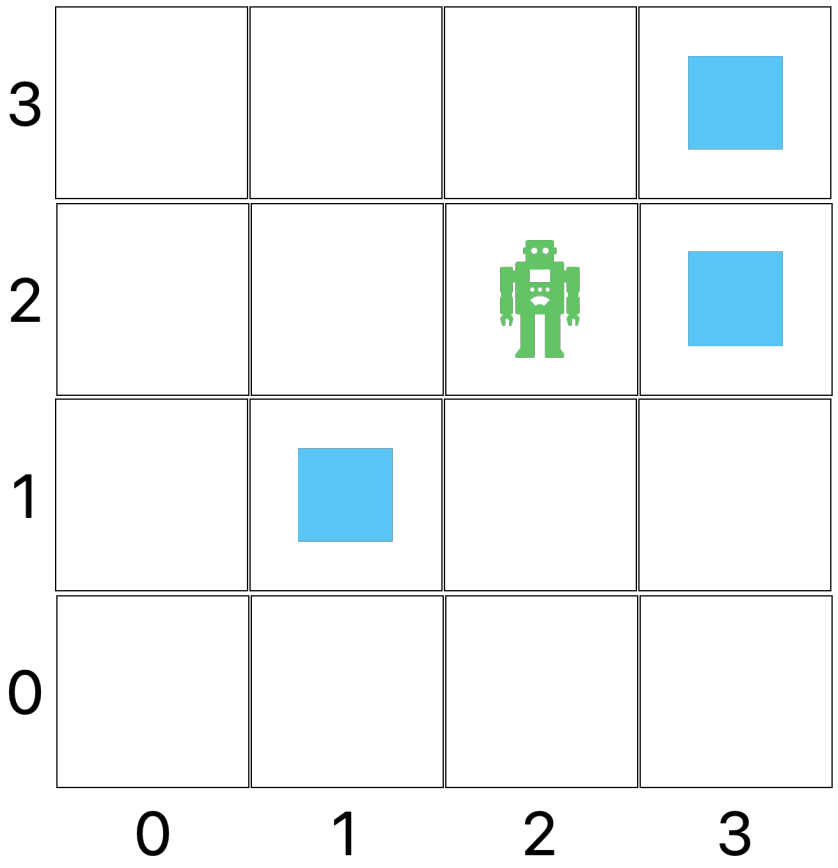


FIGURE 5.1: Spatio-temporal referencing example.

there are three identical blue boxes at positions $(3,2)$, $(3,3)$, and $(1,1)$. To refer to one of the blue boxes, the agent calculates the relative distances. For the blue box at $(3,3)$, $d(O_p, O_t) = (2-3,\ 2-3) = (-1,\ -1)$. The mapping to a phrase could then yield "The blue box diagonally ahead to my right". The agent has to specify the position more precisely, as there are two blue boxes to the right. The context O was necessary to distinguish between the blue boxes on the right using their relative positions. However, for the blue box at $(1,1)$, $d(O_p, O_t) = (2-1,\ 2-1) = (1,\ 1)$. Here the phrase can simply be "The blue box to my left", as there are no other blue boxes on the left. No additional spatial information is needed due to its uniqueness in the context O.

This example demonstrates how the necessity of context O influences whether all parts of the relative distances $d(O_p, O_t)$ are required in the referent expression to unambiguously identify the target object O_t.

We note that the interpretation of these referent expressions can encompass both spatial and temporal dimensions simultaneously. For instance, in a one-dimensional sequence, a reference such as 'before' or 'after' could be interpreted spatially (as in 'to the left' or 'to the right'), temporally (as in 'earlier' or 'later'), or both. This ambiguity is inherent in the agent's perspective, as they may not explicitly distinguish between spatial and temporal relationships. The mapping function $\text{Phrase}(O, d(O_p, O_t), O_p)$ can therefore

produce expressions that bridge both spatial and temporal references, with the precise interpretation depending on the context O and the observer's frame of reference. This flexibility in interpretation becomes particularly relevant in sequential environments, where spatial and temporal relationships naturally coincide.

The version of spatio-temporal referencing present in our environment is a specific case of the general spatio-temporal reference formalisation, where the observation O is represented as a one-dimensional tensor, and the target object is always indicated by the value -1 within the observation tensor O. The sender's task is to describe the relative position of the target O_t within this sequence, using a message that effectively communicates the spatio-temporal relationship between an agent-chosen O_p and the target O_t. However, the general framework of the spatio-temporal references remains the same, allowing for expanding the insights gained in this work to other environments.

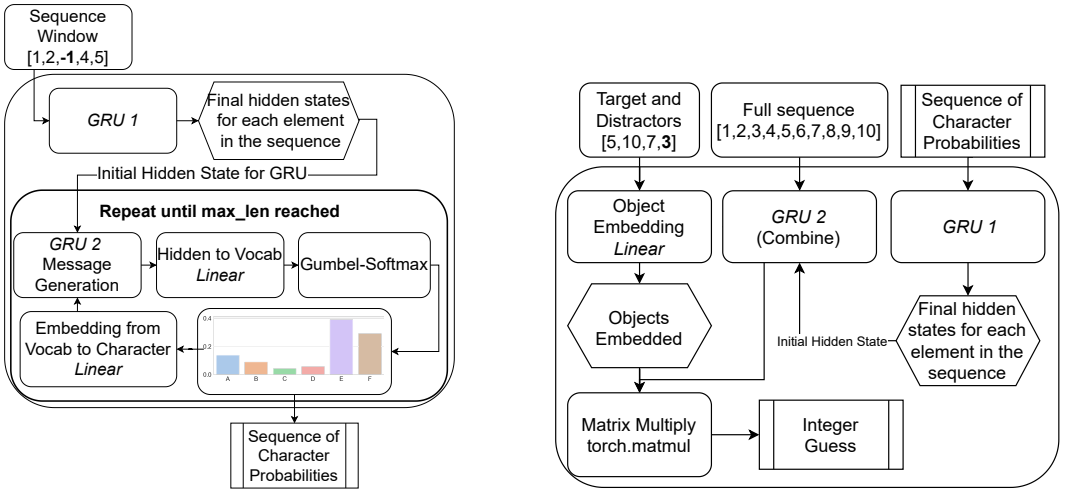
5.3 Agent Architecture

The agent architecture follows that of the most commonly used EGG agents (Kharitonov et al., 2019) (Section 2.6). This architecture is used to maintain consistency with the common approaches in emergent communication research (Kharitonov et al., 2019; Chaabouni et al., 2019, 2020; Ueda and Washio, 2021; Lipinski et al., 2023), increasing the generalization of the results presented in this work.

The sender agent, shown in Figure 5.2a, receives a single input, the vector $\boldsymbol{o}$ composed of scalar values[2], which is passed through the first GRU of the sender. The resulting hidden state is used as the initial hidden state for the message generation GRU (Cho et al., 2014). The message generation GRU is used to produce the message, character by character, using the Gumbel-Softmax reparametrization trick (Jang et al., 2017; Mordatch and Abbeel, 2018; Kharitonov et al., 2019) (Section 2.7). The sequence of character probabilities generated from the sender is used to output the message $\boldsymbol{m}$.

Message $\boldsymbol{m}$ is input to the receiver agent, shown in Figure 5.2b, together with the full sequence $\boldsymbol{s}$ and the target and distractors $\boldsymbol{td}$. The message is processed by the first receiver GRU, which produces a hidden state used as the initial hidden state for the GRU processing the sequence $\boldsymbol{s}$. This is the only change from the standard EGG architecture (Kharitonov et al., 2019). This additional GRU allows the receiver agent to process the additional input sequence $\boldsymbol{s}$, using the information contained within the message $\boldsymbol{m}$. The goal of this GRU is to use the information provided by the sender to correctly identify which integer from the sequence $\boldsymbol{s}$ is the target integer. The final hidden state from the additional GRU is multiplied with an embedding of the targets

[2]One-hot encoding of the observation vectors leads to agents memorising the dataset.



(A) The sender architecture. (B) The receiver architecture.

FIGURE 5.2: The sender and receiver architectures. Adapted from (Lipinski et al., 2023).

and distractors, to output the receiver's prediction. This prediction is in the form of the index of the target within $\boldsymbol{td}$.

Following the commonly used approach (Kharitonov et al., 2019), agent optimisation is performed using the Gumbel-Softmax reparametrization (Jang et al., 2017; Mordatch and Abbeel, 2018), allowing for direct gradient flow through the discrete channel. The agents' loss is computed by applying the cross entropy loss, using the receiver target prediction and the true target label. The resulting gradients are passed to the Adam optimiser and backpropagated through the network. Detailed training hyperparameters are provided in Appendix C.1.

5.4 Message Interpretability and Analysis using NPMI

To analyse spatio-temporal references in emergent language, we need a way to identify their presence. In discrete emergent languages, interpretation is typically done by either using dataset labels in natural language (Dessì et al., 2021), or by qualitative analysis of specific messages (Havrylov and Titov, 2017)(Section 2.4.2). However, both of these techniques require message-meaning pairs, and so neither would be able to identify the presence of spatio-temporal references. As the meaning behind each message and what the agents focus on is developed during their interactions, associated labels will not necessarily be easily available, or may require finding the meaning in the first place, creating a circular dependency. One approach that could overcome this problem is emergent language segmentation using Harris' Articulation Scheme, recently employed by Ueda et al. (2023). Ueda et al. (2023) compute the conditional entropy of each character in the emergent language, segmenting the messages where the conditional entropy increases. However, even after language segmentation, there is no easy way to

interpret the segments, as no method has been proposed to map them to specific meanings.

We present an approach to both segment the emergent language and map the segments to their meanings. We use a collocation measure called Normalised Pointwise Mutual Information (NPMI) (Bouma, 2009), often used in computational linguistics (Yamaki et al., 2023; Lim and Lauw, 2024; Thielmann et al., 2024). It is used to determine which messages are used for which observations and to analyse how the messages are composed, including whether they are trivially compositional (Korbak et al., 2020; Steinert-Threlkeld, 2020; Perkins, 2021). By applying a collocation measure to different parts of each message as well as the whole message, we can address the problems of both segmentation and interpretation of the message segments. This approach allows any part of the message to carry a different meaning. For example, if an emergent message contains segments that frequently appear in contexts involving specific integers, NPMI can help identify these segments and their meanings based on their statistical association with those integers.

NPMI is a normalised version of the Pointwise Mutual Information (PMI) (Church and Hanks, 1989), which is a measure of association between two events. PMI is widely used in computational linguistics, to measure the association between words (Paperno and Baroni, 2016; Han et al., 2013). Normalising the PMI measure results in its codomain being defined between -1 and 1, with -1 indicating a purely negative association (*i.e.*, events **never** occurring together), 0 indicating no association (*i.e.*, events being **independent**), and 1 indicating a purely positive association (*i.e.*, events **always** occurring together). Normalised PMI is used for convenience when defining a threshold at which we consider a message or n-gram to carry a specific meaning, as the threshold can be between 0 and 1, instead of unbounded numbers in the case of PMI. [3]

To determine which parts of each message are used for a given meaning, two algorithms are proposed.

1. PMI_{nc} The algorithm to measure non-compositional monolithic messages, most often used for target positional information (*e.g.*, *begin+1* (Section 5.2)); and
2. PMI_{c} the algorithm to measure compositional messages and their n-grams, used to refer to different integers in different positions.

A visual representation of the different types of messages that the algorithms can identify is provided in Figure 5.3. The PMI_{nc} algorithm can identify any non-compositional messages, while the PMI_{c} algorithm identifies both position variant and invariant compositional messages. The positional variance of the emergent

[3] Our implementation of NPMI is not numerically stable due to probability approximation, sometimes exceeding the [-1,1] co-domain.

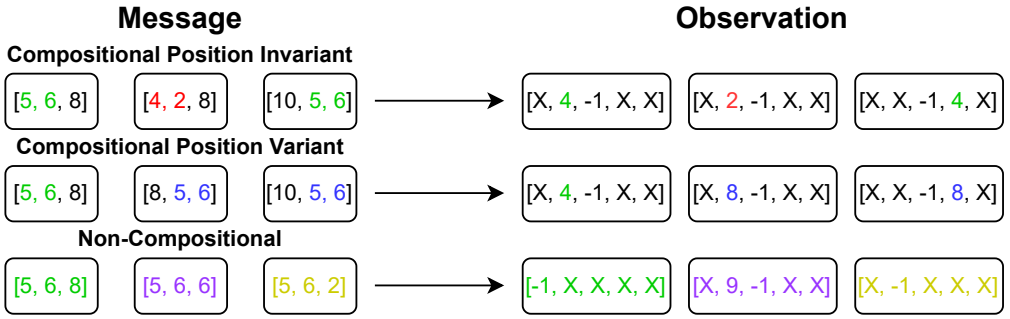


FIGURE 5.3: Examples of the different types of message compositionality that are possible to identify using the PMI algorithms.

language means that the position of an n-gram in the message also carries a part of its meaning. In this work, n-grams refer to a contiguous sequence of n integers from the sender's message. Consequently, in one message there are 3 unigrams (m_1, m_2, m_3), two bigrams ([m_1, m_2], [m_2, m_3]), and one trigram (*i.e.*, the whole message [m_1, m_2, m_3]).

Figure 5.3 shows that in the position invariant case, the bigram $[5, 6]$ always carries the meaning of 4. While in the position variant case, the bigram $[5, 6]$ in position 1 of the message means 4, but $[5, 6]$ in position 2 of the message means 8. This can also be interpreted as the position of the bigram containing additional information, meaning a single "word" could be represented as a tuple of the bigram and its position in the message, as both contribute to its underlying information. Non-compositional messages are monolithic, *i.e.*, the whole message carries the entire meaning. For example, message $[5, 6, 8]$ means the target is in the first position, while $[5, 6, 6]$ means the target is one to the right of 9, even though the two messages share the bigram $[5, 6]$.

The PMI$_{nc}$ algorithm The PMI$_{nc}$ algorithm calculates the NPMI per message by first building a dictionary of all counts of each message being sent, together with an observation that may provide positional information (*e.g.*, *begin+1*) or refer to an integer in a given position (*e.g.*, 1 left of the target). The counts of that message and the counts of the observation, including the integer position, are also collected. For example, consider the observation $\boldsymbol{o} = [4, -1, 15, 16, 13]$. For the corresponding message $\boldsymbol{m}$, the counts for each integer in each position relative to the target would increase by 1 (*i.e.*, $left1[4] += 1$, $right1[15] += 1$ *etc.*). The count for the message signifying *begin+1* would also be increased. Given these counts, the algorithm then estimates the probabilities of all respective events (messages, positional observations, and integers in given positions) and calculates the NPMI measure.

We provide a condensed pseudocode for the PMI$_{nc}$ algorithm in Algorithm 1. The n-grams in the pseudocode would be whole messages, *i.e.*, trigrams. This base pseudocode would then be duplicated, interpreting the context as either an observation that may provide positional information (*e.g.*, *begin+1*) or an integer. A detailed

Algorithm 1: PMI Algorithm Base

Gather `ngram_counts`, `context_counts`, `joint_counts`, `n_grams`;
for *each n-gram g in position p and context c* **do**
  $P(g,p) = \texttt{ngram\_counts}[g] \cdot \frac{1}{\texttt{total } n\texttt{-grams}}$;
  $P(c) = \texttt{context\_counts}[c] \cdot \frac{1}{\texttt{total contexts}}$;
  $P(g,p;c) = \texttt{joint\_counts}[(g,c)] \cdot \frac{1}{\texttt{total } n\texttt{-grams}}$;
  $\text{NPMI}(g,p;c) = \log_2 \frac{P(g,p,c)}{P(g)P(c)} \cdot \frac{1}{-\log_2 P(g,p,c)}$;
end
return NPMI;

commented pseudocode for the PMI_{nc} algorithm is available in Algorithm 2 in Appendix C.3.

The PMI_c algorithm The PMI_c algorithm first creates a dictionary of all possible n-grams, given the message space (m) and maximum message length (3). The list of all possible n-grams is pruned to contain only the n-grams present in the agents' language, avoiding unnecessary computation in the later parts of the algorithm. Given the pruned list of n-grams, the algorithm checks the context in which the n-grams have been used. The occurrence of each n-gram is counted, together with the n-gram position in the messages and the context in which it has been sent, or the integers in the observation. The n-gram position in the message is considered to account for the possible position variance of the compositional messages.

Consider the previous example, with $\boldsymbol{o} = [4, -1, 15, 16, 13]$ and a message $\boldsymbol{m} = [11, 13, 5]$. For all n-grams ([11], [13], [5], [11, 13], *etc.*) of the message, all integers are counted, irrespective of their positions (*i.e.*, $counts[4] += 1$, $counts[15] += 1$, *etc.*).

Given these counts, the PMI_c algorithm estimates the NPMI measure for all n-grams and all integers in the observations. These probabilities are estimated from the dataset using the count of their respective occurrences divided by the number of all observations/messages.

Once the NPMI measure is obtained for the n-gram-integer pairs, the algorithm calculates the NPMI measure for n-grams and referent positions or the positions of the integer in the observation the message refers to. For example, given an observation $\boldsymbol{o} = [4, -1, 15, 16, 13]$, if the message contains an n-gram which has been identified as referring to the integer 15, the rest of the message (*i.e.*, the unigram or bigram, depending on the length of the integer n-gram) is counted as a possible reference to that position, in this case, to position $right1$, or 1 to the right of the target. This procedure follows for all messages, building a count for each time an n-gram was used together

with a possible n-gram for an integer. These counts are used to calculate the NPMI measure for n-gram and position pairs.

The PMI_c algorithm also accounts for the possible position invariance of the n-grams, *i.e.*, where in the message the n-gram appears. This is achieved by calculating the respective probabilities *regardless* of the position of the n-gram in the message, by summing the individual counts for each n-gram position.

The condensed pseudocode in Algorithm 1 can also be used for the PMI_c algorithm. In that case, only the unigrams and bigrams would be evaluated. The base pseudocode would also be duplicated, once for the integer in a given position, and second for the referent position. Each would be used as the context in which to evaluate the NPMI for each n-gram. A detailed commented pseudocode for PMI_c algorithm is available in Algorithm 3 in Appendix C.3.

Both algorithms use two hyperparameters: a confidence threshold t_c and top_n t_n. The confidence threshold t_c refers to the value of the NPMI measure at which a message or n-gram can be considered to refer to the given part of the observation unambiguously. To account for polysemy (where one symbol can have multiple meanings), the agents can use a single n-gram to refer to multiple integers. This is given by the second hyperparameter, top_n t_n, which sets the degree of the polysemy, or the number of integers to be considered for a given n-gram.

5.5 Spatio-temporal Referencing Experiments

Having defined the necessary measures to analyse the presence of spatio-temporal references, we proceed to analyse their emergence. This section examines whether agents can successfully develop a communication protocol for describing spatio-temporal relationships, and whether we can interpret their emergent referential system using the NPMI-based analysis method described in Section 5.4.

Agent pairs are trained over 16 different seeds to verify the results' significance. All agent pairs achieve above 98% accuracy in the spatio-temporal referential game, suggesting that the agents are able to communicate about spatio-temporal relationships in their observations. The analysis provided in this section is based on the messages collected from the test dataset after the training has finished.

The two hyperparameters, t_c and t_n (Section 5.4), governing the NPMI measure have been determined through a grid search to maximise the understanding of the emergent language, by maximising the translation accuracy. The results in this section are obtained using the best-performing values for each of the hyperparameters. We provide the values for the grid search in Appendix C.1.

5.5.1 Emergence of non-compositional spatio-temporal references

Using the PMI_{nc} algorithm, we detect the emergence of messages tailored to convey the spatio-temporal information contained in the observations. As mentioned in Section 5.2, sender observations which require shifting convey additional information about the position of the target within the sequence. In over 90% of agent pairs, these observations, *i.e.*, *begin*, *begin+1*, *end-1* and, *end*, are assigned unique messages.

In 20% of runs which develop these specialised messages, the same repeating character is used to convey the message. The characters used for these observations are *reserved* only for these types of observations. For example, in one of the runs the agents use character 11 to signify the beginning of the sequence, with the character 11 being used only in two contexts: as the messages $[11, 11, 11]$ to signify *begin*, or as a message $[0, 11, 11]$ to signify *begin+1*. In other cases, characters are fully reserved for specific messages. *e.g.*, 22 is used only for *end*, in the message $[22, 22, 22]$.

The emergence of non-compositional references used for other observations is also detected using the PMI_{nc} algorithm. Such messages refer to a specific integer in a specific position of the sender observation, *e.g.*, $o_5 = 10$. While we allow for polysemy by considering up to the top 15 most correlated messages for each observation ($t_n = [1, 2, 3, 5, 10, 15]$), we observe the highest translation accuracy with $t_n = 1$, indicating that the non-compositional messages do not carry any additional meanings.

5.5.2 Emergence of compositional spatio-temporal references

Using the PMI_c algorithm, we also detect the emergence of *compositional spatio-temporal references* for 25% of agent pairs. Such messages are composed of two parts, a spatio-temporal reference and an integer reference. The spatio-temporal reference specifies where or when a given integer can be found in the observation, in relation to the masked target integer -1. The integer reference specifies which integer the positional reference is referring to. For example, one pair of agents has assigned the unigram 7 to mean that the *target* integer is 2 to the right of, or after, the *given* integer, and the bigram $[0, 2]$ to mean the integer 18. Together, a message can be composed $[7, 0, 2]$, which means that the target integer for the receiver to identify is 2 to the right of, or after, the integer 18, *i.e.*, $\boldsymbol{o} = [18, X, -1, X, X]$. This allows the sender to identify the target integer exactly, given the sequence $\boldsymbol{s}$. The PMI_c algorithm follows the same approach as the PMI_{nc} algorithm for evaluating polysemy when identifying compositional references, using up to the top 15 most correlated n-grams ($t_n = [1, 2, 3, 5, 10, 15]$) for both the spatio-temporal and integer component of the message.

TABLE 5.1: Average emergence and vocabulary coverage of all message types.

Message Type	Avg. % Emergence	Avg. % of Messages
Non-Compositional ST	99.3% (100%-93.75%)	1% (3%-0%)
Non-Compositional ST Reserved	18.75%	1% (3%-0%)
Non-Compositional Integer	45.1% (100%-0%)	10% (15%-0%)
Compositional Integer	100%	34% (99.7%-0%)
Compositional ST	25% (27%-0%)	56% (100%-0%)

In Table 5.1, we summarise the emergence of each type of message across all runs, together with the percentage of the vocabulary that they represent. "ST" is short for Spatio-Temporal, and "ST Reserved" refers to the messages that use specific characters exclusively dedicated to conveying specific spatio-temporal information, such as for "begin" and "end" (Section 5.5.1). The entries in the table are composed of average percentages, across all t_n and t_c choices. In the parentheses, we show the maximum and minimum values across all t_n and t_c choices. The average % of emergence represents the absolute % of runs in which a certain message type or message feature emerges. The average % of messages for a given type or feature is computed only across the runs where that type or feature emerged, excluding runs where it did not appear. For example, if a feature emerges in 8 out of 16 runs, its emergence is 50%, and the average % of messages which contain that feature is calculated using only those 8 runs.

5.5.3 Generalisation

To generalise the results presented in this work, we also run additional tests, varying the vocabulary size, training sequence length, evaluation sequence length, and the hidden size of the agents, as outlined in Appendix C.1. We observe no performance decline with either increasing or decreasing the vocabulary size or the training sequence length, given that the agents have enough capacity within their network to still learn the longer sequence lengths. We observe a decline in task accuracy at sequence lengths of 100, when the agents have a hidden size of 64. However, increasing the hidden size to 128 brings the training and validation accuracy back to over 90%.

When agents are evaluated on sequence lengths that are different from the ones they were trained on, we observe a small performance decline for small differences in sequence lengths. We present the average accuracies for the base case (Sequence shortened by 0), as well as the average difference in accuracy as compared to the baseline for different sequence lengths in Table 5.2. We observe a significant difference if the agents are evaluated on sequences that are over 50% shorter than the ones they were trained on. We hypothesise that this is due to the agents missing certain integers that they used more often than others, therefore reducing their accuracy. However, even in the worst case, the accuracy remains above 70%.

TABLE 5.2: Evaluation accuracy differences for shorter sequence lengths compared to the training sequence lengths.

Training length	Sequence shortened by				
	0	-5	-10	-20	-40
20	98.68%	98.15% (-0.53%)	91.18% (-7.50%)	N/A	N/A
40	95.59%	95.64% (0.05%)	94.84% (-0.75%)	90.04% (-5.55%)	N/A
60	92.98%	93.28% (0.30%)	92.68% (-0.30%)	90.51% (-2.47%)	77.18% (-15.8%)
100	86.23%	86.57% (0.34%)	86.20% (-0.03%)	84.97% (-1.26%)	81.03% (-5.2%)

5.5.4 Evaluating interpretation validity and accuracy

To ensure the validity of our message analysis, we present two hypotheses which, if supported, would indicate that the mappings generated by the NPMI measure are correct.

Hypothesis 6 (H6) If the correlations exist and do not require non-trivial compositionality (Perkins, 2021), and are not highly context-dependent (Nikolaus, 2023), then the evaluation accuracy should be significantly higher than chance, or above 20%, when using the identified mappings.

Hypothesis 7 (H7) If the spatio-temporal components of compositional messages are correctly identified and carry the intended meaning, then their inclusion should result in an increase in accuracy.

Given the messages identified by the NPMI method, we test **H6** and **H7** by using a dictionary of all messages successfully identified, given the values of the NPMI hyperparameters t_n and t_c. A dataset is generated to contain only targets that can be described with the messages present in the dictionary.

For the non-compositional messages, the dataset is generated by selecting a message from the dictionary at random, and creating an observation that can be described with that message. Given a non-compositional message that corresponds to the target being on the right of, or after, the integer 15, an observation $\boldsymbol{o} = [1, 15, -1, 5, 36]$ would be created. Analogously, for non-compositional spatio-temporal messages such as *begin* an observation $\boldsymbol{o} = [-1, 15, 8, 5, 36]$ would be created.

For the compositional messages, we create the observations by randomly selecting a spatio-temporal component and an integer component from the dictionary. For example, given the unigram 7 meaning that X is 2 to the left of, or before, the target, we could select the bigram $[8, 14]$ corresponding to the integer 30. The observation created could then be $\boldsymbol{o} = [30, 8, -1, 36, 5]$. The dataset creation process for the compositional messages also checks if the observations can be described given the two n-grams in their required positions within the message.

TABLE 5.3: Accuracy improvements using the NPMI-based dictionary.

Dict Type	t_n	t_c	Average Accuracy	Maximum Accuracy
Non-Compositional ST	1	0.9	0.90 $\pm$0.03	**0.94**
Non-Compositional Integer	1	0.5	0.36 $\pm$0.004	0.37
Compositional-NP	1	0.5	0.22 $\pm$0.02	0.28
Compositional-P	1 [4]	0.5	0.30 $\pm$0.21	0.78

To test **H7**, a dataset is created using **only** the integers that can be described by the dictionaries, randomly selecting integer components from the dictionary, and creating the respective observations. This process also accounts for the required positions of the message components so that a message describing the observation can always be created. For example, if the unigram 9 described the integer 11, and the bigram $[5, 1]$ described the integer 6, a corresponding observation could be $\boldsymbol{o} = [11, 6, -1, 8, 9]$. The positions of the integers in the observations are chosen at random. By generating both compositional datasets using a stochastic process, we do not assume a specific syntax. Rather, the syntax can only be identified by looking at messages which were understood by the receiver.

These datasets, together with their respective dictionaries, are then used to query the receiver agent, testing if the messages are identified correctly. We run this test for all of our trained agents, with the dictionaries that were identified for each agent pair. We provide the details of this evaluation in Table 5.3, where $\pm$ denotes the 1-sigma standard deviation. Non-Compositional ST (Spatio-Temporal) refers to messages such as *begin* or *end*, Non-Compositional Integer refers to the non-compositional monolithic messages describing both the position and the integer, Compositional-NP refers to messages only containing the identified integer components, and the Compositional-P which refers to messages containing both the identified integer and spatio-temporal components.

Using just the non-compositional spatio-temporal messages, we observe a significant increase in the performance of the agents, compared to random chance accuracy of **20%**. This provides strong evidence for the support of Hypothesis **H6**, showing that at least some messages do not require complex functions to be composed, or contextual information to be interpreted. As the accuracy for these messages reaches over 90% on average, we argue that the NPMI method has captured almost all the information transmitted using the non-compositional spatio-temporal messages.

As mentioned in **H7**, we examine the impact of the spatio-temporal components and whether they carry the information the NPMI method has identified. We, therefore, separate the compositional analysis into two parts: Compositional-NP, where the spatio-temporal components are replaced with 0, and Compositional-P, which includes

[4]The value of t_n for the referent position n-grams is set to 0.3 while the value of t_n for the integer n-grams is set to 1.

the identified spatio-temporal components. In the Compositional-NP case, the agents achieve a close to random accuracy, with a maximum recorded of 28%. In contrast, Compositional-P agents achieve above random accuracy, with some agent pairs reaching over 75% accuracy. This provides strong evidence for the support of Hypothesis **H7**, allowing us to conclude that the NPMI method successfully identifies spatio-temporal information contained in messages, together with the integer information.

5.6 Discussion

Having demonstrated the validity of Hypotheses **H6** and **H7**, we confirm the emergence and correct identification of spatio-temporal references. To provide human interpretability of the emergent language, we use the NPMI method to create a dictionary providing an understanding of the analysed messages. We present an excerpt from an example dictionary in Table 5.4. With human interpretability, we can gain a deeper understanding of the principles underlying the agents' communication protocol.

We posit that the emergence of compositional spatio-temporal references points to a first emergence of a simple syntactic structure in an emergent language. Both of the n-grams in our example from Section 5.5.2, also shown in Table 5.4, are assigned specific positions in the message by the agents. The unigram 7 must always be in the first position of the message, while the bigram $[0, 2]$ must always be in the second position. The emergence of this structure shows that even though referential games have been considered obsolete in recent research (Chaabouni et al., 2022; Rita et al., 2024), a careful design of the environment may yet elicit more of the fundamental properties of natural language.

We hypothesise that the emergence of non-compositional spatio-temporal references tailored to specific observations, such as *begin+1*, is due to observation sparsity. Compositionality would bring no benefit since the observations they describe are usually rare, representing 1–2% of the dataset and are monolithic, *i.e.*, *begin*, *begin+1*, *end-1*, and *end*. We therefore argue that the emergence of non-compositional references in these cases is **advantageous**, since these messages are easily compressible. Since these messages are monolithic, they could be compressed to a single token/character in simple encoding schemes. In contrast, compositional messages require at least two tokens/characters, one for each integer/spatio-temporal component. With a linguistic parsimony pressure (Rita et al., 2020; Chaabouni et al., 2019) applied, these messages could be more efficient at transmitting the information contained within these observations than compositional ones.

TABLE 5.4: Example dictionary of the agents' messages and their meanings

Message	Type	Meaning
$[11, 11, 11]$	Non-Compositional Spatio-temporal	*begin*
$[0, 11, 11]$	Non-Compositional Spatio-temporal	*begin+1*
$[10, 10, 10]$	Non-Compositional Spatio-temporal	*end-1*
$[18, 18, 18]$	Non-Compositional Spatio-temporal	*end*
$[12, 16, 14]$	Non-Compositional Integer	15 is 1 left of, or before, target
$[15, m_2, m_3]$	Compositional Spatio-temporal	? is 2 left of, or before, target
$[7, m_2, m_3]$	Compositional Spatio-temporal	? is 2 right of, or after, target
$[m_1, 0, 17]$	Compositional Integer	Integer 1
$[m_1, 0, 2]$	Compositional Integer	Integer 18
$[m_1, 8, 14]$	Compositional Integer	Integer 30

5.7 Limitations

The accuracy for the Non-Compositional Integer, and Compositional-P messages averages about 33%. While still above random, showing that some meaning is captured in non-compositional messages, it points to there being more to be understood about these messages. We hypothesise this may be due to the higher degree of message pragmatism, or context dependence (Nikolaus, 2023). Our method of message generation, using randomly selected parts, may not be able to capture the complexity of the messages. For example, the context in which they are used might be crucial for some n-grams, requiring the use of a specific n-gram instead of another when referring to certain integers, or when specific integers are present in the observation. Just like in English, certain verbs are only used with certain nouns, such as "pilot a plane" vs "pilot a car". While the word "pilot" in the broad sense refers to operating a vehicle, it is not used with cars specifically. This may also be the case for emergent languages. For compositional messages, an additional issue may be that some messages are non-trivially compositional, using functions other than simple concatenation to convey compositional meaning (Perkins, 2021), making them impossible to analyse with the NPMI measure. However, these issues may be addressed by scaling the emergent communication experiments as the languages become more general with the increased complexity of their environment (Chaabouni et al., 2022).

5.8 Conclusion

Recent work in the field of emergent communication has advocated for better alignment of emergent languages with natural language (Rita et al., 2024; Boldt and Mortensen, 2024b), such as through the investigation of deixis (Rita et al., 2024). Aligned to this approach, we provide a first reported emergent language containing *spatio-temporal references* (Lyons, 1977), together with a method to interpret the agents' messages in

natural language. We show that agents can learn to communicate about spatio-temporal relationships with over 90% accuracy. We identify both compositional and non-compositional spatio-temporal referencing, showing that the agents use a mixture of both. We hypothesise why the agents choose non-compositional representations of observation types which are sparse in the dataset, arguing that this behaviour can be used to increase communicative efficiency. We show that, using the NPMI language analysis method, we can create a human interpretable dictionary, of the agents' own language. We confirm that our method of language interpretation is accurate, achieving over 94% accuracy for certain dictionaries.

Chapter 6

Discussion

Throughout this thesis, we have focused on the influence of temporal dynamics on the emergent language. Having answered the research questions presented in Chapter 1, this chapter provides a critical analysis of the findings from Chapters 3 to 5, integrating insights from the impact of communication time, the emergence of temporal references, and the development of spatio-temporal languages. We discuss the broader implications of these findings and outline potential future research directions. Our findings are well-placed to deliver new and interesting results to both the EC community and the broader ML community. Moreover, with the improvements in EC, through our research into more advanced properties, we will be moving closer to improving autonomous and ad-hoc communication, and its efficiency.

6.1 Compositionality

In all studies presented in this thesis, agents develop languages with various levels of compositional structure. In Chapter 3, we show that agents develop a degenerate language, with no compositional structure, while in Chapter 4 and Chapter 5 agents develop compositional languages. These results point to two aspects of measuring compositionality — the effect of language efficiency, discussed in Section 6.2 and the issues with compositionality measurement, discussed in this section.

As mentioned in Section 2.4.1, measuring the compositionality of emergent languages is challenging. Using just the topographic similarity metric leads to a perceived low compositionality score of the languages developed by agents in Chapter 4. However, as noted by Nikolaus (2023), topographic similarity is limited because it is unable to measure more context-dependent messages. Topographic similarity would also fail to identify compositional messages, where there is variation present in message

composition (Conklin and Smith, 2022), such as using messages that are similar to indicate the same concept, for example using distance-1 messages in Chapter 3.

Having observed the limitations of the compositionality metrics in Chapter 4 we instead opt to analyse the compositionality of messages in Chapter 5 using our more tailored NPMI measure. While our NPMI measure does use knowledge about the content of the dataset to perform segmentation of the messages, it could be adapted for more general purposes, while preserving its ability to analyse the spatio-temporal references in agent messages. Given the NPMI measure relies on simple collocation between n-grams and possible meanings, it is an extensible framework upon which future work could build.

We consider the analysis of compositionality to be a pathway to more general insights about emergent languages. While analysing the compositionality alone is a poor way of measuring agent performance or language generalisation abilities (Chaabouni et al., 2022; Rita et al., 2024), analysing the compositionality of messages may lead to more interpretable emergent protocols. Thus, we consider the development of tools for analysing the emergent languages in terms of their compositionality, syntax, grammar, and other linguistic properties to be an important aspect of future work in the field of EC. We propose one such avenue in this thesis, showing that compositionality can be analysed using a technique based on the NPMI measure. Future research can further draw on inspiration from tools developed for linguistics, similar to grammar induction tools (Ueda et al., 2022; van der Wal et al., 2020), or HAS (Ueda et al., 2023) (Section 2.1). Such endeavours could then be tested on large datasets of emergent communication corpora, made possible by the recent work of Boldt and Mortensen (2024a).

Moreover, investigating compositionality in emergent languages could help bridge the gap between artificial and natural languages (Rita et al., 2024). By developing an understanding of how compositionality emerges and evolves in artificial systems, future research might be able to draw parallels to human language development and cognitive processes. This understanding could also inform the design of more human-like communication systems in AI.

6.2 Efficiency

Improving communication efficiency is one of the goals of emergent communication. Throughout this thesis, we focus on how the emergent protocols, developed by the agents, could be used to contribute to this goal.

In Chapter 3, we show that agents can develop a highly efficient communication protocol, requiring only a single message to be learned by all villagers to achieve a high win rate, even though efficiency was not explicitly encouraged. This finding

underscores the potential for spontaneous emergence of optimal communication strategies in agent-based systems.

In Chapter 4 and Chapter 5, we further illustrate that with minimal architectural modifications and specific environmental pressures, agents can learn to utilize temporal and spatio-temporal deixis effectively. These modifications enable agents to convey the spatio-temporal relationships in their observations in flexible and informative messages, showing possible increases in communication bandwidth and generalisability.

As mentioned in Chapter 4, such references, and more broadly efficient languages, can significantly increase the communication bandwidth. By compressing such references, we can transmit the emergent language more efficiently. The ability to develop and refine efficient languages has important implications for real-world applications. For instance, in autonomous systems deployed in dynamic environments, the ability to communicate efficiently can lead to significant improvements in operational performance and cost-effectiveness. Future research could explore the scalability of these findings across more complex environments and larger agent populations to validate this broader applicability.

Efficiency in communication is not merely about reducing the length of messages, but also about enhancing the quality of the transmitted information, increasing its relevance and richness. The ability of agents to distil spatio-temporal relationships into concise messages means that they can rely on shared context, allowing for more precise communication. By ensuring that every bit of communicated information is relevant and contextually appropriate, we can enhance the overall performance and reliability of multiagent systems.

6.3 Scalability

All the environments used in this thesis were relatively simple. We use a simple social deduction game Werewolf (Chapter 3), and the simple referential game (Chapters 4 and 5), to evaluate the different temporal environmental and inductive pressures and their effect on the communication protocol. We consider this a strength of our approach, as it allows us to isolate any extraneous factors and focus on what is needed to achieve the desired properties of the language.

However, this also poses a question of scalability. Our networks, while based on the state-of-the-art approaches in EC literature (Section 2.6), are quite simple. One could wonder whether such approaches would scale to wholly different network architectures such as transformers (Vaswani et al., 2017). However, this may be an unnecessary distraction. Recent literature shows that RNN-based architectures, such as the xLSTMs (Beck et al., 2024), may be just as good, if not better than, transformers. Additionally,

transformer models, and LLMs, have been shown to not necessarily be reliable when it comes to temporal features, such as time series forecasting (Tan et al., 2024). Instead, a combined approach, such as our *TemporalR* architecture in Chapter 4, may provide an avenue to improve their capabilities.

The recurrent nature of RNN-based architectures, such as the ones presented in this thesis, could pose an obstacle to scaling them. While transformers can be trained in parallel, RNNs cannot. This has helped propel transformers to become the go-to architecture for most LLMs. While it is an obstacle in widespread adoption of RNN architectures, it may not be a fatal flaw. A combined approach could be the way forward, where transformers handle System 1 reasoning (fast, intuitive, and parallel processing of information akin to human pattern recognition and heuristic decision-making) and slower RNNs manage System 2 reasoning (sequential, deliberate processing requiring step-by-step computation, similar to human logical analysis and structured problem-solving). This architectural division would mirror human cognitive processes, where System 1 enables rapid, unconscious pattern recognition and intuitive responses, while System 2 supports methodical, conscious reasoning through complex problems that require sequential processing (Kahneman, 2011). Alternatively, a small RNN could be used alongside the transformer to feed temporal information to its latent space, thus decreasing the training time required for that component. Additionally, once proven successful, parallelizable RNN architectures such as the mLSTM (Beck et al., 2024) could replace both transformers and other RNNs.

6.4 Temporality

Our work is novel in assessing the influence of time in emergent language. We have made significant progress in understanding how temporal aspects shape communication protocols. By analysing temporal and spatio-temporal deixis, as well as the impact of communication time, we have provided a foundation for future research in this area.

Exploiting the temporal and spatio-temporal deixis brings possible efficiency gains, as they are more compressible, especially when compared to purely descriptive compositional languages (Chapter 4). Showing that the amount of time to communicate increases the training and accuracy of agent populations could also improve performance in other areas of emergent communication, such as population-based approaches (Rita et al., 2022a; Michel et al., 2022).

There is much more to explore in the temporal aspects of emergent communication. Future research could investigate hierarchical temporal structures, interactions across different time scales, and more complex temporal relationships. Understanding these

aspects will be crucial as emergent communication systems are deployed in increasingly dynamic and temporally complex environments.

6.5 Broader Impact

The approaches and insights developed in this thesis also carry the potential of applications to fields outside of EC. The interpretation methods presented, including the NPMI measure or the $M_{\ominus^n}$ measure, could also be used to interpret latent representations in other neural models. As the basic setting of EC could be viewed as an auto-encoder model, with the sender being the encoder (encoding observations into messages) and the receiver being the decoder (decoding messages into objects), the approaches presented in Chapter 5 and Chapter 4 could also be employed to understand such models. For example, using the NPMI approach from Chapter 5 we could attempt to find the latent space representations of different features in the dataset. Additionally, since similarities exist in transformer models encoding their observations into latent spaces, these could also be approached with the measures developed in this thesis. Models that develop complex latent representations, such as JEPA-based models (Garrido et al., 2024), may also benefit from these approaches. Understanding of such large models will be increasingly important (Maslej et al., 2024), especially as research shows that model size is a major factor in its generalisation abilities (Hong et al., 2024).

Understanding of how AI models operate is becoming increasingly important, not just to the researchers in AI, but also to the general public (Maslej et al., 2024). As AI is being used in more industrial settings with the promise of improving worker productivity (Maslej et al., 2024), understanding of how such models make their decisions will be crucial. Additionally, AI is now being used to generate data for itself, such as through Reinforcement Learning from AI Feedback (RLAIF) (Bai et al., 2022), an extension of Reinforcement Learning from Human Feedback (RLHF). RLAIF leverages the scalability and efficiency of AI to provide feedback faster than humans can. Therefore, understanding the underlying models is essential to avoid harmful self-reinforcing loops.

With the rise of the number of incidents and increased adoption of AI among businesses, including in medical applications (Maslej et al., 2024), trustworthiness of AI systems will be of paramount importance. Approaches such as latent space analyses could be combined with, or assist the efforts of, mechanistic interpretability, which has also been trying to understand the linguistic properties in intermediate activations of language models (Rai et al., 2024).

Emergent protocols, with the rise of emergent language corpora datasets (Boldt and Mortensen, 2024a), could also be used for better training of language models (Yao et al., 2022; Cope and McBurney, 2024), especially with the drive for self-improvements, such

as through RLAIF (Bai et al., 2022). Temporality, important in LLMs, could also help their performance (Hou et al., 2024). The use of techniques not reliant on human input paves the way for faster improvements, especially where human data is scarce (Maslej et al., 2024).

Chapter 7

Conclusion

Previous research in Emergent Communication has primarily focused on simpler properties of EC. In this thesis, we explore the more advanced aspect of temporal dynamics in emergent communication, an understudied area until now. We investigate three main aspects of temporality in emergent languages: the amount of interaction time in dialogue, the emergence of inter-observational temporal references, and spatio-temporal references within a single observation.

To study the influence of the time given to agents to interact, we modified the game of Werewolf, examining how the duration of communication impacts language evolution and agent strategies. Our findings reveal that increased communication time enhances agent performance, highlighting the critical role of temporal factors in agent communication. Additionally, we demonstrate that without constraints, agents tend to develop highly effective but linguistically degenerate language.

We also explored the emergence of temporal references in agent communication. We answer the key questions of how and when such references develop, showing that temporal references naturally arise when agents can process past information. This occurs with only minimal modifications to the standard EC agent architecture, underscoring the generalisability of this approach.

We analysed the agents' ability to communicate local spatio-temporal relationships within their environment. Our findings confirm that standard agent architectures can develop these references, provided the environment contains such relationships, thereby expanding the scope of what agents can learn and communicate about. We also show that, using measures developed for natural language analysis, we can develop a better understanding of how agents communicate such relations.

Throughout this thesis, we focused on multiple aspects of the influence of time on emergent language. Through experimental analyses, we have shown how such aspects can emerge, how they influence the agents' behaviour and how agents create meaning

within their messages. We consider that the techniques developed in this thesis can advance emergent languages created by autonomous agents a step closer to human languages. Through further research into the ways to analyse emergent languages and incentivise more and more advanced and human-like language properties, we are optimistic that emergent languages can become just as rich and complex as human languages, lending them their efficiency and generalisability.

Appendix A

Werewolf

A.1 Training Details

We use the Ray (Moritz et al., 2017) and RLLib (Liang et al., 2018) libraries to train the agents, using the APPO algorithm. APPO is an asynchronous sampling variant of the Proximal Policy Optimization (Schulman et al., 2017), provided through RLLib (Liang et al., 2018). The training runs last for an average of 3M episodes. With the experimental setup as described, a total of 180 runs were performed. An average run of 3M episodes took approximately 20 hours on a single NVIDIA RTX8000 GPU. We present the overview of our compute resources used in this study in Table A.2.

We present the training hyperparameters in Appendix A.1. All our training parameters, except for n_r and t_v which were the addition of our study, follow that of (Brandizzi et al., 2021) to introduce as little variation as possible.

TABLE A.1: Training and Grid Search Parameters

Parameter	Value
Episodes	3M
Optimizer	Adam
Learning Rate α	0.0003
GAE Parameter λ	0.95
Discount Factor γ	0.998
Batch Size	500
Rollout Fragment Length	100
Batch Mode	Complete Episodes
Voting Threshold t_v	[0, 0.2, 0.4, 0.6, 1]
Number of Rounds n_r	[1, 3, 12, 36]

TABLE A.2: Compute Resources

Resource	Value
CPU Cores (Intel(R) Xeon(R) Silver 4216 × 2)	10
GPUs (NVIDIA Quadro RTX8000)	1
Wall Time	20hrs
Memory	40GB

TABLE A.3: Linear regression analysis.

Relationship	p-Value	R^2
Number of Rounds vs Win Rate	< 0.001	0.264
Voting Threshold vs Win Rate	0.600	0.002
Number of Rounds vs Convergence Episode	< 0.001	0.189
Voting Threshold vs Convergence Episode	0.686	0.001

A.2 Statistical Significance Analysis

The normality of the distribution of both dependent variables, the mean of villager win rates and the number of episodes that it takes villagers to converge, are analysed. The normality analysis is performed using the Shapiro-Wilk normality test from the SciPy package (Virtanen et al., 2020), obtaining p-values of less than 0.0001, indicating that the data is not normally distributed.

The correlation and significance of the correlation are then analysed using the Spearman rank correlation metric from the pandas package (McKinney, 2010; The pandas development team, 2020). The results of these correlation tests are presented in Figure A.1. The Spearman correlation coefficient, together with the Spearman p-value, are shown for each variable pair. The significance, or p-value, can be discerned by the number of * next to the corresponding correlation coefficient, where **no** * signifies $p > 0.05$; * is $p < 0.05$; and ** is $p < 0.01$.

The strength for the number of rounds affecting either the villager win rate or convergence episode is high. However, the relationship between the threshold and win rate or convergence is much weaker.

The results are also analysed with a simple linear regression model, from the SciPy package (Virtanen et al., 2020), shown in Table A.3. The linear regression tests predict that win rate and convergence episode are affected by the number of rounds, whereas the voting threshold does not appear to have an effect on either.

These analyses suggest that the number of rounds does have a statistically significant effect on both the win rate and convergence speed. However, no statistically significant correlation is found between the voting threshold and either of our dependent variables.

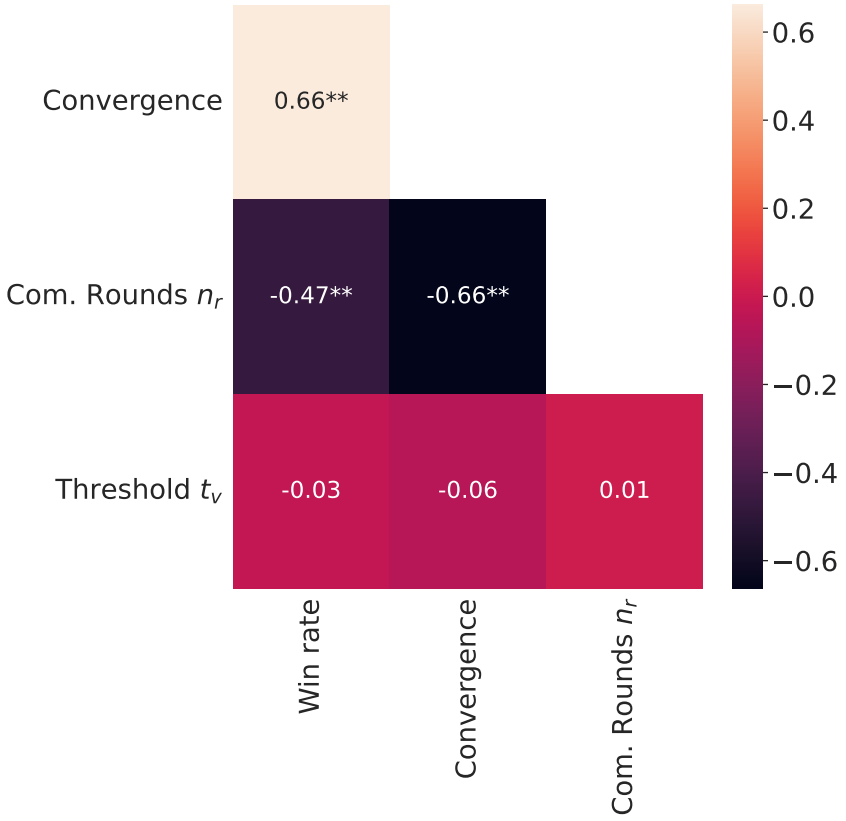


FIGURE A.1: Spearman correlation strength and its significance.

Appendix B

Temporal Referential Games

B.1 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 B.1. We ran the 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 B.2.

TABLE B.1: Training and 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]
History Length 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 B.2: Compute Resources

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

B.2 Datasets Details

In Figure B.1, we analyse our datasets, using the parameters as specified in Appendix B.1, 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 B.1, for the Classic and Hard referential games.

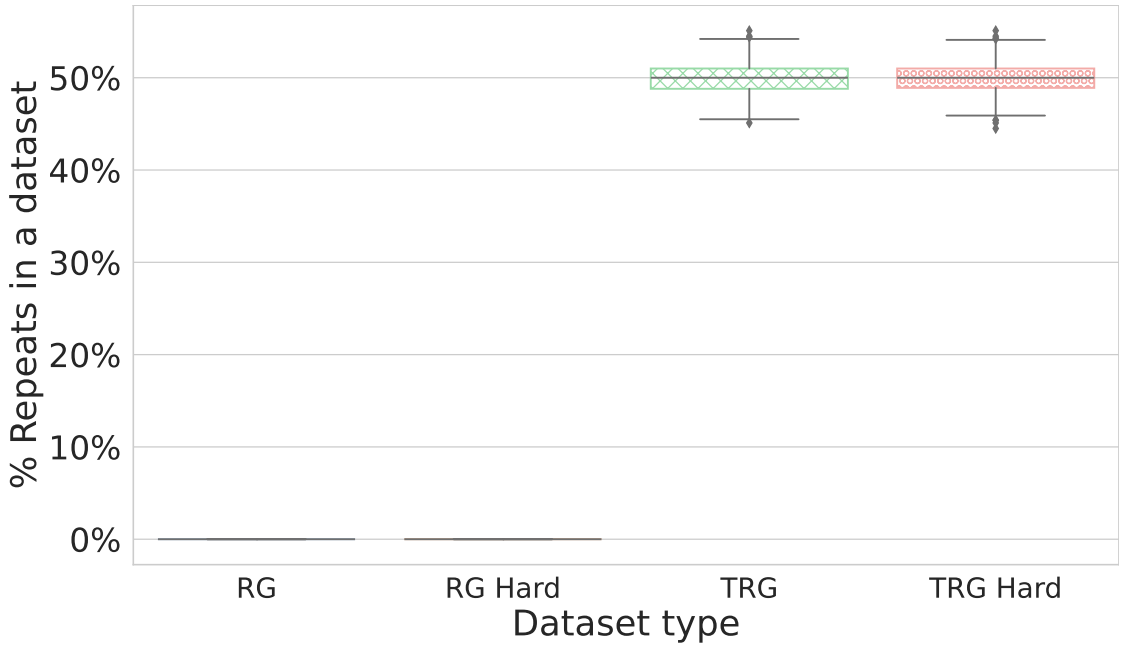


FIGURE B.1: Number of target repetitions per dataset. Regular referential games datasets very rarely encounter target repetitions. This data is an average over 1000 seeds per environment.

B.2.1 Test Environments

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

We provide example inputs and outputs for both environments in Table B.3 and Table B.4.

For the Always Same environment, in the case of the agent using temporal references, we may also see other messages instead of the message 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 B.3: Example Inputs and Outputs for Always Same.

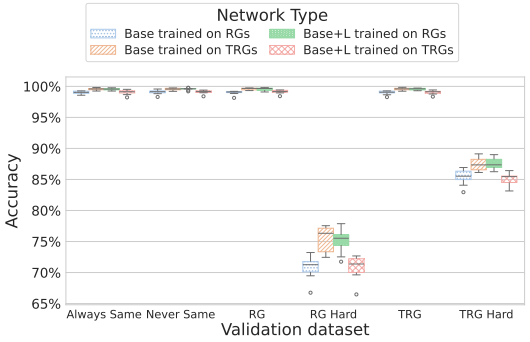
Example Type	Example Values
Input	$[x, x, x, y, y, y, z, z, z]$
Temporal Referencing	$[m_1, m_4, m_4, m_2, m_4, m_4, m_3, m_4, m_4]$
No Temporal Referencing	$[m_1, m_1, m_1, m_2, m_2, m_2, m_3, m_3, m_3]$

TABLE B.4: Example Inputs and Outputs for Never Same.

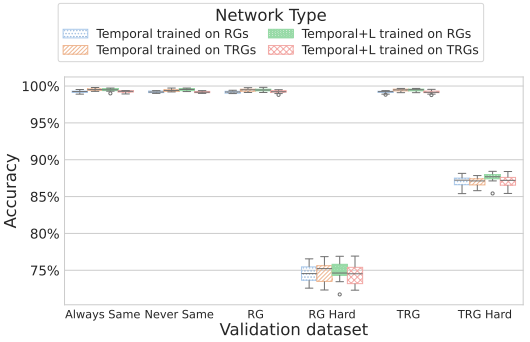
Example Type	Example Values
Input	$[x, y, z, a, b, c, d, e]$
Temporal Referencing	$[m_1, m_1, m_1, m_2, m_2, m_2, m_3, m_3, m_3]$
No Temporal Referencing	$[m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8]$

B.3 Accuracy Distributions

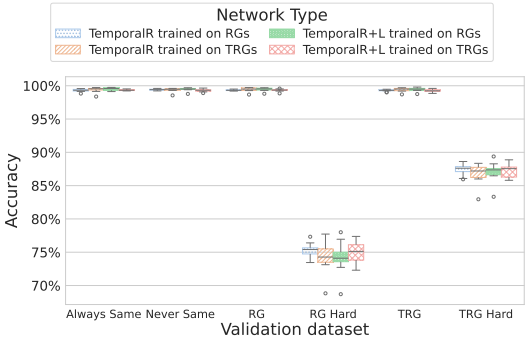
The accuracy distributions for all agent types across all evaluation environments are shown in Figures B.2a to B.2c, where "+L" refers to agents trained with the temporal loss. All agents converge to very similar levels of accuracy.



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



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

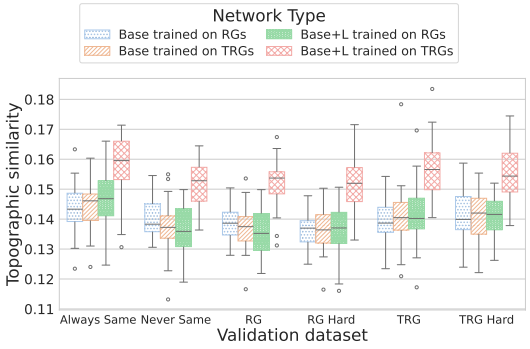


(C) The evaluation accuracy for the *TemporalR* agents across all environments.

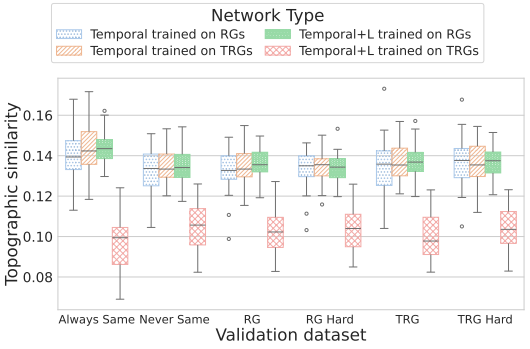
FIGURE B.2: Accuracies for each network variant on all evaluation environments.

B.4 Topographic Similarity Distributions

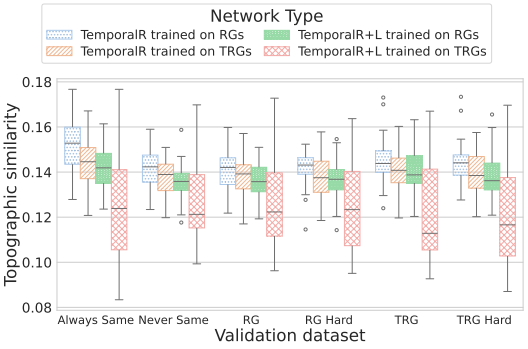
The topographic similarity distributions for all agent types across all evaluation environments are shown in Figures B.3a to B.3c. All agents converge to very similar values of topographic similarity.



(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.

FIGURE B.3: Topographic similarity scores for each network variant on all evaluation environments.

Appendix C

Temporal Progression Games

C.1 Training Details

The computational resources needed to reproduce this work are shown in Table C.1, with the hyperparameters in Table C.2 and Table C.3. The Table C.1 shows resources required for all training and evaluation. The processors used were a mixture of Intel Xeon Silver 4216s and AMD EPYC 7502s. The GPUs used for the training were a mixture of NVIDIA Quadro RTX 8000s, NVIDIA Tesla V100s, and NVIDIA A100s, hosted on the IRIDIS cluster.

TABLE C.1: Compute resources

Resource	Value (1 Run)	Value (Training Total)	Value (Evaluation & Analysis)
Nodes	1	8	1
CPU	16 cores	128 cores	64 cores
GPU	1	8	1
Memory	50 GB	400 GB	120 GB
Storage	1 GB	32 GB	32 GB
Wall time	2 hours	240 hours	24 hours

C.2 Dataset Details

To train and evaluate the agents, we use datasets consisting of 200,000 samples for training, 200,000 for validation, and 20,000 for testing. Each dataset is generated independently, with sequences created randomly. Given the sequence length of 60 and the fact that no integers are repeated, the number of possible permutations is $60! \approx 8 \times 10^{81}$, which vastly exceeds the number of samples we generate. We further ensure that there is no overlap between datasets by empirically checking the overlap rates across 1,000 randomly generated datasets, confirming an overlap rate of 0%.

TABLE C.2: Training and Grid Search Parameters

Parameter	Value
Epochs	1000
Optimizer	Adam
Learning Rate α	0.001
Gumbel-Softmax Temperature	[1.0]
Training Dataset Size	200k
Test Dataset Size	20k
No. Distractors	4
No. Points	[20,40,60,100]
Message Length	3
Vocabulary Size	[13,26,52]
Sender Hidden Size	[64,128]
Receiver Hidden Size	[64,128]

TABLE C.3: PMI Grid Search Parameters

Parameter	Values
t_c	[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]
t_n	[1, 2, 3, 5, 10, 15]

C.3 NPMI Algorithm Descriptions

For our pseudocode we will be using the Python assignments convention, *i.e.*, $=$ and $\leftarrow$ are equivalent, and x+=1 is equivalent to $x \leftarrow x + 1$. The algorithms presented are for $top\_n = 1$. To improve the computational efficiency. the probability of the integer appearing is statically defined as $\frac{1}{60}$ for $top\_n = 1$, or in Equation (C.1) for $top\_n > 1$. In the case of $top\_n > 1$ we use the probability for the integer as per Equation (C.1), to account for the polysemy, *i.e.*, the probability for any of top_n integers occurring in the observation. The lower part of the binomial is 4, as there are 4 integers that can be sampled from the 60 possible integers, instead of 5, as we exclude the target integer.

$$p(integers) = \frac{\binom{60}{4} - \binom{60-top\_n}{4}}{\binom{60}{4}} \tag{C.1}$$

Additionally, in the PMI_c algorithm, we specify a probability to equal to 0.98 in Line 74 and Line 77. This is a simplification of the calculation for clarity of the pseudocode. This probability is instead obtained using the count of a given type of observation, divided by the number of total observations. This calculation is performed for each type of observation, *i.e.*, *begin*, *begin+1*, *end*, *end-1* and *middle*. The probability of the *middle* observation is very close to 1, being on average 0.98, while the other probabilities are on average 0.005. Since the *middle* observation is most common, we included its value in the pseudocode.

Algorithm 2: The PMI_{nc} algorithm

Data: O_M ; # All observations together with sent messages
Data: $L = len(O\_M)$; # Total number of observations with sent messages
Data: $S = [begin, begin + 1, end - 1, end]$; # List of spatio-temporal observations
Result: $pmi_{nc}[m][NPMI]$

pmi_{nc} = dict;
for $o, m \in O\_M$ **do**
  $pmi_{nc}[m][count]$ += 1 ; # Message occurrences
  for $pos \in S$ **do**
    if $o == pos$ **then**
      $pmi_{nc}[pos][count]$ += 1 ; # Spatio-temporal observations count
      $pmi_{nc}[m][pos]$ += 1 ; # Message sent with spatio-temporal observation
    end
  end
  for $integer \in o$ **do**
    $pmi_{nc}[m][integer\_pos][integer]$ += 1 ; # Message sent with integer in given position
  end
end
for $pos \in S$ **do**
  $posit_{total} = pmi_{nc}[pos][count]$; # Count of spatio-temporal observations
  $p(pos) = \frac{posit_{total}}{L}$; # Estimate observation probability
  for $m \in pmi_{nc}[m]$ **do**
    $m_{total} = pmi_{nc}[m][count]$; # Total count of message
    $ms_{total} = pmi_{nc}[m][pos]$; # Total count of message with spatio-temporal obs
    $p(m) = \frac{m_{total}}{L}$; # Estimate message probability
    $p(m, pos) = \frac{ms_{total}}{L}$; # Estimate joint probability
    $h(m, pos) = -\log_2(p(m, pos))$;
    $pmi(m, pos) = \log_2(\frac{p(m,pos)}{p(m)p(pos)})$;
    $npmi(m, pos) = \frac{pmi(m,pos)}{h(m,pos)}$;
    $pmi_{nc}[m][NPMI] = npmi(m, pos)$;
  end
end
for $pos \in pmi_{nc}[m]$ **do**
  for $integer \in pmi_{nc}[m][pos]$ **do**
    $p(pos) = \frac{1}{60}$; # Estimated observation probability for 60 integers
    $m_{total} = pmi_{nc}[m][count]$; # Total count of message
    $ms_{total} = pmi_{nc}[m][pos][integer]$; # Total count of message with integer in given position
    $p(m) = \frac{m_{total}}{L}$; # Estimate message probability
    $p(m, pos) = \frac{ms_{total}}{L}$; # Estimate joint probability
    $h(m, pos) = -\log_2(p(m, pos))$;
    $pmi(m, pos) = \log_2(\frac{p(m,pos)}{p(m)p(pos)})$;
    $npmi(m, pos) = \frac{pmi(m,pos)}{h(m,pos)}$;
    $pmi_{nc}[m][pos][integer][NPMI] = npmi(m, pos)$;
  end
end

Algorithm 3: The PMI_c algorithm

Input: t_c ; # Confidence value
Data: O_M ; # All observations together with sent messages
Data: $L = len(O\_M)$; # Total number of observations with sent messages
Data: $ngrams$; # List of all message ngrams present in O_M
Result: $pmi_c[m][NPMI]$

pmi_c = dict;
; # First we identify ngrams corresponding to integers.
for $ngram \in ngrams$ **do**
  for $o, m \in O\_M$ **do**
    if $ngram \in m$ **then**
      $pmi_c[ngram][count]$ += 1 ; # Total ngram occurrences
      $pmi_c[ngram][ngram\_pos][count]$ += 1 ; # ngram occurrences including ngram position
      for $integer \in o$ **do**
        $pmi_c[ngram][integer][count]$ += 1 ; # ngram sent with integer in given position
        $pmi_c[ngram][ngram\_pos][integer][count]$ += 1 ; # ngram in given position sent with integer in given position
      end
    end
  end
end
; # Calculate integer NPMI.
for $ngram \in ngrams$ **do**
  ; # Position variant NPMI.
  for $pos \in pmi_c[ngram][ngram\_pos]$ **do**
    $p(integer) = \frac{1}{60}$; # Estimated observation probability for 60 integers
    $integer_p = max(pmi_c[ngram][integer][count])$;; # Find integer with highest co-ocurrence given position
    $ngram_{pos} = pmi_c[ngram][ngram\_pos][count]$;
    $p(ngram_{pos}) = \frac{ngram_{pos}}{L}$
    $p(ngram_{pos}, integer) = \frac{pmi_c[ngram][ngram\_pos][integer][count]}{L}$;
    $h(ngram_{pos}, integer) = -\log_2(p(ngram_{pos}, integer))$;
    $pmi(ngram_{pos}, integer) = \log_2(\frac{p(ngram_{pos}, integer)}{p(ngram_{pos})p(integer)})$;
    $npmi(ngram_{pos}, integer) = \frac{pmi(ngram_{pos}, integer)}{h(ngram_{pos}, integer)}$;
    $pmi_c[ngram][ngram\_pos][integer] = npmi(ngram_{pos}, integer)$;
  end
  ; # Position invariant NPMI.
  $integer = max(pmi_c[ngram][integer][count])$; # Find integer with highest co-ocurrence
  $p(integer) = \frac{1}{60}$; # Estimated observation probability for 60 integers
  $ngram_{total} = pmi_c[ngram][count]$;
  $p(ngram) = \frac{ngram_{total}}{L \times (4 - len(ngram))}$; # If ngram is length 1, it could appear 3 times per message
  $p(ngram, integer) = \frac{pmi_c[ngram][integer][count]}{L}$;
  $h(ngram, integer) = -\log_2(p(ngram, integer))$;
  $pmi(ngram, integer) = \log_2(\frac{p(ngram, integer)}{p(ngram)p(integer)})$;
  $npmi(ngram, integer) = \frac{pmi(ngram, integer)}{h(ngram, integer)}$;
  $pmi_c[ngram][integer] = npmi(ngram, integer)$;
end

Algorithm 4: The PMI_c algorithm cont.

; # Now we identify ngrams corresponding to referent positions.
$ngram_{pr}$ = dict;
; # Prune ngrams with NPMI below c
for $ngram \in pmi_c$ **do**
  for $integer \in pmi_c[ngram]$ **do**
    if $pmi_c[ngram][integer] < t_c$ **then**
      del $pmi_c[ngram][integer]$;
    end
    for $pos \in pmi_c[ngram]$ **do**
      for $integer \in pmi_c[ngram][pos]$ **do**
        if $pmi_c[ngram][pos][integer] < t_c$ **then**
          del $pmi_c[ngram][pos][integer]$;
        end
      end
    end
  end
end
; # Find messages with integer ngrams
for $ngram \in pmi_c[ngram]$ **do**
  for $o, m \in O\_M$ **do**
    ; # Position variant ngram
    if $pmi_c[ngram][pos]$ **then**
      if $ngram \in m[pos]$ **then**
        $new\_ngram = m - ngram$; # Get leftover ngram
        $pr = pos(pmi_c[ngram][pos][integer], msg)$; # Get the possible referent position
        $ngram_{pr}[new\_ngram][pr][count] += 1$; # Count leftover ngram occurence
        $ngram_{pr}[new\_ngram][pos][pr][count] += 1$; # Count leftover ngram occurence in given positions
      end
    end
    ; # Position invariant ngram
    else
      if $ngram \in m$ **then**
        $new\_ngram = m - ngram$; # Get leftover ngram
        $pr = pos(pmi_c[ngram][integer], msg)$; # Get the possible referent position
        $ngram_{pr}[new\_ngram][pr][count] += 1$; # Count leftover ngram occurence
        $ngram_{pr}[new\_ngram][pos][pr][count] += 1$; # Count leftover ngram occurence in given positions
      end
    end
  end
end

Algorithm 5: The PMI_c algorithm cont.

; # Calculate referent position NPMI.
for $ngram \in ngram_{pr}$ **do**
  for $pr \in ngram_{pr}[ngram][pr]$ **do**
    ; # Position variant NPMI.
    for $pos \in ngram_{pr}[ngram][pos][pr]$ **do**
      $p(pr) = 0.98$; # Estimated observation probability for given position
      $ngram_{pos} = ngram_{pr}[ngram][pos][pr][count]$; $p(ngram_{pos}) = \frac{ngram_{pos}}{L}$
      $p(ngram_{pos}, pr) = \frac{ngram_{pr}[ngram][pos][pr][count]}{L}$;
      $h(ngram_{pos}, pr) = -\log_2(p(ngram_{pos}, integer))$;
      $pmi(ngram_{pos}, pr) = \log_2(\frac{p(ngram_{pos}, pr)}{p(ngram_{pos})p(pr)})$;
      $npmi(ngram_{pos}, pr) = \frac{pmi(ngram_{pos}, pr)}{h(ngram_{pos}, pr)}$;
      $pmi_c[ngram][pos][pr] = npmi(ngram_{pos}, pr)$;
    end
    ; # Position invariant NPMI.
    $p(pr) = 0.98$; # Estimated observation probability for given position
    $ngram = max(ngram_{pr}[ngram][pr][count])$; # Find highest spatio-temporal reference count
    $p(ngram) = \frac{ngram}{L}$;
    $p(ngram, pr) = \frac{ngram_{pr}[ngram][pr][count]}{L}$;
    $h(ngram, pr) = -\log_2(p(ngram, integer))$;
    $pmi(ngram, pr) = \log_2(\frac{p(ngram, pr)}{p(ngram)p(pr)})$;
    $npmi(ngram, pr) = \frac{pmi(ngram, pr)}{h(ngram, pr)}$;
    $pmi_c[ngram][pr] = npmi(ngram, pr)$;
  end
end

Glossary

anaphoric A word whose meaning depends on the context of previous communication (*e.g.*... .This means...).

compositionality The possibility of concatenation, or otherwise combination, of atomic symbols of a language to convey new meanings. For example, two atomic symbols of "red" and "dog" create new meaning when combined.

degenerate language A language which assigns all meanings to the same expression. Therefore, it is maximally ambiguous as only one "word" exists in the whole language, but fully compressible (Kirby et al., 2015).

deixis A word whose meaning depends on the context of the speaker or interlocutor (*e.g.*...over there...).

holistic language A language which assigns all meanings their own separate expression. Therefore, it is fully unambiguous, but also incompressible (Kirby et al., 2015).

idiolect The unique language of an individual, which includes the way that the individual slightly changes the grammar, syntactic or semantic structures.

polysemy The capacity for a word or phrase to have multiple distinct meanings.

productivity The ability of the language to have an almost infinite number of uses. In other words, the language can refer to infinitely many concepts, through the usage of rules such as compositionality.

structured language A language which has structure and rules as to how to create meaningful expressions. It is the mid-point between holistic language and degenerate language, where it is compressible, but also ambiguous (Kirby et al., 2015).

References

Prince Abudu and Andrew Markham. Deep Emergent Communication for the IoT. In *2020 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 130–137, 2020. .

Jacob Andreas. Measuring compositionality in representation learning. In *Proc. of ICLR*. OpenReview.net, 2019.

Jacob Andreas, Anca Dragan, and Dan Klein. Translating neuralese. In *Proc. of ACL*, pages 232–242. Association for Computational Linguistics, 2017. .

Yannis Assael, Thea Sommerschield, Brendan Shillingford, Mahyar Bordbar, John Pavlopoulos, Marita Chatzipanagiotou, Ion Androutsopoulos, Jonathan Prag, and Nando de Freitas. Restoring and attributing ancient texts using deep neural networks. *Nature*, 603(7900):280–283, 2022. ISSN 1476-4687. .

Michal Auersperger and Pavel Pecina. Defending compositionality in emergent languages. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop*, pages 285–291. Association for Computational Linguistics, 2022. .

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional AI: Harmlessness from AI Feedback, 2022.

Marco Baroni. Autonomous Linguistic Emergence in Neural Networks (ALiEN) ERC Advanced Grant 2020, 2020a.

Marco Baroni. Linguistic generalization and compositionality in modern artificial neural networks. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 375 (1791):20190307, 2020b. .

Marco Baroni. Rat big, cat eaten! Ideas for a useful deep-agent protolanguage. *ArXiv preprint*, abs/2003.11922, 2020c.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended Long Short-Term Memory, 2024.

Lukas Biewald. Experiment Tracking with Weights and Biases, 2020.

Ben Bogin, Mor Geva, and Jonathan Berant. Emergence of Communication in an Interactive World with Consistent Speakers. *ArXiv preprint*, abs/1809.00549, 2018.

Brendon Boldt and David Mortensen. ELCC: the Emergent Language Corpus Collection, 2024a.

Brendon Boldt and David R. Mortensen. A Review of the Applications of Deep Learning-Based Emergent Communication. *Transactions on Machine Learning Research*, 2024b. ISSN 2835-8856.

Tom Bosc. Varying meaning complexity to explain and measure compositionality. In *Emergent Communication Workshop at ICLR 2022*, 2022.

Tom Bosc and Pascal Vincent. The Emergence of Argument Structure in Artificial Languages. *Transactions of the Association for Computational Linguistics*, 10:1375–1391, 2022. ISSN 2307-387X. .

Diane Bouchacourt and Marco Baroni. Miss tools and mr fruit: Emergent communication in agents learning about object affordances. In *Proc. of ACL*, pages 3909–3918. Association for Computational Linguistics, 2019. .

Gerlof J. Bouma. Normalized (pointwise) mutual information in collocation extraction. In *Von der Form zur Bedeutung: Texte automatisch verarbeiten - From Form to Meaning: Processing Texts Automatically*, volume 30, pages 31–40, 2009. ISBN 978-3-8233-7511-1.

Nicolo' Brandizzi, Davide Grossi, and Luca Iocchi. RLupus: Cooperation through emergent communication in The Werewolf social deduction game. *Intelligenza Artificiale*, 15(2):55–70, 2021. ISSN 2211-0097. .

Henry Brighton. Compositional Syntax From Cultural Transmission. *Artificial Life*, 8(1): 25–54, 2002. ISSN 1064-5462. .

Henry Brighton and Simon Kirby. Understanding Linguistic Evolution by Visualizing the Emergence of Topographic Mappings. *Artificial Life*, 12(2):229–242, 2006. ISSN 1064-5462. .

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 2020 (NeurIPS 2020)*, 2020.

Boaz Carmeli, Yonatan Belinkov, and Ron Meir. Concept-Best-Matching: Evaluating Compositionality in Emergent Communication, 2024.

Rahma Chaabouni, Eugene Kharitonov, Emmanuel Dupoux, and Marco Baroni. Anti-efficient encoding in emergent communication. In *Advances in Neural Information Processing Systems 2019 (NeurIPS 2019)*, pages 6290–6300, 2019.

Rahma Chaabouni, Eugene Kharitonov, Diane Bouchacourt, Emmanuel Dupoux, and Marco Baroni. Compositionality and generalization in emergent languages. In *Proc. of ACL*, pages 4427–4442. Association for Computational Linguistics, 2020. .

Rahma Chaabouni, Eugene Kharitonov, Emmanuel Dupoux, and Marco Baroni. Communicating artificial neural networks develop efficient color-naming systems. *Proceedings of the National Academy of Sciences*, 118(12), 2021. ISSN 0027-8424, 1091-6490. .

Rahma Chaabouni, Florian Strub, Florent Altché, Eugene Tarassov, Corentin Tallec, Elnaz Davoodi, Kory Wallace Mathewson, Olivier Tieleman, Angeliki Lazaridou, and Bilal Piot. Emergent communication at scale. In *Proc. of ICLR*. OpenReview.net, 2022.

Ruxiao Chen and Shuaishuai Guo. Emergent communication for AR. *ArXiv preprint*, abs/2308.07342, 2023.

Emily Cheng, Diego Doimo, Corentin Kervadec, Iuri Macocco, Jade Yu, Alessandro Laio, and Marco Baroni. Emergence of a High-Dimensional Abstraction Phase in Language Transformers, 2024.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proc. of EMNLP*, pages 1724–1734. Association for Computational Linguistics, 2014. .

Edward Choi, Angeliki Lazaridou, and Nando de Freitas. Compositional obverter communication learning from raw visual input. In *Proc. of ICLR*. OpenReview.net, 2018.

Morten H. Christiansen and Simon Kirby. Language evolution: consensus and controversies. *Trends in Cognitive Sciences*, 7(7):300–307, 2003. ISSN 1364-6613. .

Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. In *27th Annual Meeting of the Association for Computational Linguistics*, pages 76–83. Association for Computational Linguistics, 1989. .

Michael Cogswell, Jiasen Lu, Stefan Lee, Devi Parikh, and Dhruv Batra. Emergence of Compositional Language with Deep Generational Transmission. *ArXiv preprint*, abs/1904.09067, 2019.

Henry Conklin and Kenny Smith. Compositionality with Variation Reliably Emerges in Neural Networks. In *The Eleventh International Conference on Learning Representations*, 2022.

WJ Conover and RL Iman. On multiple-comparisons procedures. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 1979.

Dylan Cope and Peter McBurney. Joining the Conversation: Towards Language Acquisition for Ad Hoc Team Play. In *Emergent Communication Workshop at ICLR 2022*, 2022.

Dylan Cope and Peter McBurney. Learning Translations: Emergent Communication Pretraining for Cooperative Language Acquisition. *ArXiv preprint*, abs/2402.16247, 2024.

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de las Casas, Craig Donner, Leslie Fritz, Cristian Galperti, Andrea Huber, James Keeling, Maria Tsimpoukelli, Jackie Kay, Antoine Merle, Jean-Marc Moret, Seb Noury, Federico Pesamosca, David Pfau, Olivier Sauter, Cristian Sommariva, Stefano Coda, Basil Duval, Ambrogio Fasoli, Pushmeet Kohli, Koray Kavukcuoglu, Demis Hassabis, and Martin Riedmiller. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022. ISSN 1476-4687. .

Gregoire Deletang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, and Pedro A. Ortega. Neural Networks and the Chomsky Hierarchy. In *The Eleventh International Conference on Learning Representations*, 2022.

Roberto Dessì, Eugene Kharitonov, and Marco Baroni. Interpretable agent communication from scratch (with a generic visual processor emerging on the side). In *Advances in Neural Information Processing Systems 2021 (NeurIPS 2021)*, pages 26937–26949, 2021.

Tom Eccles, Yoram Bachrach, Guy Lever, Angeliki Lazaridou, and Thore Graepel. Biases for emergent communication in multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems 2019 (NeurIPS 2019)*, pages 13111–13121, 2019.

Nicholas Edwards, Hannah Rohde, and Henry Coxe-Conklin. Anaphoric Structure Emerges Between Neural Networks. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 45(45), 2023.

William Falcon and The PyTorch Lightning Team. PyTorch Lightning, 2019.

Yicheng Feng, Boshi An, and Zongqing Lu. Learning Multi-Object Positional Relationships via Emergent Communication, 2023.

Lukas Galke and Limor Raviv. Emergent communication and learning pressures in language models: a language evolution perspective, 2024.

Lukas Galke, Yoav Ram, and Limor Raviv. Emergent Communication for Understanding Human Language Evolution: What's Missing? In *Emergent Communication Workshop at ICLR 2022*, 2022.

Quentin Garrido, Mahmoud Assran, Nicolas Ballas, Adrien Bardes, Laurent Najman, and Yann LeCun. Learning and Leveraging World Models in Visual Representation Learning, 2024.

M. A. K. Halliday and Ruqaiya Hasan. *Cohesion in English*. Routledge, 1976. ISBN 978-1-315-83601-0. .

Lushan Han, Tim Finin, Paul McNamee, Anupam Joshi, and Yelena Yesha. Improving Word Similarity by Augmenting PMI with Estimates of Word Polysemy. *TKDE*, 25(6): 1307–1322, 2013. .

Laura Harding Graesser, Kyunghyun Cho, and Douwe Kiela. Emergent linguistic phenomena in multi-agent communication games. In *Proc. of EMNLP*, pages 3700–3710. Association for Computational Linguistics, 2019. .

Zellig S. Harris. From Phoneme to Morpheme. *Language*, 31(2):190–222, 1955. ISSN 0097-8507. .

Serhii Havrylov and Ivan Titov. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. In *Advances in Neural Information Processing Systems 2017 (NeurIPS 2017)*, pages 2149–2159, 2017.

Felix Hill, Andrew K. Lampinen, Rosalia Schneider, Stephen Clark, Matthew Botvinick, James L. McClelland, and Adam Santoro. Environmental drivers of systematicity and generalization in a situated agent. In *Proc. of ICLR*. OpenReview.net, 2020.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997. ISSN 0899-7667. .

Charles F Hockett. The origin of speech. *Scientific American*, 203(3):88–97, 1960.

Sture Holm. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. ISSN 0303-6898.

Zhuoqiao Hong, Haocheng Wang, Zaid Zada, Harshvardhan Gazula, David Turner, Bobbi Aubrey, Leonard Niekerken, Werner Doyle, Sasha Devore, Patricia Dugan, Daniel Friedman, Orrin Devinsky, Adeen Flinker, Uri Hasson, Samuel A. Nastase, and Ariel Goldstein. Scale matters: Large language models with billions (rather than millions) of parameters better match neural representations of natural language, 2024.

Guiyang Hou, Wenqi Zhang, Yongliang Shen, Linjuan Wu, and Weiming Lu. TimeToM: Temporal Space is the Key to Unlocking the Door of Large Language Models' Theory-of-Mind, 2024.

David A. Huffman. A Method for the Construction of Minimum-Redundancy Codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952. ISSN 2162-6634. .

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *Proc. of ICLR*. OpenReview.net, 2017.

Daniel Kahneman. *Thinking, fast and slow*. Farrar, Straus and Giroux, 2011.

Aleksandra Kalinowska, Elnaz Davoodi, Florian Strub, Kory Mathewson, Todd Murphey, and Patrick Pilarski. Situated Communication: A Solution to Over-communication between Artificial Agents. In *Emergent Communication Workshop at ICLR 2022*, 2022.

Yipeng Kang, Tonghan Wang, and Gerard de Melo. Incorporating pragmatic reasoning communication into emergent language. In *Advances in Neural Information Processing Systems 2020 (NeurIPS 2020)*, 2020.

Ruth Kempson. Pragmatics: Language and Communication. In *The Handbook of Linguistics*, pages 394–424. John Wiley & Sons, Ltd, 2003. ISBN 978-0-470-75640-9. .

Eugene Kharitonov and Marco Baroni. Emergent language generalization and acquisition speed are not tied to compositionality. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 11–15. Association for Computational Linguistics, 2020. .

Eugene Kharitonov, Rahma Chaabouni, Diane Bouchacourt, and Marco Baroni. EGG: a toolkit for research on emergence of lanGuage in games. In *Proc. of EMNLP*, pages 55–60. Association for Computational Linguistics, 2019. .

Eugene Kharitonov, Rahma Chaabouni, Diane Bouchacourt, and Marco Baroni. Entropy minimization in emergent languages. In *Proc. of ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 5220–5230. PMLR, 2020.

Najoung Kim and Paul Smolensky. Structural Generalization of Modification in Adult Learners of an Artificial Language, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *Proc. of ICLR*, 2015.

Simon Kirby. Syntax out of Learning: The Cultural Evolution of Structured Communication in a Population of Induction Algorithms. In Dario Floreano, Jean-Daniel Nicoud, and Francesco Mondada, editors, *Advances in Artificial Life, 5th European Conference, ECAL'99, Lausanne, Switzerland, September 13-17, 1999, Proceedings*, volume 1674 of *Lecture Notes in Computer Science*, pages 694–703. Springer, 1999. .

Simon Kirby, Monica Tamariz, Hannah Cornish, and Kenny Smith. Compression and communication in the cultural evolution of linguistic structure. *Cognition*, 141:87–102, 2015. ISSN 0010-0277. .

Donald E Knuth. Dynamic huffman coding. *Journal of Algorithms*, 6(2):163–180, 1985. ISSN 0196-6774. .

Kavya Kopparapu, Edgar A. Duéñez-Guzmán, Jayd Matyas, Alexander Sasha Vezhnevets, John P. Agapiou, Kevin R. McKee, Richard Everett, Janusz Marecki, Joel Z. Leibo, and Thore Graepel. Hidden Agenda: a Social Deduction Game with Diverse Learned Equilibria. *ArXiv preprint*, abs/2201.01816, 2022.

Tomasz Korbak, Julian Zubek, and Joanna Raczaszek-Leonardi. Measuring non-trivial compositionality in emergent communication. In *4th Workshop on Emergent Communication, NeurIPS 2020*, 2020.

Kepa Korta and John Perry. Pragmatics. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2020 edition, 2020.

Satwik Kottur, José Moura, Stefan Lee, and Dhruv Batra. Natural language does not emerge 'naturally' in multi-agent dialog. In *Proc. of EMNLP*, pages 2962–2967. Association for Computational Linguistics, 2017. .

M. A. Kramer. Autoassociative neural networks. *Computers & Chemical Engineering*, 16 (4):313–328, 1992. ISSN 0098-1354. .

William H. Kruskal and W. Allen Wallis. Use of Ranks in One-Criterion Variance Analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952. ISSN 0162-1459. .

Lukasz Kucinski, Tomasz Korbak, Pawel Kolodziej, and Piotr Milos. Catalytic role of noise and necessity of inductive biases in the emergence of compositional communication. In *Advances in Neural Information Processing Systems 2021 (NeurIPS 2021)*, pages 23075–23088, 2021.

Tatsuki Kuribayashi, Ryo Ueda, Ryo Yoshida, Yohei Oseki, Ted Briscoe, and Timothy Baldwin. Emergent Word Order Universals from Cognitively-Motivated Language Models, 2024.

Shalom Lappin. An Introduction to Formal Semantics. In *The Handbook of Linguistics*, pages 369–393. John Wiley & Sons, Ltd, 2003. ISBN 978-0-470-75640-9. .

Ammar Ahmed Pallikonda Latheef, Alberto Santamaria-Pang, Craig K. Jones, and Haris I. Sair. Emergent Language Symbolic Autoencoder (ELSA) with Weak Supervision to Model Hierarchical Brain Networks, 2024.

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

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language. In *Proc. of ICLR*. OpenReview.net, 2017.

Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. Emergence of linguistic communication from referential games with symbolic and pixel input. In *Proc. of ICLR*. OpenReview.net, 2018.

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

Fushan Li and Michael Bowling. Ease-of-teaching and language structure from emergent communication. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 15825–15835, 2019.

Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph Gonzalez, Michael I. Jordan, and Ion Stoica. Rllib: Abstractions for distributed reinforcement learning. In Jennifer G. Dy and Andreas Krause, editors, *Proc. of ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 3059–3068. PMLR, 2018.

Paul Pu Liang, Jeffrey Chen, Ruslan Salakhutdinov, Louis-Philippe Morency, and Satwik Kottur. On Emergent Communication in Competitive Multi-Agent Teams. In Amal El Fallah Seghrouchni, Gita Sukthankar, Bo An, and Neil Yorke-Smith, editors, *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, pages 735–743. International Foundation for Autonomous Agents and Multiagent Systems, 2020.

Orna Lichtenstein, Amir Pnueli, and Lenore Zuck. The glory of the past. In *Workshop on Logic of Programs*, pages 196–218. Springer, 1985.

Jia Peng Lim and Hady W. Lauw. Aligning Human and Computational Coherence Evaluations. *Computational Linguistics*, pages 1–58, 2024. ISSN 0891-2017. .

Olaf Lipinski. emlangkit: Emergent Language Analysis Toolkit, 2023. URL https://github.com/olipinski/emlangkit.

Olaf Lipinski, Adam Sobey, Federico Cerutti, and Timothy J. Norman. Emergent Password Signalling in the Game of Werewolf. In *Emergent Communication Workshop at ICLR 2022*, 2022.

Olaf Lipinski, Adam J. Sobey, Federico Cerutti, and Timothy J. Norman. It's About Time: Temporal References in Emergent Communication. *ArXiv preprint*, abs/2310.06555, 2023.

Olaf Lipinski, Adam Sobey, Federico Cerutti, and Timothy J. Norman. Speaking Your Language: Spatial Relationships in Interpretable Emergent Communication. In *Advances in Neural Information Processing Systems 2024 (NeurIPS 2024)*, 2024.

Ryan Lowe, Jakob N. Foerster, Y.-Lan Boureau, Joelle Pineau, and Yann N. Dauphin. On the Pitfalls of Measuring Emergent Communication. In Edith Elkind, Manuela Veloso, Noa Agmon, and Matthew E. Taylor, editors, *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, pages 693–701. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

John Lyons. Deixis, space and time. In *Semantics*, volume 2, pages 636–724. Cambridge University Press, 1977. ISBN 978-0-521-29186-6. .

Matéo Mahaut, Francesca Franzon, Roberto Dessì, and Marco Baroni. Referential communication in heterogeneous communities of pre-trained visual deep networks, 2023.

Oded Maler, Dejan Nickovic, and Amir Pnueli. Checking Temporal Properties of Discrete, Timed and Continuous Behaviors. In Arnon Avron, Nachum Dershowitz, and Alexander Rabinovich, editors, *Pillars of Computer Science: Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday*, Lecture Notes in Computer Science, pages 475–505. Springer, 2008. ISBN 978-3-540-78127-1. .

Yuri I. Manin. Zipf's law and L. Levin probability distributions. *Functional Analysis and Its Applications*, 48(2):116–127, 2014. ISSN 1573-8485. .

Daniel J. Mankowitz, Andrea Michi, Anton Zhernov, Marco Gelmi, Marco Selvi, Cosmin Paduraru, Edouard Leurent, Shariq Iqbal, Jean-Baptiste Lespiau, Alex Ahern, Thomas Köppe, Kevin Millikin, Stephen Gaffney, Sophie Elster, Jackson Broshear, Chris Gamble, Kieran Milan, Robert Tung, Minjae Hwang, Taylan Cemgil, Mohammadamin Barekatain, Yujia Li, Amol Mandhane, Thomas Hubert, Julian Schrittwieser, Demis Hassabis, Pushmeet Kohli, Martin Riedmiller, Oriol Vinyals, and David Silver. Faster sorting algorithms discovered using deep reinforcement learning. *Nature*, 618(7964):257–263, 2023. ISSN 1476-4687. .

Nestor Maslej, Loredana Fattorini, Raymond Perrault, Vanessa Parli, Anka Reuel, Erik Brynjolfsson, John Etchemendy, Katrina Ligett, Terah Lyons, James Manyika, Juan Carlos Niebles, Yoav Shoham, Russell Wald, and Jack Clark. Artificial Intelligence Index Report 2024, 2024.

Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010. .

William Merrill and Ashish Sabharwal. The Parallelism Tradeoff: Limitations of Log-Precision Transformers, 2022.

William Merrill, Jackson Petty, and Ashish Sabharwal. The Illusion of State in State-Space Models, 2024.

Alex Mesoudi, Alan G McElligott, and David Adger. Introduction: Integrating Genetic and Cultural Evolutionary Approaches to Language. *Human Biology*, 83(2):141–151, 2011.

Paul Michel, Mathieu Rita, Kory Wallace Mathewson, Olivier Tieleman, and Angeliki Lazaridou. Revisiting Populations in multi-agent Communication. In *The Eleventh International Conference on Learning Representations*, 2022.

Daniela Mihai and Jonathon S. Hare. Learning to draw: Emergent communication through sketching. In *Advances in Neural Information Processing Systems 2021 (NeurIPS 2021)*, pages 7153–7166, 2021.

George A. Miller. WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*, 1992.

Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proc. of AAAI*, pages 1495–1502. AAAI Press, 2018.

Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A Distributed Framework for Emerging AI Applications. *ArXiv preprint*, abs/1712.05889, 2017.

Salwa Mostafa, Mohammed S. Elbamby, Mohamed K. Abdel-Aziz, and Mehdi Bennis. Intent Profiling and Translation Through Emergent Communication, 2024a.

Salwa Mostafa, Mateus P. Mota, Alvaro Valcarce, and Mehdi Bennis. Emergent Communication Protocol Learning for Task Offloading in Industrial Internet of Things, 2024b.

Jesse Mu and Noah D. Goodman. Emergent communication of generalizations. In *Advances in Neural Information Processing Systems 2021 (NeurIPS 2021)*, pages 17994–18007, 2021.

Salmane Naoumi, Reda Alami, Hakim Hacid, Ebtesam Almazrouei, Merouane Debbah, Mehdi Bennis, and Marwa Chafii. Emergent Communication in Multi-Agent Reinforcement Learning for Flying Base Stations. In *2023 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, pages 133–138, 2023. .

Mitja Nikolaus. Emergent Communication with Conversational Repair. In *The Twelfth International Conference on Learning Representations*, 2023.

Michael Noukhovitch, Travis LaCroix, Angeliki Lazaridou, and Aaron C. Courville. Emergent Communication under Competition. In Frank Dignum, Alessio Lomuscio, Ulle Endriss, and Ann Nowé, editors, *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, pages 974–982. ACM, 2021.

Xenia Ohmer, Marko Duda, and Elia Bruni. Emergence of hierarchical reference systems in multi-agent communication. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5689–5706. International Committee on Computational Linguistics, 2022a.

Xenia Ohmer, Michael Marino, Michael Franke, and Peter König. Mutual influence between language and perception in multi-agent communication games. *PLOS Computational Biology*, 18(10):e1010658, 2022b. ISSN 1553-7358. .

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris,

Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, C. J. Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. GPT-4 Technical Report, 2023.

Marie Ossenkopf, Kevin Sebastian Luck, and Kory Wallace Mathewson. Which Language Evolves Between Heterogeneous Agents? - Communicating Movement Instructions With Widely Different Time Scopes. In *Emergent Communication Workshop at ICLR 2022*, 2022.

Denis Paperno and Marco Baroni. Squibs: When the whole is less than the sum of its parts: How composition affects PMI values in distributional semantic vectors. *Computational Linguistics*, 42(2):345–350, 2016. .

Hugh Perkins. Neural networks can understand compositional functions that humans do not, in the context of emergent communication. *ArXiv preprint*, abs/2103.04180, 2021.

Steven Pinker and Paul Bloom. Natural language and natural selection. *Behavioral and Brain Sciences*, 13(4):707–727, 1990. .

Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 46–57. ieee, 1977.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. *OpenAI*, 2018.

Daking Rai, Yilun Zhou, Shi Feng, Abulhair Saparov, and Ziyu Yao. A Practical Review of Mechanistic Interpretability for Transformer-Based Language Models, 2024.

Yi Ren, Shangmin Guo, Matthieu Labeau, Shay B. Cohen, and Simon Kirby. Compositional languages emerge in a neural iterated learning model. In *Proc. of ICLR*. OpenReview.net, 2020.

Ryokan Ri, Ryo Ueda, and Jason Naradowsky. Emergent Communication with Attention, 2023.

Mathieu Rita, Rahma Chaabouni, and Emmanuel Dupoux. "LazImpa": Lazy and impatient neural agents learn to communicate efficiently. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 335–343. Association for Computational Linguistics, 2020. .

Mathieu Rita, Florian Strub, Jean-Bastien Grill, Olivier Pietquin, and Emmanuel Dupoux. On the role of population heterogeneity in emergent communication. In *Proc. of ICLR*. OpenReview.net, 2022a.

Mathieu Rita, Corentin Tallec, Paul Michel, Jean-Bastien Grill, Olivier Pietquin, Emmanuel Dupoux, and Florian Strub. Emergent Communication: Generalization and Overfitting in Lewis Games. In *Advances in Neural Information Processing Systems 2022, NeurIPS 2022*, 2022b.

Mathieu Rita, Paul Michel, Rahma Chaabouni, Olivier Pietquin, Emmanuel Dupoux, and Florian Strub. Language Evolution with Deep Learning, 2024.

Diana Rodríguez Luna, Edoardo Maria Ponti, Dieuwke Hupkes, and Elia Bruni. Internal and external pressures on language emergence: least effort, object constancy and frequency. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4428–4437. Association for Computational Linguistics, 2020. .

John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems 2015 (NeurIPS 2015)*, pages 3528–3536, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, 2017.

David Silver, Satinder Singh, Doina Precup, and Richard S. Sutton. Reward is enough. *Artificial Intelligence*, 299:103535, 2021. ISSN 0004-3702. .

Kenny Smith, Simon Kirby, and Henry Brighton. Iterated learning: a framework for the emergence of language. *Artificial Life*, 9(4):371–386, 2003. ISSN 1064-5462. .

C. Spearman. The Proof and Measurement of Association between Two Things. *The American Journal of Psychology*, 15(1):72–101, 1904. ISSN 0002-9556. .

Andreea Stapleton. Deixis in Modern Linguistics. *Essex Student Journal*, 9(1), 2017. ISSN 2633-7045. .

Luc Steels and Frederic Kaplan. Collective learning and semiotic dynamics. In *European Conference on Artificial Life*, pages 679–688. Springer, 1999.

Shane Steinert-Threlkeld. Toward the Emergence of Nontrivial Compositionality. *Philosophy of Science*, 87(5):897–909, 2020. ISSN 0031-8248, 1539-767X. .

Shane Steinert-Threlkeld, Xuhui Zhou, Zeyu Liu, and C. M. Downey. Emergent Communication Fine-tuning (EC-FT) for Pretrained Language Models. In *Emergent Communication Workshop at ICLR 2022*, 2022.

Zoltán Gendler Szabó. Compositionality. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, fall 2020 edition, 2020.

Valentin Taillandier, Dieuwke Hupkes, Benoît Sagot, Emmanuel Dupoux, and Paul Michel. Neural Agents Struggle to Take Turns in Bidirectional Emergent Communication. In *The Eleventh International Conference on Learning Representations*, 2023.

Mingtian Tan, Mike A. Merrill, Vinayak Gupta, Tim Althoff, and Thomas Hartvigsen. Are Language Models Actually Useful for Time Series Forecasting?, 2024.

The pandas development team. pandas-dev/pandas: Pandas, 2020.

Anton Thielmann, Arik Reuter, Quentin Seifert, Elisabeth Bergherr, and Benjamin Säfken. Topics in the Haystack: Enhancing Topic Quality through Corpus Expansion. *Computational Linguistics*, pages 1–37, 2024. ISSN 0891-2017. .

Fujio Toriumi, Hirotaka Osawa, Michimasa Inaba, Daisuke Katagami, Kosuke Shinoda, and Hitoshi Matsubara. AI Wolf Contest — Development of Game AI Using Collective Intelligence. In Tristan Cazenave, Mark H.M. Winands, Stefan Edelkamp, Stephan Schiffel, Michael Thielscher, and Julian Togelius, editors, *Computer Games*, Communications in Computer and Information Science, pages 101–115. Springer International Publishing, 2017. ISBN 978-3-319-57969-6. .

Mycal Tucker, Huao Li, Siddharth Agrawal, Dana Hughes, Katia P. Sycara, Michael Lewis, and Julie A. Shah. Emergent discrete communication in semantic spaces. In *Advances in Neural Information Processing Systems 2021 (NeurIPS 2021)*, pages 10574–10586, 2021.

Mycal Tucker, Roger P. Levy, Julie Shah, and Noga Zaslavsky. Trading off Utility, Informativeness, and Complexity in Emergent Communication. In *Advances in Neural Information Processing Systems*, 2022.

A. M. Turing. Computing Machinery and Intelligence. *Mind*, LIX(236):433–460, 1950. ISSN 0026-4423. .

Ryo Ueda and Tadahiro Taniguchi. Lewis's Signaling Game as beta-VAE For Natural Word Lengths and Segments, 2023.

Ryo Ueda and Koki Washio. On the relationship between Zipf's law of abbreviation and interfering noise in emergent languages. In *Proc. of ACL*, pages 60–70. Association for Computational Linguistics, 2021. .

Ryo Ueda, Taiga Ishii, Koki Washio, and Yusuke Miyao. Categorial Grammar Induction as a Compositionality Measure for Emergent Languages in Signaling Games. In *Emergent Communication Workshop at ICLR 2022*, 2022.

Ryo Ueda, Taiga Ishii, and Yusuke Miyao. On the Word Boundaries of Emergent Languages Based on Harris's Articulation Scheme. In *The Eleventh International Conference on Learning Representations*, 2023.

Oskar van der Wal, Silvan de Boer, Elia Bruni, and Dieuwke Hupkes. The grammar of emergent languages. In *Proc. of EMNLP*, pages 3339–3359. Association for Computational Linguistics, 2020. .

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 2017 (NeurIPS 2017)*, pages 5998–6008, 2017.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan

Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. .

Jeffrey Scott Vitter. Design and analysis of dynamic Huffman codes. *Journal of the ACM*, 34(4):825–845, 1987. ISSN 0004-5411. .

Paul Vogt. The emergence of compositional structures in perceptually grounded language games. *Artificial Intelligence*, 167(1):206–242, 2005. ISSN 0004-3702. .

Alex Warstadt and Samuel R. Bowman. What artificial neural networks can tell us about human language acquisition. In *Algebraic Structures in Natural Language*, pages 17–60. CRC Press, 2022.

Ronald J. Williams. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.*, 8:229–256, 1992. .

Zhenlin Xu, Marc Niethammer, and Colin A. Raffel. Compositional Generalization in Unsupervised Compositional Representation Learning: A Study on Disentanglement and Emergent Language. *Advances in Neural Information Processing Systems 2022 (NeurIPS 2022)*, 35:25074–25087, 2022.

Ryosuke Yamaki, Tadahiro Taniguchi, and Daichi Mochihashi. Holographic CCG Parsing. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 262–276. Association for Computational Linguistics, 2023. .

Shunyu Yao, Mo Yu, Yang Zhang, Karthik R. Narasimhan, Joshua B. Tenenbaum, and Chuang Gan. Linking emergent and natural languages via corpus transfer. In *Proc. of ICLR*. OpenReview.net, 2022.

Yongjing Yin, Lian Fu, Yafu Li, and Yue Zhang. On compositional generalization of transformer-based neural machine translation. *Information Fusion*, 111:102491, 2024. ISSN 1566-2535. .

Enshuai Zhou, Yifan Hao, Rui Zhang, Yuxuan Guo, Zidong Du, Xishan Zhang, Xinkai Song, Chao Wang, Xuehai Zhou, Jiaming Guo, Qi Yi, Shaohui Peng, Di Huang, Ruizhi Chen, Qi Guo, and Yunji Chen. Emergent Communication for Numerical Concepts Generalization. *Proc. of AAAI*, 38(16):17609–17617, 2024. ISSN 2374-3468. .

George Kingsley Zipf. *Human behavior and the principle of least effort.* Human behavior and the principle of least effort. Addison-Wesley Press, 1949.