Université de Montréal

On Iterated learning for Task-Oriented Dialogue

par Soumye Singhal

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des arts et des sciences
en vue de l'obtention du grade de Maître ès sciences (M.Sc.)
en informatique

Jan, 2022

Université de Montréal
Faculté des arts et des sciences

Ce mémoire intitulé:

On Iterated learning for Task-Oriented Dialogue

présenté par:

Soumye Singhal

a été évalué par un jury composé des personnes suivantes:

Pierre-Luc Bacon,	président-rapporteur
Aaron Courville,	directeur de recherche
Aishwarya Agrawal,	membre du jury

Mémoire accepté le: . . . . . .

Résumé

Dans le traitement de langue et des système de dialogue, il est courant de pré-entraîner des modèles de langue sur corpus humain avant de les affiner par le biais d'un simulateur et de résolution de tâches. Malheuresement, ce type d'entrainement tend aussi à induire un phénomène connu sous le nom de dérive du langage. Concrétement, les propriétés syntaxiques et sémantiques de la langue intiallement apprise se détériorent: les agents se concentrent uniquement sur la résolution de la tâche, et non plus sur la préservation de la langue. En s'inspirant des travaux en sciences cognitives, et notamment l'apprentigssage itératif Kirby and Griffiths (2014), nous proposons ici une approche générique pour contrer cette dérive du langage. Nous avons appelé cette méthode *Seeded iterated learning* (SIL), ou apprentissage itératif capitalisé. Ce travail a été publié sous le titre (Lu et al., 2020b) et est présenté au chapitre 2. Afin d'émuler la transmission de la langue entre chaque génération d'agents, un agent étudiant est d'abord pré-entrainé avant d'être affiné de manière itérative, et ceci, en imitant des données échantillonnées à partir d'un agent enseignant nouvellement formé. À chaque génération, l'enseignant est créé en copiant l'agent étudiant, avant d'être de nouveau affiné en maximisant le taux de réussite de la tâche sous-jacente. Dans un second temps, nous présentons *Supervised Seeded iterated learning* (SSIL) dans le chapitre 3, où apprentissage itératif capitalisé avec supervision, qui a été publié sous le titre (Lu et al., 2020a). SSIL s'appuie sur SIL en le combinant avec une autre méthode populaire appelée *Supervised SelfPlay* (S2P) (Gupta et al., 2019), où apprentissage supervisé par auto-jeu. SSIL est capable d'atténuer les problèmes de S2P et de SIL, i.e. la dérive du langage dans les dernier stades de l'entrainement tout en préservant une plus grande diversité linguistique. Tout d'abord, nous évaluons nos méthodes dans sous la forme d'une preuve de concept à traver le Jeu de Lewis avec du langage synthetique. Dans un second temps, nous l'étendons à un jeu de traduction se utilisant du langage naturel. Dans les deux cas, nous soulignons l'efficacité de nos méthodes par rapport aux autres méthodes de la litterature.

Dans le chapitre 1, nous discutons des concepts de base nécessaires à la compréhension des articles présentés dans les chapitres 2 et 3. Nous décrivons le problème spécifique du dialogue orienté tâche, y compris les approches actuelles et les défis auxquels ils sont confrontés : en particulier, la dérive linguistique. Nous donnons également un aperçu du cadre d'apprentissage itéré. Certaines sections du chapitre 1 sont empruntées aux articles pour des raisons de cohérence et de facilité de compréhension. Le chapitre 2 comprend les travaux publiés sous le nom de (Lu et al., 2020b) et le chapitre 3 comprend les travaux publiés sous le nom de (Lu et al., 2020a), avant de conclure au chapitre 4.

mots-clés: apprentissage en profondeur, apprentissage multi-agents, dialogue orienté tâche, apprentissage itératif, traitement du langage naturel, apprentissage multi-tâches, dérive du langage

Summary

In task-oriented dialogue, pretraining on human corpus followed by finetuning in a simulator using selfplay suffers from a phenomenon called language drift. The syntactic and semantic properties of the learned language deteriorates as the agents only focuses on solving the task. Inspired by the iterative learning framework in cognitive science Kirby and Griffiths (2014), we propose a generic approach to counter language drift called *Seeded iterated learning* (SIL). This work was published as (Lu et al., 2020b) and is presented in Chapter 2. In an attempt to emulate transmission of language between generations, a pretrained student agent is iteratively refined by imitating data sampled from a newly trained teacher agent. At each generation, the teacher is created by copying the student agent, before being finetuned to maximize task completion. We further introduce *Supervised Seeded iterated learning* (SSIL) in Chapter 3, work which was published as (Lu et al., 2020a). SSIL builds upon SIL by combining it with the other popular method called *Supervised SelfPlay* (S2P) (Gupta et al., 2019). SSIL is able to mitigate the problems of both S2P and SIL namely late-stage training collapse and low language diversity. We evaluate our methods in a toy setting of Lewis Game, and then scale it up to the translation game with natural language. In both settings, we highlight the efficacy of our methods compared to the baselines.

In Chapter 1, we talk about the core concepts required for understanding the papers presented in Chapters 2 and 3. We describe the specific problem of task-oriented dialogue including current approaches and the challenges they face: particularly, the challenge of language drift. We also give an overview of the iterated learning framework. Some sections in Chapter 1 are borrowed from the papers for coherence and ease of understanding. Chapter 2 comprises of the work published as (Lu et al., 2020b) and Chapter 3 comprises of the work published as (Lu et al., 2020a). Chapter 4 gives a conclusion on the work.

Keywords: deep learning, multi-agent learning, task-oriented dialogue, iterated learning, natural language processing, multi-task learning, language drift

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

AI	Artificial Intelligence
NLP	Natural Language Processing
ML	Machine Learning
MLP	Multi-Layer Perceptron aka fully-connected NN
NN	Neural Network
LSTM	Long-Short Term Memory (Hochreiter and Schmidhuber, 1997)
RNN	Recurent Neural Network
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
SGD	Stochastic Gradient Descent
NLL	Negative Log Likelihood
BPE	Byte-Pair Encoding
S/R Game	Sender-Receiver Game
GSTE	Gumbel Straight-Through Estimator (Jang et al., 2017a)
S2P	Supervised Self-Play (Gupta et al., 2019)
SIL	Seeded Iterated Learning (Lu et al., 2020b)
SSIL	Supervised Seeded Iterated Learning (Lu et al., 2020a)

Acknowledgments

I would like to dedicate this thesis to all my friends and family who continue to inspire me. I also would like to thank my collaborators Yuchen, Florian, Olivier, Sai, and Pau from whom I learned a lot. Finally, I would like to thank my advisor, Aaron Courville, for giving me the guidance and time that helped me become better as a researcher.

1 Introduction

1.1 Machine Learning

Artificial Intelligence (AI) has been a long-standing subfield of computer science aimed at understanding intelligence and developing machines capable of exhibiting intelligent behavior (Russell and Norvig, 2016). The early efforts to develop AI involved classical rule-based programming where the machine designer tries to distill the intelligent behavior into a fixed set of rules. Then, given an input, the machine can follow the set of rules to arrive at the corresponding action. Trying to define a fixed set of rules ourselves is quite challenging and limits the application of these classical approaches. This thesis is based on the approach of using Machine Learning (ML) to build intelligent systems. ML is one of the most popular branches of AI research. It takes the approach of learning the patterns automatically from data and experience rather than explicitly programming them. More recently, the increased access to computing and large datasets has made training large neural network models with stochastic gradient descent possible. This approach of training large deep neural networks with many layers is referred to as Deep Learning (LeCun et al., 2015). It has led to many breakthroughs in machine learning and its applications in speech, computer vision (Krizhevsky et al., 2012), and natural language processing (Adiwardana et al., 2020; Radford et al., 2019). This thesis deals with a particular area of natural language processing called task-oriented dialogue. We focus primarily on applying deep learning methods to this task and analyzing their limitations.

Machine learning is usually split into three subdomains depending on the training data and signal provided. First is **supervised learning**, in which we are given a dataset of inputs and targets, and the goal is to learn the mapping between them using function approximation or probabilistic modeling. The classical supervised learning tasks are classification and regression. Second is **unsupervised** learning, where we are provided with

raw data without any human supervised labels, and the goal is to discover underlying patterns in the data. Clustering data, density estimation, and language modeling are some examples. Third, we have **Reinforcement learning**, where an interactive environment is given and the environment rewards specific actions. The goal then is to discover the behavior to maximize the rewards. This thesis will primarily involve concepts from supervised and unsupervised learning. A background in reinforcement learning is helpful but not necessary.

The following sections will go over the relevant concepts required to understand this thesis. We assume basic familiarity with fundamental concepts in deep learning and natural language processing including: feedforward neural networks, long-short term memory (LSTMs), attention mechanisms, word-embeddings, regularization, and gradient-based optimization methods implementing backpropagation using Stochastic Gradient Descent (SGD). For an in-depth background on machine learning, deep learning, and natural language processing, the readers are encouraged to look at references like (Bishop, 2006; Goodfellow et al., 2016; Murphy, 2012).

1.1.1 Supervised learning

In supervised learning, we are given a dataset $\mathcal{D} = \{\boldsymbol{x}^n, \boldsymbol{y}^n\}_1^N$ of input $\boldsymbol{x}$ and target $\boldsymbol{y}$ and we model it using, say, a neural network f with parameters $\boldsymbol{\theta}$. Given input $\boldsymbol{x}$, the network outputs $\hat{\boldsymbol{y}} = f(\boldsymbol{x}; \boldsymbol{\theta})$. We then compute the loss between output $\hat{\boldsymbol{y}}$ and the target $\boldsymbol{y}$ using a loss function ℓ like the mean-square error or binary cross-entropy. This loss is summed for all points in the dataset to get the empirical loss:

$$\mathcal{L}(\boldsymbol{\theta}|\mathcal{D}) = \frac{1}{N} \sum_n \ell\left(f\left(\boldsymbol{x}^n; \boldsymbol{\theta}\right), \boldsymbol{y}^n\right) \tag{1.1}$$

This loss is then minimized, and this process is called Empirical Risk Minimization.

Negative Log Likelihood (NLL) is a very common choice for the loss function in machine learning. Given data $\mathcal{D}$, model-parameters $\boldsymbol{\theta}$ and likelihood model $P(\boldsymbol{x}|\boldsymbol{\theta})$:

$$NLL = -\log P(\mathcal{D}|\boldsymbol{\theta}) = -\sum_n \log P(\boldsymbol{x}^n \mid \boldsymbol{\theta}) \tag{1.2}$$

This essentially corresponds to maximizing the likelihood of the data. The higher the likelihood, the better the parameters can explain the data. Consider classification with K classes where the output of the neural network is a vector of class probabilities i.e. $f\left(\boldsymbol{x}^n;\boldsymbol{\theta}\right)_k = P(y=k|\boldsymbol{x}^n)$ for $1 \leq k \leq K$. Assuming multinomial likelihood, the NLL loss becomes cross-entropy loss, i.e.:

$$NLL = -\sum_n \sum_k 1_{y=k} \log f\left(\boldsymbol{x}^n;\boldsymbol{\theta}\right)_k \tag{1.3}$$

$$= -\sum_n \log f\left(\boldsymbol{x}^n;\boldsymbol{\theta}\right)_y \tag{1.4}$$

1.1.2 Student-teacher methods

Knowledge distillation is a method originally intended to do model compression (Hinton et al., 2015; Kim and Rush, 2016). It has been observed that bigger and deeper models have better generalization than shallow models. So a large over-parametrized neural network with parameters $\boldsymbol{\theta}^T$ is trained on the supervised dataset $\mathcal{D} = \{\boldsymbol{x}^n, \boldsymbol{y}^n\}_1^N$ to get an accuracy which a smaller model with parameters $\boldsymbol{\theta}^S$ would not have been able to get. Afterwards, this large model serves as a teacher to train the student model. Knowledge distillation essentially involves minimizing the distance between the student and the teacher class probability distributions. We use KL-divergence as the distance metric between two probability distributions P and Q.

$$D_{KL}\left(P||Q\right) = -\sum_{\boldsymbol{x}\in\mathbb{X}} P(\boldsymbol{x}) \log\left(\frac{Q(\boldsymbol{x})}{P(\boldsymbol{x})}\right) \tag{1.5}$$

$$\mathcal{L}\left(\boldsymbol{\theta}^S|\mathcal{D}\right) = \sum_n D_{KL}\left[f\left(\boldsymbol{x}^n;\boldsymbol{\theta}^T,\tau\right)||f\left(\boldsymbol{x}^n;\boldsymbol{\theta}^S,\tau\right)\right] \tag{1.6}$$

Here τ denotes the temperature used in the softmax applied to the logits of the neural networks. It is used to make a trade-off between learning from soft or hard teacher labels. Specifically, for the case of classification, this reduces to the cross-entropy loss:

$$\mathcal{L}\left(\boldsymbol{\theta}^S|\mathcal{D}\right) = -\sum_n \sum_k f\left(\boldsymbol{x}^n;\boldsymbol{\theta}^T,\tau\right)_k \log f\left(\boldsymbol{x}^n;\boldsymbol{\theta}^S,\tau\right)_k \tag{1.7}$$

It is observed that using this scheme; the much smaller student model is able to approach the performance of the larger teacher model.

In recent times, this student-teacher training process has seen applications beyond compression via distillation, like improvements in generalization (Mobahi et al., 2020; Kim et al., 2020). These involve either distilling an ensemble of teacher models into a single one or using the same teacher and student model, a regime referred to as **self-distillation**.

Another application of student-teacher methods is in semi-supervised learning. In supervised learning, the more data fed to the models, the better it learns. In the real world, labeled data is often quite limited because labeling data is very time-consuming and expensive. However, we have access to large amounts of unlabelled data. In semi-supervised learning, we make the best use of this unlabelled data by using the model's own predictions as pseudo-labels. This is then used to augment the original labeled dataset (He et al., 2020; Xie et al., 2019). The specific technique of **self-training** involves the following steps:

1. Train supervised classifier called teacher on labeled data.
2. Use this teacher classifier to predict class labels on the unlabeled data. In practice, we retain only the class labels with confidence higher than some threshold. These predicted labels are clubbed with the unlabeled data to get pseudo-labeled data.
3. Train a student model on the combined labeled and pseudo-labeled data.

This student-teacher training process can be iterated until convergence. A downside of these methods is that some of the pseudo-labels are bound to be incorrect, which can hamper the learning as we scale. To alleviate this problem, noise is added to the student's input like random data augmentation and dropout. This forces the student to learn harder from the pseudo labels, and the effect of incorrect pseudo-labels is minimized (Xie et al., 2019). It is observed that after this kind of self-training, the student is often able to outperform the teacher in fields like conditional text generation (He et al., 2020), image classification (Xie et al., 2019) and unsupervised machine translation (Lample et al., 2018).

1.2 Natural Language Processing (NLP)

NLP deals with applying computational techniques to the analysis and synthesis of natural language. Due to the wide availability of textual data on the web and increasing

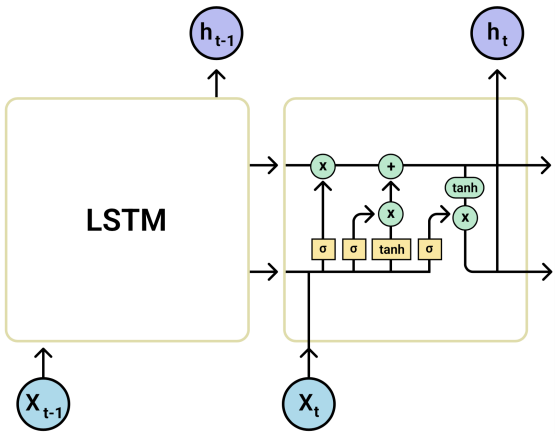


FIGURE 1.1 – Figure depicting a kind of RNN architecture called LSTM which has gating mechanisms for effective information propagation.

computation power, deep learning approaches have been able to significantly advance state of the art on a lot of NLP tasks (Bengio et al., 2003; Collobert et al., 2011; Sutskever et al., 2014; Bahdanau et al., 2015). These include learning rich word and sentence representations, language modeling, machine translation and question answering etc.

1.2.1 Recurrent Neural Networks

NLP often involves dealing with sequential data like sentences. Ordinary feedforward neural networks are meant for independent data points and can't handle sequences. To handle sequential data $\{\boldsymbol{x}_t\}_1^T$, we can use architectures like Recurrent Neural Networks (RNNs) or transformers Vaswani et al. (2017). In this thesis, we restrict ourselves to using just RNNs. For each element in the sequence $\boldsymbol{x}_t$, we compute a hidden state $\boldsymbol{h}_t$ processing the input $\boldsymbol{x}_t$ and the previous hidden state $\boldsymbol{h}_{t-1}$. This hidden state could be further processed to produce output $\boldsymbol{y}_t$. For RNN parameters $\boldsymbol{\theta}_{RNN} = \{\boldsymbol{w}_x, \boldsymbol{w}_h, \boldsymbol{w}_y, \boldsymbol{b}_h, \boldsymbol{b}_y\}$:

$$\boldsymbol{h}_t = f\left(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}; \boldsymbol{\theta}_{RNN}\right) = f\left(\boldsymbol{w}_x \boldsymbol{x}_t + \boldsymbol{w}_h \boldsymbol{h}_{t-1} + \boldsymbol{b}_h\right) \tag{1.8}$$

$$\boldsymbol{y}_t = f\left(\boldsymbol{w}_y \boldsymbol{h}_t + \boldsymbol{b}_y\right) \tag{1.9}$$

Vanilla RNNs, however, are limited by their inability to process long sequences. They suffer from the problem of vanishing gradients which inhibits the learning of long sequences. The backpropagated gradients get diminishingly smaller as we move fur-

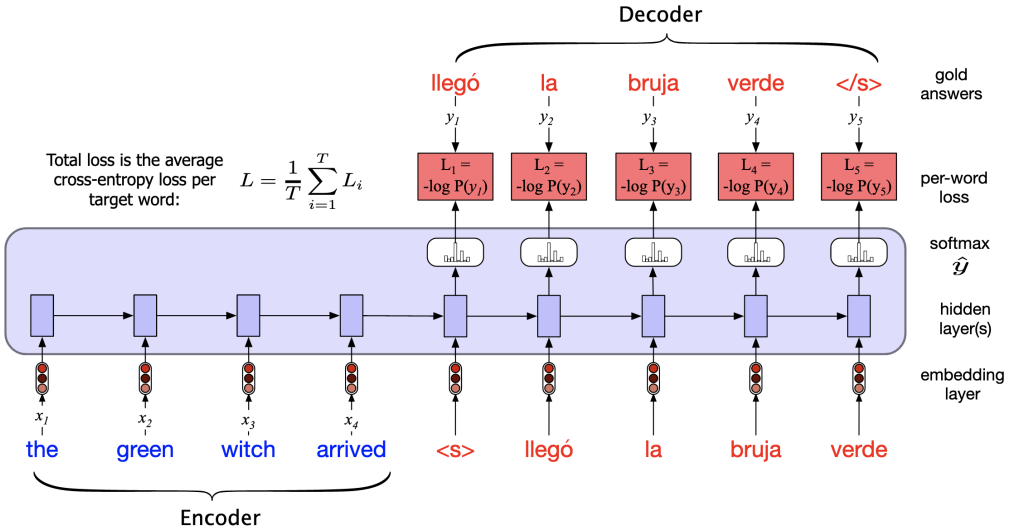


FIGURE 1.2 – Figure from Jurafsky and Martin depicting the training process of a seq2seq machine translation model using teacher forcing. Both the encoder and decoder here are modelled using RNNs.

ther back into the sequence. The diminished gradients hamper the learning process. To capture longer dependencies in sequences, architectures like LSTMs (Hochreiter and Schmidhuber, 1997) are often used as shown in Fig. 1.1. These architectures make use of gating mechanisms that selectively retain and pass information. LSTMs help in efficient backpropagation by alleviating the problem of vanishing gradients (Hochreiter and Schmidhuber, 1997).

1.2.2 Sequence to Sequence Learning

Sequence to Sequence (seq2seq) learning (Sutskever et al., 2014) involves using an encoder-decoder architecture that takes a sequence as an input and also generates a sequence as an output. Numerous NLP tasks can be cast into the seq2seq framework. The input sentence is tokenized and embedded in some word-embedding and encoded by the encoder. The decoder then uses this encoding to generate the desired output sentence. These models are trained using teacher-forcing (Sutskever et al., 2014) and often with attention mechanisms (Bahdanau et al., 2015). The two most common encoder-decoder architectures are RNNs and Transformers. The use cases for these seq2seq models are numerous, including: machine translation, summarization, question answering, etc. Our work involves agents which have to be trained to do machine translation. In tasks with limited amounts of data like ours, RNNs are popular. We, therefore, use seq2seq RNN

models with LSTM architectures (Hochreiter and Schmidhuber, 1997) and attention mechanism similar to (Bahdanau et al., 2015). However, in tasks with large amounts of data, transformer models are ubiquitous. For example, (Vaswani et al., 2017) achieved state-of-the-art results on WMT2014 translation datasets.

1.2.3 Language Modeling

Language modeling is one of the important tasks in natural language processing. It helps in tasks like next word prediction, language generation tasks like machine translation and summarization, etc. At its core, language modeling is about modelling the distribution of a sequence of words $p(x_1, \cdots, x_T)$. Often these models are trained by factorizing the sequence distribution as:

$$p(x_1, \cdots x_T|\boldsymbol{\theta}) = \prod_{t=1}^{T} p(x_t|x_1, \cdots, x_{t-1}, \boldsymbol{\theta}) \tag{1.10}$$

RNNs discussed above or transformer models (Vaswani et al., 2017) like GPT (Radford et al., 2019) can easily model this distribution while capturing long-range dependencies.

These models can then be trained on massive textual corpora using NLL loss. The models trained using language modeling are able to extract rich textual representations and learn meanings of input sentences due to the massive data they are trained on.

Numerous NLP tasks are deficient in data, and naively training on just task-specific data does not lead to good performance. A simple yet effective approach for these settings is **transfer learning**. In this technique, we first pre-train a model on some data-rich task like language-modeling, which can be used to learn meaningful textual representation. Then this model is finetuned on the target task, requiring much less supervised data. Recently, the BERT model (Devlin et al., 2018) has seen tremendous success by pretraining a transformer encoder via masked language modeling and next sentence prediction (Devlin et al., 2018). A portion of the input sentence is masked out at random in masked language modeling, and the model is asked to predict the masked parts. Next sentence prediction involves predicting whether given sentence succeeds the previous. Finetuning these large language models (Adiwardana et al., 2020; Radford et al., 2019) has resulted in state of the art on the GLUE benchmark (Wang et al., 2018), which consists of 11

natural language understanding tasks.

1.3 TASK-ORIENTED DIALOGUE

Designing conversational agents to interact with humans through language has been an active area of research since the early days of NLP (Winograd, 1971). The objective of task-oriented or goal-oriented dialogue systems is to develop automated systems that can assist human users in achieving their desired goals and tasks by communicating with them using natural language. A classic example is the airline ticket booking system, where the user's objective would be to book tickets to their desired destination, and they need to communicate with the automated system to accomplish that. Other examples include movie ticket booking, restaurant reservations, etc. In the past, efforts to build these systems involved creating a rule-based pipeline system that keeps track of user states and intents during the dialogue and has an internal logic to guide it to generating next actions (Lemon and Pietquin, 2007; Williams et al., 2014; Young et al., 2013). In general, these pipeline approaches have following components:

— **Dialogue State Tracking** involves updating the state based on the conversation history and the current user input using slot-value pairs. For the ticket booking case, for example, it would extract the domain (air-travel), intent (flight booking), and slots (source, destination, date, time).

— **Dialogue Policy Learning** is then used to decide upon the next action to take. This policy could be rule-based or learned.

— **Natural Language generation** is used after this to produce natural language output using some fixed template or some learned generative NLP system.

Modular structure makes these pipeline systems very stable, efficient and interpretable. A lot of real world systems are therefore built this way. However these require training each component on large scale dialogue data. These methods also involve a complex design with many individual interacting components. Hence, performance improvements in these individual components doesn't guarantee improvements in the whole systems. On the contrary, End-to-end methods use a simpler design using seq2seq models which are end-to-end differentiable and optimized using gradient descent.

1.3.1 End-to-end Approaches

Inspired by the success of Deep Learning approaches in NLP, especially of supervised learning, dialogue, and conversational AI, researchers have tried to curate big dialogue datasets like ATIS, Airdialogue, MultiWoz (Tur et al., 2010; Wei et al., 2018a; Budzianowski et al., 2018). These dialogue datasets contain the conversations between user and system in different settings like air-ticket booking and consists of user utterances and system responses. These datasets can then be used to train an end-to-end deep neural network agent using supervised learning. The model is trained to optimize the prediction probability of responses in the curated datasets. The input to the agent would be $\boldsymbol{x}^i = \{dialogue\_history, user\_utterance\}$ with target output $\boldsymbol{y}^i = \{system\_response\}$. In the end, the model also needs to take the desired action to solve the task. The curated conversational datasets can be easily set up to train this agent. The agents thus trained can learn to hold a grammatically sound conversation and take actions by learning the statistical properties of language like syntax and some semantic components over the large dataset.

On the downside, often, the utterances are not correctly grounded towards the task and unrelated to the conversation at hand (Strub et al., 2017; Lewis et al., 2017; Bordes et al., 2017). In short, these methods are limiting in terms of their ability to achieve the final task and show inconsistent behavior. Another problem is the lack of human-labeled conversational data. The scenarios where these agents have to be deployed are often different from the scenarios on which the dataset was collected, leading to a domain mismatch. Purely supervised end-to-end models fail to generalize to the test domains. Pretraining then finetuning approach is a simple solution to this problem that the community has widely adopted.

1.3.2 Pretraining then finetuning approach

A natural follow-up to improve the robustness of the agent and make it better at completing the task could consist of training the agent to solve the actual language task, rather than solely training it to generate grammatically correct sentences using supervised learning. Ideally, such training would incorporate pairing the conversational agent with a human and then letting them interact to help the conversational agent learn to solve the task (Skantze and Hjalmarsson, 2010; Li et al., 2016a). However, this kind of human interaction quickly faces scalability and reproducibility issues. As a consequence,

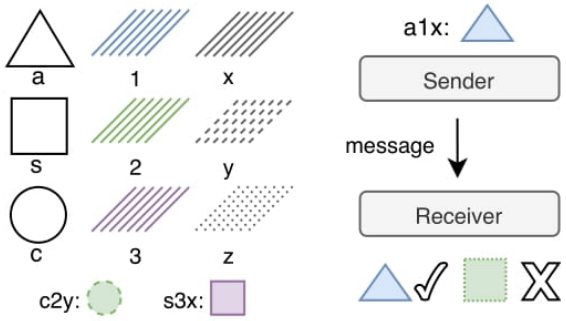


FIGURE 1.3 – Lewis game. Given the input object, the sender emits a compositional message that is parsed by the receiver to retrieve object properties. In the language drift setting, both models are trained towards an identity map while solving the reconstruction task.

agents are often trained by interacting with a second model to simulate the task-oriented scenarios (Levin et al., 2000; Schatzmann et al., 2006; Lemon and Pietquin, 2012). The first model works as a sender and is meant to simulate the human. The second model is the receiver which will work as our conversational agent upon training. In recent literature, a common setting is to pretrain these two neural models with supervised learning to acquire the language structure. They are then paired to interact and learn to solve the task, a process we call finetuning. At least one of the models is trained to maximize task completion using either reinforcement learning (Williams, 1992) or Gumbel-softmax straight-through estimator (Jang et al., 2017a; Maddison et al., 2017). In this thesis, we restrict ourselves to using Gumbel softmax, which is described in detail in section 1.3.4. This finetuning step has shown consistent improvement in dialogue games (Li et al., 2016b; Strub et al., 2017; Das et al., 2017), referential games (Havrylov and Titov, 2017; Yu et al., 2017) or instruction following (Fried et al., 2018). Note that in the end, we will just use the receiver agent, which serves as a conversational agent capable of communicating with a human and taking the desired action on their behalf, like booking flight tickets.

1.3.3 Sender-Receiver Games

These games are widely studied in the dialogue and emergent communication literature. These are fully cooperative two-player language games that involve the first player acting as the *sender* and the second player acting as the *receiver*. The sender is given some information or context with a given objective or task in hand. The sender must

then communicate its knowledge using a shared vocabulary to the receiver. The receiver then needs to process that information and take specific actions to solve the arbitrary task given. The game can involve many rounds of feedback messages between the two agents to model multi-turn dialogue. Alternatively, it can be a single turn where the sender outputs a single utterance. In this thesis, our objective is to understand the linguistic properties of the communication between the sender and the receiver. Therefore, we focus only on the single-turn scenario as it eases the language analysis and avoids the complication of multi-turn dialogue. Nevertheless, our analysis and methods proposed are relatively general and may be generalized to multi-turn scenarios.

Formally, a single-turn S/R game is defined as a 5-tuple $\mathcal{G} = (\mathcal{O}, \mathcal{M}, \mathcal{A}, R, \mathcal{V})$. At the beginning of each episode, an observation (or scenario) $\boldsymbol{o} \in \mathcal{O}$ is sampled. Then, the sender s emits a message $\boldsymbol{m} = s(\boldsymbol{o}) \in \mathcal{M}$, where the message can be a sequence of words ie $\boldsymbol{m} = [w_t]_{t=1}^{T}$ from a vocabulary $\mathcal{V}$, i.e. $w_t \in \mathcal{V}$. The receiver r gets the message and performs an action or gives an output $\boldsymbol{a} = r(\boldsymbol{m}) \in \mathcal{A}$. Finally, both agents receive the same reward $R(\boldsymbol{o}, \boldsymbol{a})$ which they aim to maximize. We study two S/R games in this thesis namely the Lewis game (Figure 1.3) and the Translation game (Figure 1.5).

— Lewis game (Kottur et al., 2017) is a toy S/R game that we use to build intuition. This game is a good testbed for understanding the communication structure. However, being a simple referential game, it lacks the richness of natural language. Figure 1.3 is an illustration depicting this game. In this game, the sender will first observe an object $\boldsymbol{o}$ with p properties and each property has t possible values ie $\boldsymbol{o}[i] \in [1 \dots t]$ for $i \in [1 \dots p]$. The communication language would involve a vocabulary $\mathcal{V}$ of size $p \times t$, equal to the number of total property values. The sender would then send a message m consisting of p words, one for each property of the input object. The receiver is then tasked with reconstructing the object $\boldsymbol{o}$.

— The Translation game was first used by (Lee et al., 2019) and involves translation from French(FR) to German (DE) using English (EN) as a pivot language. The sender and receiver will be translation agents here, modeled by seq2seq models like an LSTM. Sender will be given a French sentence $\boldsymbol{o} = Fr$ and it'll output an English sentence $\boldsymbol{m} = En$. The receiver receives this En sentence and translates this into output German sentence $\boldsymbol{a} = De$. Both the agents are then rewarded according to the translation quality. Although it is a contrived setup, the communication is in natural language, and hence it serves our purpose of evaluating in a

setting that captures the richness and complexity of natural language.

1.3.4 Pretraining then finetuning in single turn S/R games

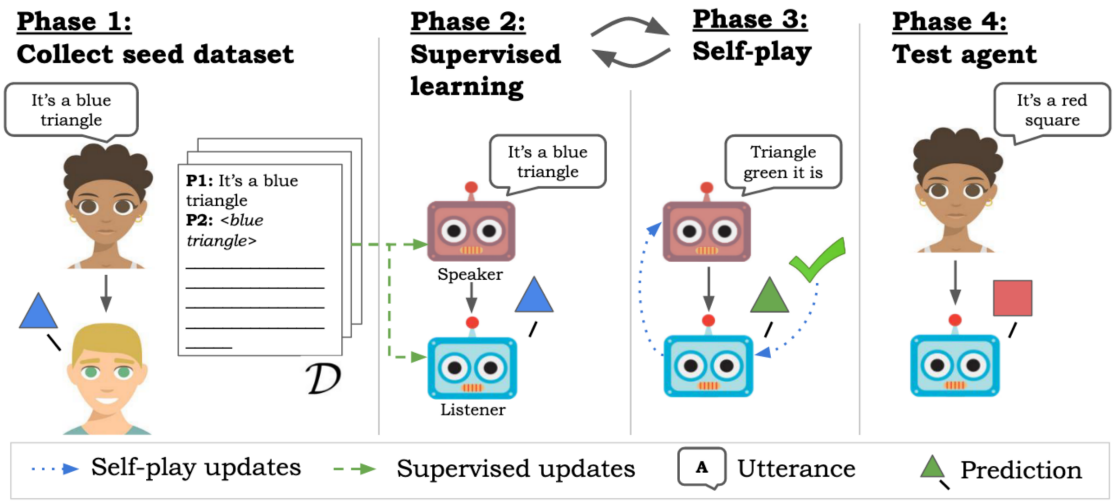


FIGURE 1.4 – A graphic illustration from (Gupta et al., 2019) explaining all the four phases of the pretrain then finetune approach.

It usually involves 4 phases which are succinctly described in Figure 1.4.

The first phase is just collecting the pre-train data $\mathcal{D}_{pre} = \left\{\boldsymbol{o}^n_{pre}, \boldsymbol{m}^n_{pre}, \boldsymbol{a}^n_{pre}\right\}_1^N$. In realistic scenarios it'll involve using human conversational data. But for Lewis game this can be automatically generated and for translation game we can use a triply aligned translation dataset like IWSLT (Cettolo et al., 2012).

The second phase, called pretraining, is essentially just supervised learning by computing the loss $L_{supervised}$ on the dataset $\mathcal{D}_{pre}$.

The third phase is called finetuning or training via **self-play** which involves pairing both the sender and receiver agents together. For this we need a finetuning dataset $\mathcal{D}_{train} = \{\boldsymbol{o}^n, \boldsymbol{a}^n\}_1^N$. This dataset helps us establish the simulator for the dialogue game. Note that this doesn't include the intermediate messages $\boldsymbol{m}$ which would correspond to human conversations in real scenarios. The agents are then made to play the dialogue game on unseen input objects i.e. $\boldsymbol{o}^n_{train} \notin \left\{\boldsymbol{o}^n_{pre}\right\}_1^N$. These objects can be from a domain which need not be same as the pre-training domain. The sender s parametrized by θ and the receiver r is parametrized by ϕ. The message output by the sender is $\boldsymbol{m} \sim s(\boldsymbol{m}|\boldsymbol{o}, \theta)$ and it consists of words $\boldsymbol{m} = [w_t]_{t=1}^T$. The receiver gets this message and generates

the ouput action distribution $r(\boldsymbol{a}|\boldsymbol{m}, \phi)$. We now need to maximize the probability of the target action $\boldsymbol{a}^n$ under this distribution similar to cross entropy training. Our task completion function to be optimized is:

$$\mathbf{R}_{total}(\theta, \phi) = \sum_n \mathbf{R}^n(\theta, \phi) \tag{1.11}$$

$$= \sum_n r\left(\boldsymbol{a}^n|\boldsymbol{m}, \phi\right) \tag{1.12}$$

This reward needs to be maximized with respect to both the sender and the receiver using stochastic gradient descent and backpropagation. Computing the gradient $\frac{\partial \mathbf{R}}{\partial \boldsymbol{\phi}}$ is straightforward and essentially translates to standard supervised training using cross-entropy loss. However, computing $\frac{\partial \mathbf{R}}{\partial \boldsymbol{\theta}}$ is not straightforward because it involves a non-differentiable step of sampling the message $\boldsymbol{m}$. There are two standard techniques to compute this gradient, the first being policy gradient as used in reinforcement learning (Sutton et al., 2000) and the second being gumbel-softmax reparametrization trick (Jang et al., 2017a,b). In this thesis, we only use the gumbel-softmax approach.

Gumbel Straight-Through Estimator To estimate the task loss gradient with respect to the sender s parameters θ, the receiver gradient can be further backpropagated using the Gumbel softmax straight-through estimator (GSTE) (Jang et al., 2017a; Maddison et al., 2017). Hence, the sender parameters are directly optimized toward task loss. Given a sequential message $\boldsymbol{m} = [w_t]_{t=1}^T$, we define $\boldsymbol{y}_t$ as follows:

$$\boldsymbol{y}_t = \text{softmax}\big((\log s(w|\boldsymbol{o}, w_{t-1}, \cdots, w_0; \theta) + g_t)/\tau\big) \tag{1.13}$$

where $s(w|\boldsymbol{o}, w_{t-1}, \cdots, w_0)$ is the categorical probability of next word given the sender observation o and previously generated words, $g_t \sim \text{Gumbel}(0, 1)$ and τ is the Gumbel temperature that levels exploration. When not stated otherwise, we set $\tau = 1$. Finally, we sample the next word by taking $w_t = \arg\max \boldsymbol{y}_t$ before using the straight-through gradient estimator to approximate the sender gradient:

$$\frac{\partial \mathbf{R}}{\partial \theta} = \frac{\partial \mathbf{R}}{\partial w_t}\frac{\partial w_t}{\partial y_t}\frac{\partial y_t}{\partial \theta} \approx \frac{\partial \mathbf{R}}{\partial w_t}\frac{\partial y_t}{\partial \theta}. \tag{1.14}$$

The fourth and last phase just involves testing on some held-out test set $\mathcal{D}_{ho} = \{\boldsymbol{o}_{ho}^n, \boldsymbol{m}_{ho}^n, \boldsymbol{a}_{ho}^n\}_1^N$.

1.3.5 Language Drift

The hope with finetuning by interaction is that the agents will bootstrap off of the language understanding gained in the pretraining step and then use the finetuning step to learn how to solve the task more robustly and in newer contexts. This leads to a massive jump in the task completion score, which is our objective with self-play. However, note that our end goal is to develop a conversational agent which can understand a human. To that end, we wish to pair a human with the receiver agent to help the human accomplish some tasks in the real world, like ticket booking. So even though we are explicitly trying to maximize the task completion score in self-play, the receiver agents must be able to understand human language. In the single-turn S/R games, this means that we want the message $\boldsymbol{m}$ distribution to remain close to the pretraining distribution as captured by the dataset $\left\{\boldsymbol{m}_{pre}^{n}\right\}_{1}^{N}$ which the agents were initialized to do. However, it is observed that the gains in task-completion score come at the cost of deterioration in the communication language $\boldsymbol{m}$ between the sender and receiver. The sender and receiver were initialized on pretrained data (human language data in real scenarios) and learned to communicate using that. However, in the finetuning phase, the agents are optimized with a language-agnostic task reward, and the agents might co-adapt to each other and create their own communication protocol with a singular focus on solving the task. The agents have no incentive to maintain the language they learned during pretraining. As a result, the final communication language learned by the models might not look like pretrained language anymore, but the agents can still complete the task. This behavior is not desirable, and we call this phenomenon during finetuning as **language drift**. You can have a high task score but very poor language. As a result, the final agents perform poorly when paired with humans. There can be different types of language drift (Lazaridou et al., 2020) including

— Syntactic or structural drift which involves removing grammar redundancy (e.g. "is it a cat ?" can become "is cat ?" (Strub et al., 2017))

— The drift can be a semantic one, where words have a different meaning than intended. (e.g. "an old teaching" means "an old man" (Lee et al., 2019))

— The drift can also be at pragmatic/functional level with unexpected actions or intentions after the language (e.g., after agreeing on a deal, the agent performs another trade (Li et al., 2016b))

Thus, these agents perform poorly when paired with humans (Chattopadhyay et al.,

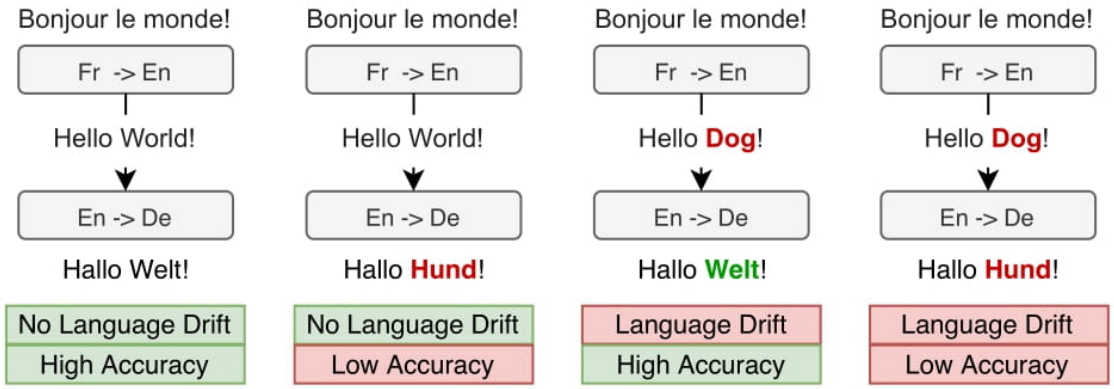


FIGURE 1.5 – In the translation game, the sentence is translated into English then into German. The second and fourth cases are regular failures, while the third case reveals a form of agent co-adaptation.

2017; Zhu et al., 2017; Lazaridou et al., 2020). Figure 1.5 gives an example of the language drift problem in the translation game.

1.3.6 Evaluation metrics used for Language Drift

For the Lewis game, detecting drift is as simple as verifying whether the correct object is communicated or not. However, for the translation game, we use metrics like BLUE and NLL. More details are given in Chapter 2.

BLEU (bilingual evaluation understudy) is a metric used in NLP to evaluate the quality of machine translations from a given language by comparing with a given list of reference translations. It is a precision-based metric and computes similarity using n-gram overlap. The score ranges from 0 to 1.0, with 1.0 being a perfect translation. It correlates very well with human judgment, and its use is ubiquitous in the translation literature (Papineni et al., 2002a).

Negative Log-Likelihood (NLL) is a standard loss function used in machine learning. As described in 1.2.3, language models with parameters $\boldsymbol{\theta}$ try to model the distribution $p(x_1, \cdots x_T|\boldsymbol{\theta}) = \prod_{t=1}^{T} p(x_t|x_1, \cdots, x_{t-1}, \boldsymbol{\theta})$. They are trained by minimizing NLL loss on some big text corpus $\mathcal{D}$ using stochastic gradient descent.

$$NLL = -\log P(\mathcal{D}|\boldsymbol{\theta})$$

After training, however, NLL computed on some given sentences using parameters θ of this trained language model can be used as a metric to judge the quality of those sentences. The lower this NLL score, the more is the likelihood of the given text being from the text distribution which the language model tries to model.

1.4 Countering Language Drift

The recent literature on countering language drift includes a few distinct groups of methods. The first group requires an external labeled dataset that can be used for visual grounding (i.e., aligning language with visual cues (Lee et al., 2019)), reward shaping (i.e., incorporating a language metric in the task success score (Li et al., 2016b)) or KL minimization (Havrylov and Titov, 2017). However, these methods depend on the existence of an extra supervision signal and ad-hoc reward engineering, making them less suitable for general tasks. The second group is of the population-based methods, which enforce social grounding through a population of agents, preventing them from straying away from the common language (Agarwal et al., 2019).

The third group of methods involves an alternation between the selfplay training phase and the supervised training phase on pretraining dataset (Wei et al., 2018b; Lazaridou et al., 2016). This is essentially multi-task learning. This approach has been formalized in Gupta et al. (2019) as *Supervised-2-selfPlay* (S2P). Empirically, the S2P approach has shown impressive resistance to language drift and is relatively task-agnostic. We use S2P as the main baseline, and it is realized by using a weighted sum of the losses of supervised training on pretraining dataset $L_{supervised}$ and self-play loss L_{Gumbel} with the parameter α to module between the two (default value is 1). Another common approach is to alternate training between the two losses.

$$L_{Gumbel} = -\mathbf{R}_{total}(\theta, \phi) \tag{1.15}$$

$$L_{S2P} = L_{Gumbel} + \alpha L_{supervised} \tag{1.16}$$

However, the success of S2P is highly dependent on the quality of the fixed pre-training dataset, which in practice may be noisy, small, and only tangentially related to the task.

Lifelong Learning The problem of language drift is closely related to the problem of catastrophic forgetting (McCloskey and Cohen, 1989) found in neural networks. It is often observed that when a neural network trained on Task 1 is finetuned on Task 2, the network forgets what it learned on Task 1. It can be argued that the language drift problem can also be viewed as an instantiation of lifelong learning since the agent needs to keep the knowledge about language while acquiring new knowledge on using language to solve the task. From this perspective, S2P can be viewed as a method of task rehearsal strategy (Silver and Mercer, 2002) for lifelong learning. The success of iterated learning for language drift could motivate the development of similar methods in countering catastrophic forgetting.

1.5 Language and Iterated Learning

Humans are the only animals to communicate using natural language. Some researchers attribute language to the higher-level cognition in humans as it allows them to form abstract concepts and assign a symbol to those which can be further built upon recursively. Moreover, language also helps accomplish social and cultural transmission of information acquired by the older generations to the new generations. Thereby language helps expand the information flow between generations from being purely genetic also to include cultural experience. This has helped humans build on top of the knowledge and experience of older generations and catapult society to where it is now.

Language emergence has been a long-standing problem in the field of psychology and cognitive science (Clark and Clark, 1977; Ellis and Larsen-Freeman, 2006). It deals with understanding how and why languages emerged in humans. Animals have some form of communication, but none can be classified as a formal language.

Language acquisition by human children is another perplexing question closely related to emergence (Pinker, 1995; De Villiers et al., 1978). Chomsky talks about the poverty of stimulus (Chomsky, 2010), which questions how a human child is able to acquire the language from the set of select few examples provided by its parents. Furthermore, all human children in a region are able to converge to the same language despite being given different sets of examples to learn from. Chomsky argues from a nativist perspective and

hypothesizes the existence of universal grammar or inductive biases in human brains, which help all the children to converge to the same language despite this poverty of stimulus. This topic is still an ongoing research field with lots of discussion and is related to the classic nativism vs. empiricism debate (Spelke, 1998).

Language evolution is the process of dynamic change of language over time. Language evolves over time due to transmission within a large community and transmission through generations, involving the parents teaching the language to their children. The question of interest then is how this constant language transmission impacts the language's structure (Christiansen and Kirby, 2003).

Iterated learning is a framework in the cognitive science literature that tries to tackle the fundamental problems of language emergence, acquisition, evolution, and persistence of linguistic structure. Kirby first proposed the iterated learning hypothesis by arguing about the need for human languages to be both compressible and expressive for them to be effectively transmitted through human generations (Kirby, 2001, 2002). Expressiveness and generalizability to unseen situations are necessary for the language to be functional. And compressibility is required for the children to pick it up from their parents. Kirby argued that these evolutionary pressures and bottlenecks favor highly structured languages with properties like compositionality. This helps the language to be easily learned by the offspring and also to generalize to variations in the environment. So in effect, the iterated transmission of language is guided by these learning bottlenecks, and the language evolves to be highly structured and compositional. In particular, it was shown by (Kirby et al., 2014) that unstructured proto-language consistently converges to some form of compositional language in both human experiments and mathematical modeling. This phenomenon can be easily illustrated using a quick human experiment done by (Kirby and Griffiths, 2014) as described in Figure 1.6. This experiment tries to simulate cultural transmission by transmitting some symbols across generations. At each generation, human participants were asked to learn about the language only from a subset of the objects. Then the participants were asked to produce the language for the whole set of objects, which will serve as the dataset for the following participants. The language at generation 0 is randomly generated. (Kirby and Griffiths, 2014) find that by the time of generation 10, the language converges to a somewhat compositional structure.

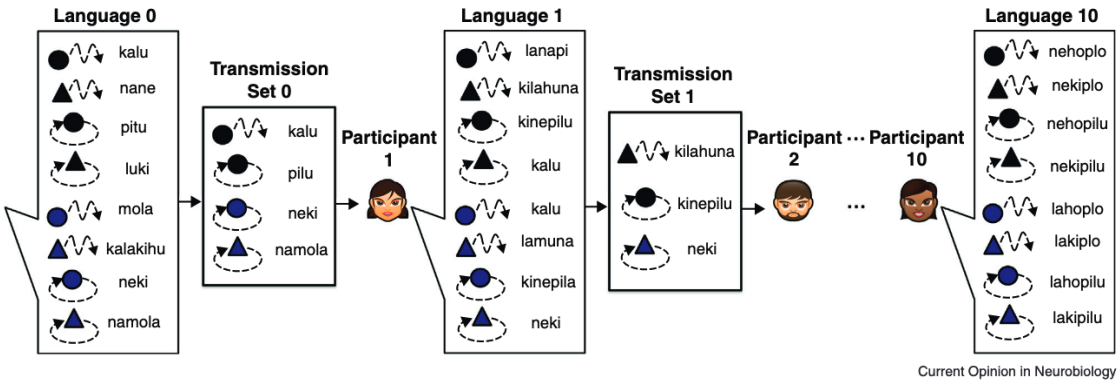


FIGURE 1.6 – An illustration from (Kirby and Griffiths, 2014) depicting the iterated learning phenomenon in a quick human experiment.

Inspired by these observations, there has been much work to apply iterated learning in many application domains of NLP using deep neural networks (Guo et al., 2019; Li and Bowling, 2019; Ren et al., 2020; Cogswell et al., 2019; Dagan et al., 2020). These applications involve training a teacher and student network modeled as deep neural networks iteratively. The teacher models are trained first to solve some tasks involving languages like emergent communication (Guo et al., 2019; Cogswell et al., 2019; Ren et al., 2020) or visual question answering (Vani et al., 2021). After that, there is a step of self-distillation to a student model where the student model learns to mimic teacher outputs. This process is repeated till convergence. These applications show that the inductive learning bottleneck during the imitation learning phase encourages compositionality in the emerged language. In the following chapters we seek to use iterated learning to *preserve* the structure of an existing language rather than use it for *emergence* of a new structured language.

2 Countering Language Drift with Seeded Iterated Learning

Authors: Yuchen Lu (YL), **Soumye Singhal** (SS), Florian Strub(FS), Olivier Pietquin(OP), and Aaron Courville (AC).

Contributions: AC proposed the research problem and the collaboration. YL and SS led the project with FS, OP and AC serving as advisors. YL found the relevant literature on iterated learning and proposed to use it to counter language drift, which motivated our method. YL, SS, FS, OP & AC concretised the idea over many iterations in the daily meetings. YL and SS jointly managed the project schedule, led all the meetings, conducted the experiments and presented them at the meetings. YL and SS did the early extensive experiments on the lewis game to make the method work. The final results and experiments in the paper for the lewis game were done by SS. YL did the final Translation game experiments and wrote all the code for it. YL, SS and FS contributed to the first draft of the paper.

FS and OS advised on the domain knowledge about task-oriented Dialogue.

AC proposed the research problem, supervised the project, helped define the scope and gave clarity on the research direction. AC and OP gave feedback and edited the paper. Everyone attended and contributed to the meetings.

Affiliation

— Yuchen Lu, Mila, Université de Montréal

— Soumye Singhal, Mila, Université de Montréal

— Florian Strub, Deepmind

— Olivier Pietquin, Google Brain

— Aaron Courville, Mila, Université of Montréal

2.1 INTRODUCTION

Recently, neural language modeling methods have achieved a high level of performance on standard natural language processing tasks (Adiwardana et al., 2020; Radford et al., 2019). Those agents are trained to capture the statistical properties of language by applying supervised learning techniques over large datasets (Bengio et al., 2003; Collobert et al., 2011). While such approaches correctly capture the syntax and semantic components of language, they give rise to inconsistent behaviors in goal-oriented language settings, such as question answering and other dialogue-based tasks (Gao et al., 2019). Conversational agents trained via traditional supervised methods tend to output uninformative utterances such as, for example, recommend generic locations while booking for a restaurant (Bordes et al., 2017). As models are optimized towards generating grammatically-valid sentences, they fail to correctly ground utterances to task goals (Strub et al., 2017; Lewis et al., 2017).

To solve this the approach of pretraining the agents and then finetuning in a simulator using interactive learning is used. This approach was described in detail in introduction 1.3.2 and has achieved great results. Unfortunately, interactive learning gives rise to the *language drift* phenomenon. As the agents are solely optimizing for task completion, they have no incentive to preserve the initial language structure. They start drifting away from the pretrained language output by shaping a task-specific communication protocol. We thus observe a co-adaptation and overspecialization of the agent toward the task, resulting in significant changes to the agent's language distribution. In practice, there are different forms of language drift which were discussed in 1.3.5

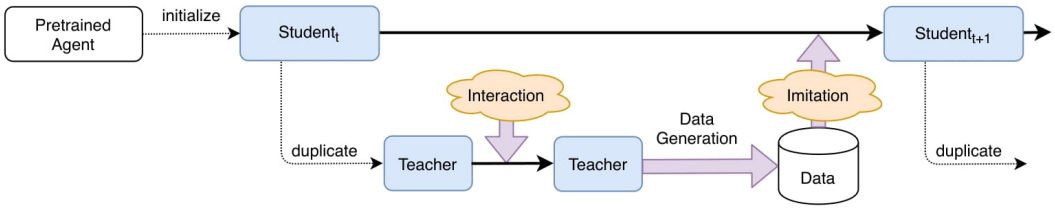


FIGURE 2.1 – Sketch of Seeded Iterated Learning. A **student** agent is iteratively refined using newly generated data from a **teacher** agent. At each iteration, a teacher agent is created on top of the student before being finetuned by interaction, e.g. maximizing a task completion-score. The teacher then generates a dataset with greedy sampling, which is then used to refine the student through supervised learning. Note that the interaction step involves interaction with another language agent.

In this paper, we introduce the *Seeded Iterated Learning* (SIL) protocol to counter language drift. This process is directly inspired by the iterated learning procedure to

model the emergence and evolution of language structure (Kirby, 2001; Kirby et al., 2014). SIL does not require human knowledge intervention, it is task-agnostic, and it preserves natural language properties while improving task objectives.

As illustrated in Figure 2.1, SIL starts from a pretrained agent that instantiates a first generation of *student* agent. The teacher agent starts as a duplicate of the student agent and then goes through a short period of interactive training. Then the teacher generates a training dataset by performing the task over multiple scenarios. Finally, the student is finetuned – via supervised learning – to imitate the teacher data, producing the student for next generation, and this process repeats. As further detailed in Section 2.2, the imitation learning step induces a bias toward preserving the well-structured language, while discarding the emergence of specialized and inconsistent language structure (Kirby, 2001). Finally, SIL successfully interleaves interactive and supervised learning agents to improves task completions while preserving language properties.

Our contribution In this work, we propose Seeded Iterated Learning and empirically demonstrate its effectiveness in countering language drift. More precisely,

1. We study core Seeded Iterated Learning properties on the one-turn Sender-Receiver version of the Lewis Game.
2. We demonstrate the practical viability of Seeded Iterated Learning on the French-German translation game that was specifically designed to assess natural language drift (Lee et al., 2019). We observe that our method preserves both the semantic and syntactic structure of language, successfully countering language drift while outperforming strong baseline methods.
3. We provide empirical evidence towards understanding the algorithm mechanisms [1].

2.2 METHOD

2.2.1 Learning Bottleneck in Iterated Learning

The core component of iterated learning is the existence of the *learning bottleneck* (Kirby, 2001): a newly initialized student only acquires the language from a *limited number of examples* generated by the teacher. This bottleneck implicitly favors

1. Code for Lewis game and translation game

any structural property of the language that can be exploited by the learner to generalize, such as compositionality.

Yet, Kirby (2001) assumes that the student to be a perfect inductive learner that can achieve systematic generalization (Bahdanau et al., 2019). Neural networks are still far from achieving such goal. Instead of using a limited amount of data as suggested, we propose to use a regularization technique, like *limiting the number of imitation steps*, to reduce the ability of the student network to memorize the teacher's data, effectively simulating the learning bottleneck.

2.2.2 Seeded Iterated Learning

As previously mentioned, Seeded Iterated Learning (SIL) is an extension of Iterated Learning that aims at preserving an initial language distribution while finetuning the agent to maximize task-score. SIL iteratively refines a pretrained agent, namely the *student*. The *teacher* agent is initially a duplicate of the student agent, and it undergoes an interactive training phase to maximize task score. Then the teacher generates a new training dataset by providing pseudo-labels, and the student performs imitation learning via supervised learning on this synthetic dataset. The final result of the imitation learning will be next student. We repeat the process until the task score converges. The full pipeline is illustrated in Figure 2.1. Methodologically, the key modification of SIL from the original iterated learning framework is the use of the student agent to seed the imitation learning rather than using a randomly initialized model or a pretrained model. Our motivation is to ensure a smooth transition during the imitation learning and to retain the task progress.

Although this paper focuses on countering language drift, we emphasize that SIL is task-agnostic and can be extended to other machine learning settings.

2.2.3 SIL for Sender-Receiver Framework

Sender-Receiver games are the experimental framework we use to study the impact of SIL on language drift. This framework was explained in details in introduction 1.3.3. Here we explain in detail the instantiation of SIL for this setting.

We consider two parametric models, the sender $s(.;\boldsymbol{\theta})$ and the receiver $r(.;\boldsymbol{\phi})$. Following the SIL pipeline, we use the uppercase script S and T to respectively denote the parameters of the student and teacher. For instance, $r(.;\boldsymbol{\phi}^T)$ refers to the teacher recei-

Algorithm 1 Seeded Iterate Learning for S/R Games

Require: Pretrained parameters of sender $\boldsymbol{\theta}$ and receiver $\boldsymbol{\phi}$.
Require: Training scenarios $\mathcal{O}_{train}$ {or scenario generator}
Copy $\boldsymbol{\theta}, \boldsymbol{\phi}$ to $\boldsymbol{\theta}^S, \boldsymbol{\phi}^S$ {*Prepare Iterated Learning*}
repeat
  Copy $\boldsymbol{\theta}^S, \boldsymbol{\phi}^S$ to $\boldsymbol{\theta}^T, \boldsymbol{\phi}^T$ {*Initialize Teacher*}
  for $i = 1$ **to** k_1 **do**
    Sample a batch $\boldsymbol{o} \in \mathcal{O}_{train}$
    Get $\boldsymbol{m} = s(\boldsymbol{o}; \boldsymbol{\theta}^T)$ and $\boldsymbol{a} = r(\boldsymbol{m}; \boldsymbol{\phi}^T)$ to have $R(\boldsymbol{o}, \boldsymbol{a})$
    Update $\boldsymbol{\theta}^T$ and $\boldsymbol{\phi}^T$ to maximize R
  end for {*Finish Interactive Learning*}
  for $i = 1$ **to** k_2 **do**
    Sample a batch of $\boldsymbol{o} \in \mathcal{O}_{train}$
    Sample $\boldsymbol{m} = s(\boldsymbol{o}; \boldsymbol{\theta}^T)$
    Update $\boldsymbol{\theta}^S$ with supervised learning on $(\boldsymbol{o}, \boldsymbol{m})$
  end for {*Finish Sender Imitation*}
  for $i = 1$ **to** k_2' **do**
    Sample a batch of $\boldsymbol{o} \in \mathcal{O}_{train}$
    Get $\boldsymbol{m} = s(\boldsymbol{o}; \boldsymbol{\theta}^S)$ and $\boldsymbol{a} = r(\boldsymbol{m}; \boldsymbol{\phi}^S)$ to have $R(\boldsymbol{o}, \boldsymbol{a})$
    Update $\boldsymbol{\phi}^S$ to maximize R
  end for {*Finish Receiver Finetuning*}
until Convergence or maximum steps reached

ver. We also assume that we have a set of scenarios $\mathcal{O}_{train}$ that are fixed or generated on the fly. We detail the SIL protocol for single-turn S/R games in Algorithm 1.

In one-turn S/R games, the language is only emitted by the sender while the receiver's role is to interpret the sender's message and use it to perform the remaining task. With this in mind, we train the sender through the SIL pipeline as defined in Section 2.2.2 (i.e., interaction, generation, imitation), while we train the receiver to quickly adapt to the new sender's language distribution with a goal of stabilizing training (Ren et al., 2020). First, we jointly train $s(.; \boldsymbol{\phi}^T)$ and $r(.; \boldsymbol{\phi}^T)$ during the SIL interactive learning phase. Second, the sender student imitates the labels generated by $s(.; \boldsymbol{\phi}^T)$ through greedy sampling. Third, the receiver student is trained by maximizing the task score $R(r(\boldsymbol{m}; \boldsymbol{\phi}^S), \boldsymbol{o})$ where $\boldsymbol{m} = s(\boldsymbol{o}; \boldsymbol{\theta}^S)$ and $\boldsymbol{o} \in \mathcal{O}_{train}$. In other words, we finetune the receiver with interactive learning while freezing the new sender parameters. SIL has three training hyperparameters: (i) k_1, the number of interactive learning steps that are performed to obtain the teacher agents, (ii) k_2, the number of sender imitation steps, (iii) k_2', the number of interactive steps that are performed to finetune the receiver with the

new sender. Unless stated otherwise, we define $k_2 = k'_2$.

SIL can be applied with RL methods when dealing with non-differential reward metrics (Lee et al., 2019), however RL has high gradient variance and we want to GSTE as a start. Since GSTE only optimizes for task completion, language drift will also appear.

2.3 Building Intuition: The Lewis Game

In this section, we explore a toy-referential game based on the Lewis Game (Lewis, 1969) to have a fine-grained analysis of language drift while exploring the impact of SIL.

2.3.1 Experimental Setting

We summarize the Lewis game instantiation described in Gupta et al. (2019) to study language drift, and we illustrate it in Figure 1.3. First, the sender observes an object o with p properties and each property has t possible values: $o[i] \in [1 \dots t]$ for $i \in [1 \dots p]$. The sender then sends a message m of length p from the vocabulary of size $p \times t$, equal to the number of property values. Our predefined language $\mathcal{L}$ uniquely map each property value to each word, and the message is defined as,

$$\mathcal{L}(o) = [o_1, t + o_2, ..., (p-1)t + o_p] \tag{2.1}$$

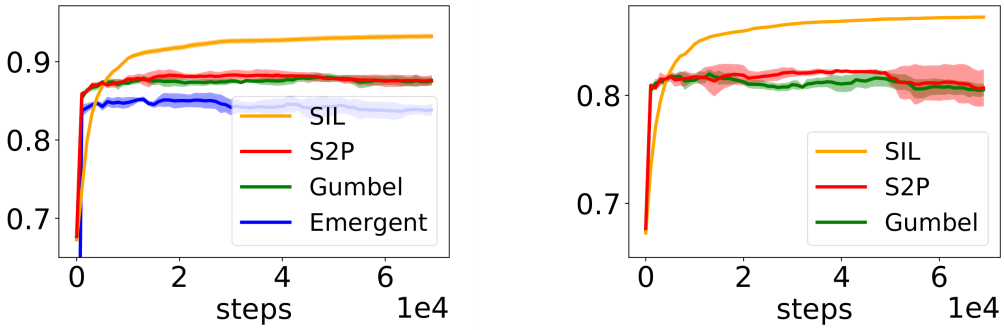
We study whether this language mapping is preserved during S/R training.

In our task, we use $p = t = 5$ with a total of 3125 unique objects. We split this set of objects into three parts:

— The first split (pre-train) is labeled with correct messages to pre-train the initial agents. This split only contains 10 combination of objects. As soon as we provide additional objects, the sender and receiver fully solve the game by using the target language, which is not suitable to study the language drift phenomenon.

— The second split is used for the interactive training scenarios. This interactive split contains 30 objects. This choice is arbitrary, and choosing a additional objects gives similar results.

— The third split is held out (HO) for final evaluation and contains the 3.1k remaining objects.



(a) Task Score

(b) Sender Language Score

FIGURE 2.2 – Task Score and Language Score for SIL ($\tau = 10$) vs baselines ($\tau = 1$). SIL clearly outperforms the baselines. For SIL: $k_1 = 1000, k_2 = k_2' = 400$. The emergent language score is close to zero. All results are averaged over four seeds.

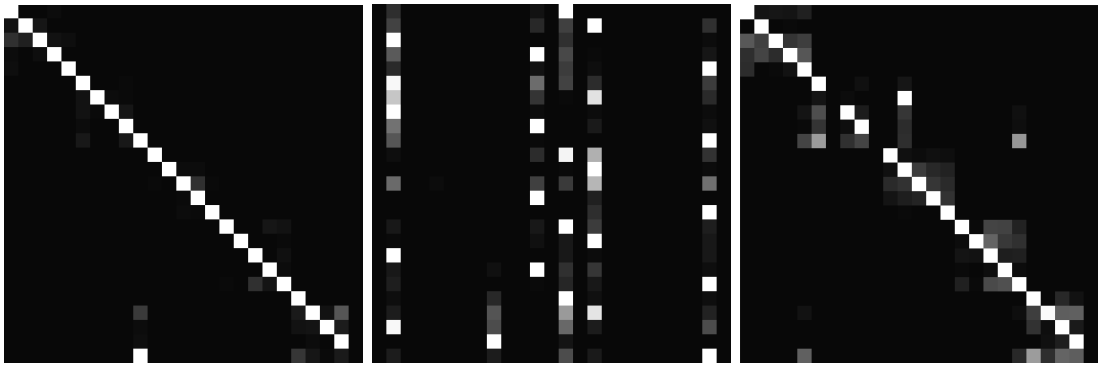
In the Lewis game, the sender and the receiver architecture are modeled by two-layer feed-forward networks ie MLPs with a hidden size of 200 and no-activation ($ReLU$ activations lead to similar scores). During interactive learning, we use a learning rate of 1e-4 for SIL. For the baselines, we used a learning rate of 1e-3 as it provides better performance on both the language and score tasks. In both cases, we use a training batch size of 100. Finally, for the teacher imitation phase, the student uses a learning rate of 1e-4.

We use two main metrics to monitor our training: *Sender Language Score* (LS) and *Task Score* (TS). For the sender language score, we enumerate the held-out objects and compare the generated messages with the ground-truth language on a per token basis. For task accuracy, we compare the reconstructed object vs. the ground-truth object for each property. Formally, we have:

$$LS = \frac{1}{|\mathcal{O}_{HO}|p} \sum_{o \in \mathcal{O}_{HO}} \sum_{l=1}^{p} [\mathcal{L}(o)[l] == s(o)[l]], \tag{2.2}$$

$$TS = \frac{1}{|\mathcal{O}_{HO}|p} \sum_{o \in \mathcal{O}_{HO}} \sum_{l=1}^{p} [o[l] == r(s(o))[l]]. \tag{2.3}$$

where $[\cdot]$ is the Iverson bracket.



(a) SIL **(b)** Emergent **(c)** Gumbel

FIGURE 2.3 – Comparison of sender's map, where the columns are words and rows are property values. Emergent communication uses the same word to refer to multiple property values. A perfect mapped language would be the identity matrix.

2.3.2 Baselines

In our experiments, we compare SIL with different baselines. All methods are initialized with the same pretrained model unless stated otherwise. The *Gumbel* baselines are finetuned with GSTE during interaction. These correspond to naive application of interactive training and are expected to exhibit language drift. *Emergent* is a random initializion trained with GSTE. *S2P* indicates that the agents are trained with Supervised-2-selfPlay. Our *S2P* is realized by using a weighted sum of the losses at each step:

$$L_{S2P} = L_{Gumbel} + \alpha L_{supervised} \tag{2.4}$$

where $L_{supervised}$ is the loss on the pre-train dataset and α is a hyperparameter with a default value of 1 as detailed in (Lazaridou et al., 2016, 2020).

2.3.3 Results

We present the main results for the Lewis game in Figure 2.2. For each method we used optimal hyperparameters namely $\tau = 10$ for SIL and $\tau = 1$ for rest. We also observed that SIL outperforms the baselines for any τ. Additional results in Appendix A.2 (Figures A.1 & A.2). , including a section on language drift preliminary symptoms. , where we compare the performance of Seeded Iterated Learning with vanilla gumbel and S2P.

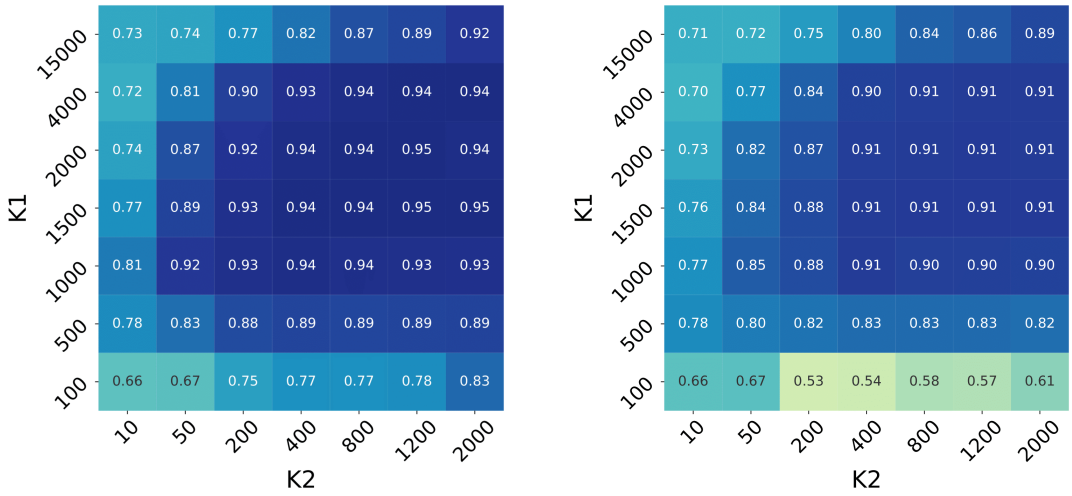
The pretrained agent has an initial task score and language score of around 65%, showing an imperfect language mapping while allowing room for task improvement. Both Gumbel and S2P are able to increase the task and language score on the held-out dataset. For both baselines, the final task score is higher than the language score. This means that some objects are reconstructed successfully with incorrect messages, suggesting language drift has occurred.

Note that, for S2P, there is some instability of the language score at the end of training. We hypothesize that it could be because our pretrained dataset in this toy setting is too small, and as a result, S2P overfits that small dataset. Emergent communication has a sender language score close to zero, which is expected. However, it is interesting to find that emergent communication has slightly lower held-out task score than Gumbel, suggesting that starting from pretrained model provides some prior for the model to generalize better. Finally, we observe that SIL achieves a significantly higher task score and sender language score, outperforming the other baselines. A high language score also shows that the sender leverages the initial language structure rather than merely re-inventing a new language, countering language drift in this synthetic experiment.

To better visualize the underlying language drift in this settings, we display the sender's map from property values to words in Figure 2.3. We observe that the freely emerged language results in re-using the same words for different property values. If the method has a higher language score, the resulting map is closer to the identity matrix.

2.3.4 SIL Properties

We perform a hyper-parameter sweep for the Lewis Game in Figure 2.4 over the core SIL parameters, k_1 and k_2, which are, respectively, the length of interactive and imitation training phase. We simply set $k_2' = k_2$ since in a toy setting the receiver can always adjust to the sender quickly. We find that for each k_2, the best k_1 is in the middle. This is expected since a small k_1 would let the imitation phase constantly disrupt the normal interactive learning, while a large k_1 would entail an already drifted teacher. We see that k_2 must be high enough to successfully transfer teacher distributions to the student. However, when a extremely large k_2 is set, we do not observe the expected performance drop predicted by the learning bottleneck: The overfitting of the student to the teacher should reduce SIL's resistance to language drift. To resolve this dilemma, we slightly modify our imitation learning process. Instead of doing supervised learning on the samples from



(a) Task Score **(b)** Language Score

FIGURE 2.4 – Sweep over length of interactive learning phase k_1 and length of imitation phase k_2 on the Lewis game (darker is higher). Low or high k_1 result in poor task and language score. Similarly, low k_2 induces poor results while high k_2 do not reduce performance as one would expect.

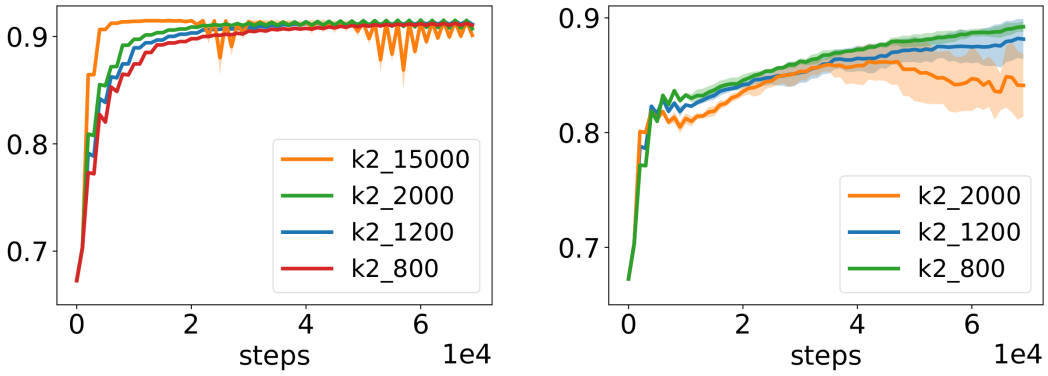
teachers, we explicitly let student imitate the complete teacher distribution by minimizing $KL(s(;\theta^T)||s(;\theta^S))$. The result is in Figure 2.5, and we can see that increasing k_2 now leads to a loss of performance, which confirms our hypotheses. In conclusion, SIL has good performance in a (large) valley of parameters, and a proper imitation learning process is also crucial for constructing the learning bottleneck.

2.4 EXPERIMENTS: THE TRANSLATION GAME

Although being insightful, the Lewis game is missing some core language properties, e.g., word ambiguity or unrealistic word distribution etc. As it relies on a basic finite language, it would be premature to draw too many conclusions from this simple setting (Hayes, 1988). In this section, we present a larger scale application of SIL in a natural language setting by exploring the translation game (Lee et al., 2019).

2.4.1 Experimental Setting

The translation game is a S/R game where two agents translate a text from a source language, French (FR), to a target language, German (De), through a pivot language,



(a) argmax **(b)** KL Minimization

FIGURE 2.5 – Language score for different k_2 by imitating greedy sampling with cross-entropy (Left) vs distilling the teacher distribution with KL minimization (Right). As distillation relaxes the learning bottleneck, we observe a drop in language score with overfitting when the student imitation learning length increases.

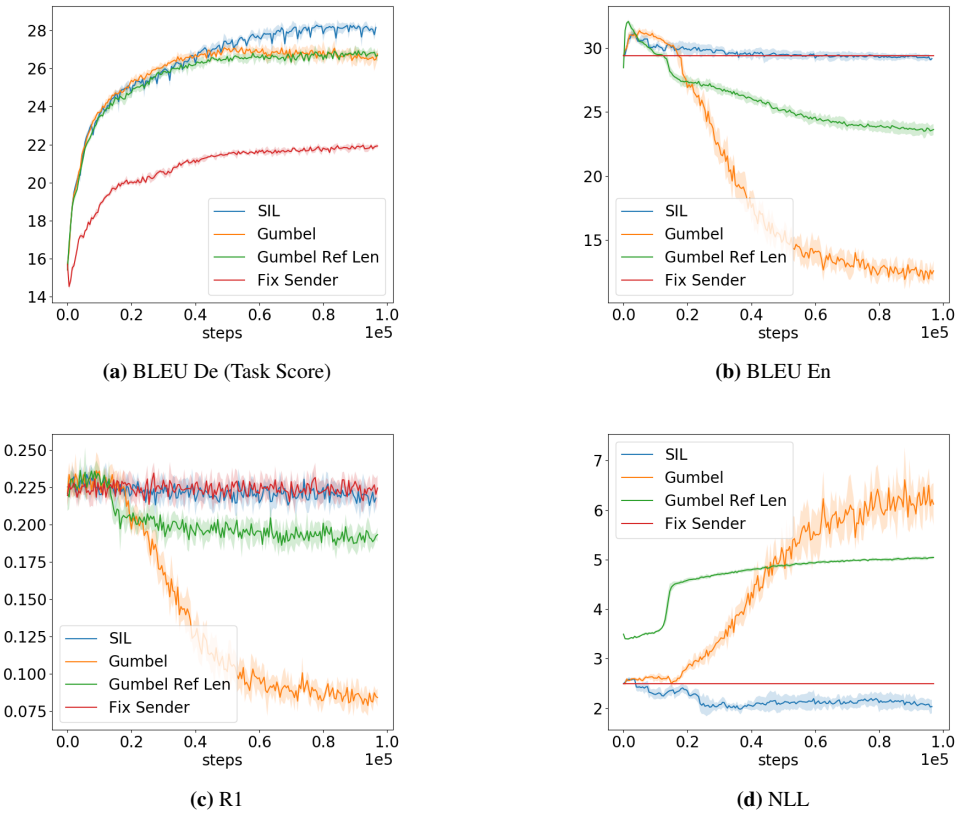
English (En). This framework allows the evaluation of the English language evolution through translation metrics while optimizing for the Fr→De translation task, making it a perfect fit for our language drift study.

The translation agents are sequence-to-sequence models with gated recurrent units (Cho et al., 2014) and attention (Bahdanau et al., 2015). First, they are independently pretrained on the IWSLT dataset (Cettolo et al., 2012) to learn the initial language distribution. The agents are then finetuned with interactive learning by sampling new translation scenarios from the Multi30k dataset (Elliott et al., 2016), which contains 30k images with the same caption translated in French, English, and German. Generally, we follow the experimental setting of Lee et al. (2019) for model architecture, dataset, and preprocessing, which we describe in Appendix A.3.2 for completeness.

However, in our experiments, we use GSTE to optimize the sender, whereas Lee et al. (2019) rely on policy gradient methods to directly maximize the task score.

2.4.2 Evaluation metrics

We monitor our task score with *BLEU(De)* (Papineni et al., 2002b), it estimates the quality of the Fr→De translation by comparing the translated German sentences to the ground truth German. We then measure the sender language score with three metrics. First, we evaluate the overall language drift with the *BLEU(En)* score from the



(a) BLEU De (Task Score)

(b) BLEU En

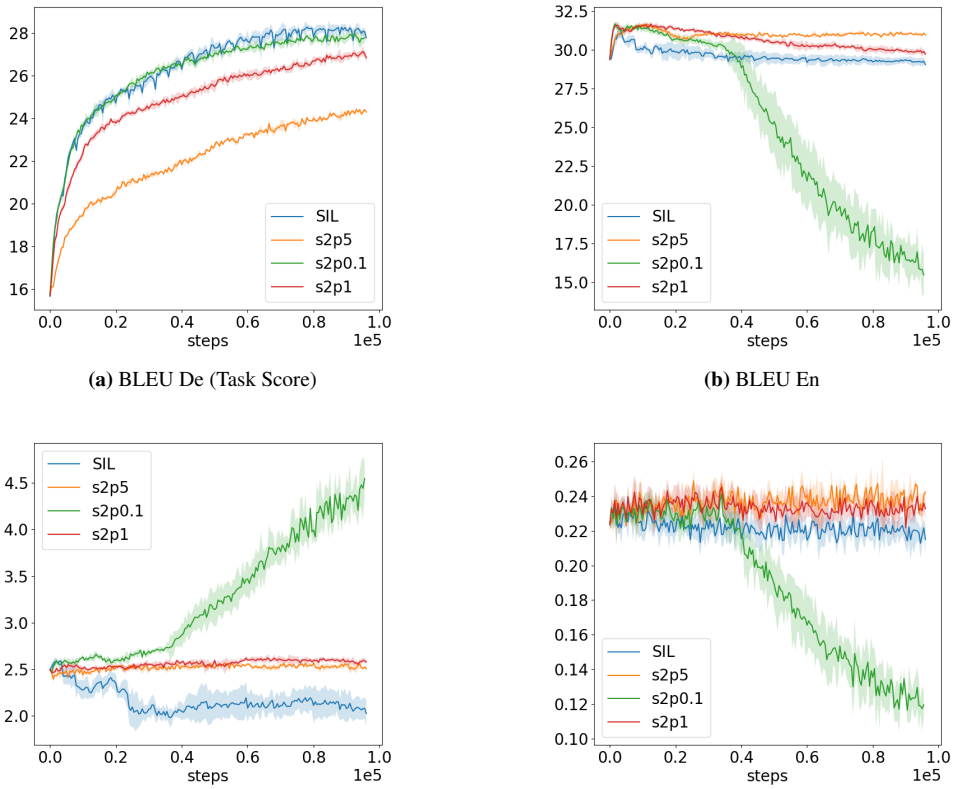
(c) R1

(d) NLL

FIGURE 2.6 – The task score and the language score of NIL, S2P, and Gumbel baselines. Fix Sender indicates the maximum performance the sender may achieve without agent co-adaptation. We observe that Gumbel language start drifting when the task score increase. Gumbel Ref Len artificially limits the English message length, which caps the drift. Finally, SIL manages to both increase language and task score

ground truth English captions. As the BLEU score controls the alignment between intermediate English messages and the French input texts, it captures basic syntactic and semantic language variations. Second, we evaluate the structural drift with the negative log-likelihood (*NLL*) of the generated English under a pretrained language model. Third, we evaluate the semantic drift by computing the image retrieval accuracy ($R1$) with a pretrained image ranker; the model fetches the ground truth image given 19 distractors and generated English.

The language and image ranker models are further detailed in Appendix A.3.3.



(a) BLEU De (Task Score)

(b) BLEU En

(c) NLL

(d) R1

FIGURE 2.7 – S2P sweep over imitation loss weight vs. interactive loss. S2P displays a trade-off between a high task score, which requires a low imitation weight, and high language score, which requires high imitation weight. SIL appears less susceptible to a tradeoff between these metrics.

2.4.3 Results

We show our main results in Figure 2.6, and a full summary in Table A.1 in Appendix A.3. Runs are averaged over five seeds and shaded areas are one standard deviation. The x-axis shows the number of interactive learning steps.

After pretraining our language agents on the IWSLT corpus, we obtain the single-agent BLEU score of 29.39 for Fr→En and 20.12 for En→De on the Multi30k captions. When combining the two agents, the Fr→De task score drops to 15.7, showing a compounding error in the translation pipeline. We thus aim to overcome this misalignment between translation agents through interactive learning while preserving an intermediate fluent English language.

As a first step, we freeze the sender to evaluate the maximum task score without

agent co-adaptation. The Fix Sender then improves the task score by 5.3 BLEU(De) while artificially maintaining the language score constant. As we latter achieve a higher task score with Gumbel, it shows that merely fixing the sender would greatly hurt the overall task performance.

We observe that the Gumbel agent improves the task score by 11.32 BLEU(De) points but the language score collapse by 10.2 BLEU(En) points, clearly showing language drift while the two agents co-adapt to solve the translation game. Lee et al. (2019) also constrain the English message length to not exceed the French input caption length, as they observe that language drift often entails long messages. Yet, this strong inductive bias only slows down language drift, and the language score still falls by 6.0 BLEU(En) points. Finally, SIL improves the task score by 12.6 BLEU(De) while preserving the language score of the pretrained model. Thus, SIL successfully counters language drift in the translation game while optimizing for task-completion.

2.4.4 S2P vs SIL

We compare the S2P and SIL learning dynamics in Figure 2.7.

S2P balances the supervised and interactive losses by setting a weight α for the imitation loss (Lazaridou et al., 2016). First, we observe that a low α value, i.e, 0.1, improves the task score by 11.8 BLEU(De), matching SIL performances, but the language score diverges. We thus respectively increase α to 1, and 5, which stops the language drift, and even outperforms SIL language score by 1.2 BLEU(En) points. However, this language stabilization also respectively lowers the task score by 0.9 BLEU(De) and 3.6 BLEU(De) compared to SIL. In other words, S2P has an inherent trade-off between task score (with low α), and language score (with high α), whereas SIL consistently excels on both task and language scores. We assume that S2P is inherently constrained by the initial training dataset.

2.4.5 Syntactic and Semantic Drifts

As described in Section 2.4.1, we attempt to decompose the Language Drift into syntactic drifts, by computing language likelihood (NLL), and semantic drifts, by aligning images and generated captions ($R1$). In Figure 2.6, we observe a clear correlation between those two metrics and a drop in the language BLEU(En) score. For instance, Vanilla-Gumbel simultaneously diverges on these three scores, while the sequence length

	SIL successfully prevent language drift	*SIL can remain close to the valid pretrained models*
Human	**two men, one in blue and one in red, compete in a boxing match.**	**there are construction workers working hard on a project**
Pretrain	two men, one in blue and the other in red, fight in a headaching game	there are workers working hard work on a project.
Gumbel	two men one of one in blue and the other in red cfighting in a acacgame.....	there are construction working hard on a project .....
S2P	two men, one in blue and the other in red, fighting in a kind of a kind.	there are workers working hard working on a project ..
SIL	two men, one in blue and the other in red, fighting in a game.	there are workers working hard on a project .
	SIL partially recovers the sentence without drifting	*SIL/S2P still drift when facing rare word occurrences (shaped lollipop)*
Human	**a group of friends lay sprawled out on the floor enjoying their time together.**	**a closeup of a child's face eating a blue , heart shaped lollipop.**
Pretrain	a group of friends on the floor of fun together.	a big one 's face plan a blue box.
Gumbel	a group of defriends comadeof on the floor together of of of of of together.....	a big face of a child eating a blue th-acof of of of chearts.....
S2P	a group of friends of their commodities on the floor of fun together.	a big face plan of eating a blue of the kind of hearts.
SIL	a group of friends that are going on the floor together.	a big plan of a child eating a blue datadof the datadof the datadof the data@@

TABLE 2.1 – Selected generated English captions. Vanilla Gumbel drifts by losing grammatical structure, repeating patches of words, and inject noisy words. Both S2P and SIL counter language drift by generating approximately correct and understandable sentences. However, they become unstable when dealing with rare word occurrences.

constraint caps the drifts. We observe that SIL does not improve language semantics, i.e., $R1$ remains constant during training, whereas it produces more likely sentences as the NLL is improved by 11%. Yet, S2P preserves slightly better semantic drift, but its language likelihood does not improve as the agent stays close to the initial distribution.

2.4.6 SIL Mechanisms

We here verify the initial motivations behind SIL by examining the impact of the learning bottleneck in Figure 2.8 and the structure-preserving abilities of SIL in Figure 2.9. As motivated in Section 2.2.2, each imitation phase in the SIL aims to filtering-out emergent unstructured language by generating an intermediate dataset to train the student. To verify this hypothesis, we examine the change of negative language likelihood (NLL) from the teacher to the student after imitation. We observe that after imitation, the student consistently improves the language likelihood of its teacher, indicating a more regular language production induced by the imitation step. In another experiment, we stop the iterated learning loop after 20k, 40k and 60k steps and continue with standard interactive training. We observe that the agent's language score starts dropping dramatically as soon as we stop SIL while the task score keep improving. This finding supports the view that SIL persists in preventing language drift throughout training, and that the language drift phenomenon itself appear to be robust and not a result of some unstable

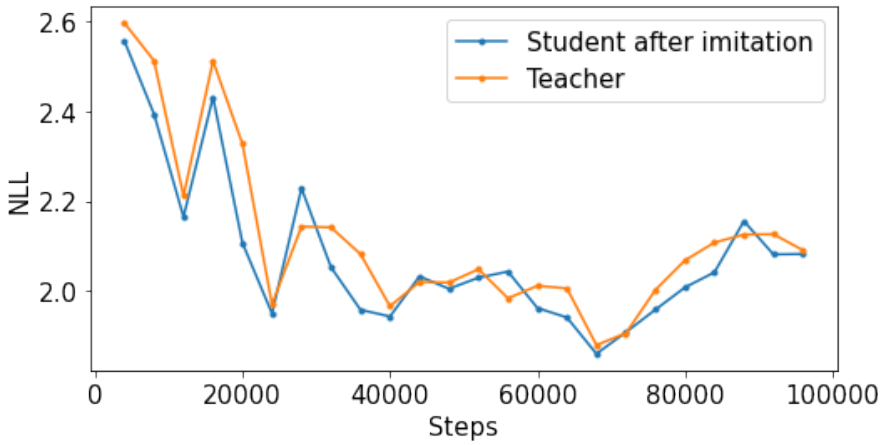
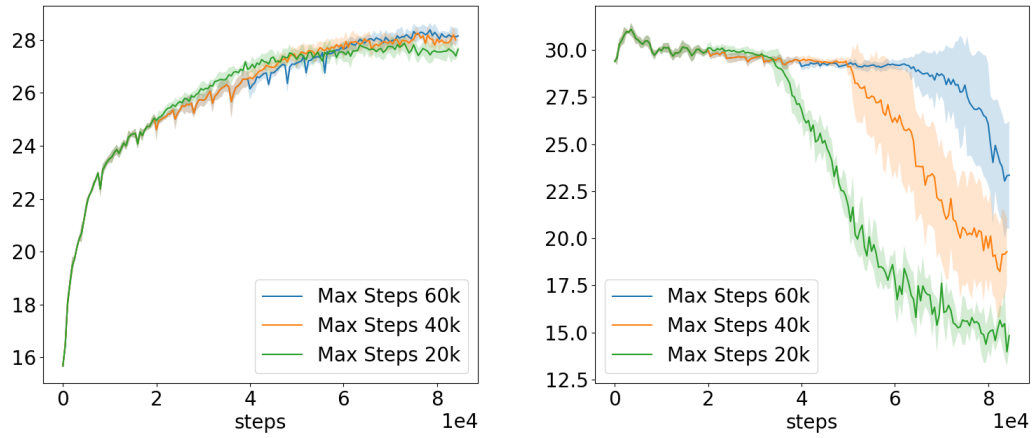


FIGURE 2.8 – NLL of the teacher and the student after imitation learning phase. In the majority of iterations, the student after imitation obtains a lower NLL than the teacher, after supervised training on the teacher's generated data.

initialization point.

2.4.7 Qualitative Analysis

In Table 2.1, we show some hand-selected examples of English messages from the translation game. As expected, we observe that the vanilla Gumbel agent diverges from the pretrained language models into unstructured sentences, repeating final dots or words. It also introduce unrecognizable words such as "cfighting" or "acacgame" by randomly pairing up sub-words whenever it faces rare word tokens. S2P and SIL successfully counter the language drift, producing syntactically valid language. However, they can still produce semantically inconsistent captions, which may be due to the poor pretrained model, and the lack of grounding (Lee et al., 2019). Finally, we still observe language drift when dealing with rare word occurrences. Additional global language statistics can be found in Appendix that supports that SIL preserves language statistical properties.



(a) BLEU De

(b) BLEU En

FIGURE 2.9 – Effect of stopping SIL earlier in the training process. SIL maximum steps set at 20k, 40k and 60k. SIL appears to be important in preventing language drift through-out training.

3 Supervised Seeded Iterated Learning for Interactive Language Learning

Authors: Yuchen Lu (YL), **Soumye Singhal** (SS), Florian Strub(FS), Olivier Pietquin(OP), and Aaron Courville (AC).

This work is an extension of the work presented in Chapter 2 and hence the contributions are similar to Chapter 2. This work appeared as a short paper at EMNLP 2020 (Lu et al., 2020a).

Contributions: AC proposed the research problem and the collaboration. YL and SS led the project with FS, OP, and AC as advisors. YL found the relevant literature on iterated learning and proposed to use it to counter language drift, which motivated our method. YL, SS, FS, OP & AC concretized the idea over many iterations in the daily meetings. YL and SS jointly managed the project schedule, led all the meetings, conducted the experiments, and presented them at the meetings. YL and SS did extensive early experiments on the lewis game to make the method work. YL did the final Translation game experiments and adapted the code for it. YL, SS, and FS contributed to the first draft of the paper.

FS and OS advised on the domain knowledge about task-oriented Dialogue. AC supervised the project, helped define the scope and gave clarity on the research direction. AC and OP gave feedback and edited the paper. Everyone attended and contributed to the meetings.

Affiliation

— Yuchen Lu, Mila, Université de Montréal

— Soumye Singhal, Mila, Université de Montréal

— Florian Strub, Deepmind

— Olivier Pietquin, Google Brain

— Aaron Courville, Mila, Université of Montréal

3.1 INTRODUCTION

Since the early days of NLP Winograd (1971), conversational agents have been designed to interact with humans through language to solve diverse tasks, e.g., remote instructions Thomason et al. (2015) or booking assistants Bordes et al. (2017); El Asri et al. (2017). In this goal-oriented dialogue setting, the conversational agents are often designed to compose with predefined language utterances Lemon and Pietquin (2007); Williams et al. (2014); Young et al. (2013). Even if such approaches are efficient, they also tend to narrow down the agent's language diversity. To remove this restriction, recent work has been exploring interactive word-based training. In this setting, the agents are generally trained through a two-stage process Wei et al. (2018b); De Vries et al. (2017); Shah et al. (2018); Li et al. (2016a); Das et al. (2017): Firstly, the agent is pretrained on a human-labeled corpus through supervised learning to generate grammatically reasonable sentences. Secondly, the agent is finetuned to maximize the task-completion score by interacting with a user. Due to sample-complexity and reproducibility issues, the user is generally replaced by a game simulator that may evolve with the conversational agent. Unfortunately, this pairing may lead to the *language drift* phenomenon, where the conversational agents gradually co-adapt, and drift away from the pretrained natural language. The model thus becomes unfit to interact with humans Chattopadhyay et al. (2017); Zhu et al. (2017); Lazaridou et al. (2020).

While domain-specific methods exist to counter language drift Lee et al. (2019); Li et al. (2016b), a simple task-agnostic method consists of combining interactive and supervised training losses on a pretraining corpus Wei et al. (2018b); Lazaridou et al. (2016), which was later formalized as Supervised SelfPlay (S2P) Lowe et al. (2020).

Inspired by language evolution and cultural transmission Kirby (2001); Kirby et al. (2014), recent work proposes Seeded Iterated Learning (SIL) Lu et al. (2020b) as another task-agnostic method to counter language drift. SIL modifies the training dynamics by iteratively refining a pretrained student agent by imitating interactive agents, as illustrated in Figure 3.1. At each iteration, a teacher agent is created by duplicating the student agent, which is then finetuned towards task completion. A new dataset is then generated by greedily sampling the teacher, and those samples are used to refine the student through supervised learning. The authors empirically show that this iterated learning procedure induces an inductive learning bias that successfully maintains the language grounding while improving task-completion.

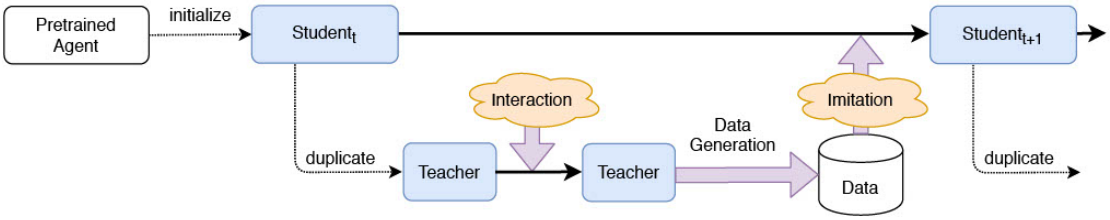


FIGURE 3.1 – SIL Lu et al. (2020b). A student agent is iteratively refined using newly generated data from a teacher agent. At each iteration, a teacher agent is created on top of the student before being finetuned by interaction, e.g. maximizing a task completion-score. The teacher generates a dataset with greedy sampling and students imitate those samples. The interaction step involves interaction with another language agent.

As a first contribution, we further examine the performance of these two methods in the setting of a translation game Lee et al. (2019). We show that S2P is unable to maintain a high grounding score and experiences a late-stage collapse, while SIL has a higher negative likelihood when evaluated on human corpus.

We propose to combine SIL with S2P by applying an S2P loss in the interactive stage of SIL. We show that the resulting *Supervised Seeded Iterated Learning* (SSIL) algorithm manages to get the best of both algorithms in the translation game. Finally, we observe that the late-stage collapse of S2P is correlated with conflicting gradients before showing that SSIL empirically reduces this gradient discrepancy.

3.2 PREVENTING LANGUAGE DRIFT

We describe here our interactive training setup before introducing different approaches to prevent language drift. In this setting, we have a set of collaborative agents that interact through language to solve a task. To begin, we train the agents to generate natural language in a word-by-word fashion. Then we finetune the agents to optimize a task completion score through interaction, i.e., learning to perform the task better. Our goal is to prevent language drift in this second stage.

3.2.1 Initializing the Conversational Agents

For a language agent f parameterized by $\boldsymbol{\theta}$, and a sequence of generated words $\boldsymbol{w}_{1:i} = [w_j]_{j=1}^{i}$ and an arbitrary context $\boldsymbol{c}$, the probability of the next word w_i is

$$p(w_{i+1}|\boldsymbol{w}_{1:i}, \boldsymbol{c}) = f_{\boldsymbol{\theta}}(\boldsymbol{w}_{1:i}, \boldsymbol{c})$$

We pretrain the language model to generate meaningful sentences by minimizing the cross-entropy loss $\mathcal{L}^{\text{CE}}_{pretrain}$ where the word sequences are sampled from a language corpus $D_{pretrain}$. Note that this language corpus may either be task-related or generic. Its role is to get our conversational agents a reasonable initialization.

3.2.2 Supervised Selfplay (S2P)

A common way to finetune the language agents while preventing language drift is to replay the pretraining data during the interaction stage. In S2P the training loss encourages both maximizing task-completion while remaining close to the initial language distribution. Formally,

$$\mathcal{L}^{\text{S2P}} = \mathcal{L}^{\text{INT}} + \alpha\mathcal{L}^{\text{CE}}_{pretrain} \tag{3.1}$$

where $\mathcal{L}^{\text{INT}}$ is a differentiable interactive loss maximizing task completion, e.g. reinforcement learning with policy gradients Sutton et al. (2000), Gumbel Straight-through Estimator (STE) Jang et al. (2017b) etc., $\mathcal{L}^{\text{CE}}_{pretrain}$ is a cross-entropy loss over the pretraining samples. α is a positive scalar which balances the two losses.

3.2.3 Seeded Iterated Learning (SIL)

Seeded Iterated Learning (SIL) iteratively refines a pretrained *student* model by using data generated from newly trained *teacher* agents Lu et al. (2020b). As illustrated in Figure 3.1, the student agent is initialized with the pretrained model. At each iteration, a new teacher agent is generated by duplicating the student parameters. It is tuned to maximize the task-completion score by optimizing the interactive loss $\mathcal{L}^{\text{TEACHER}} = \mathcal{L}^{\text{INT}}$ In a second step, we sample from the teacher to generate new training data $D_{teacher}$, and we refine the student by minimizing the cross-entropy loss $\mathcal{L}^{\text{STUDENT}} = \mathcal{L}^{\text{CE}}_{teacher}$ where sequence of words are sampled from $D_{teacher}$. This imitation learning stage can induce an information bottleneck, encouraging the student to learn a well-formatted language

Finetuning Methods	*Training Losses*
Gumbel	$\mathcal{L}^{\text{INT}}$
S2P	$\mathcal{L}^{\text{INT}} + \alpha\mathcal{L}^{\text{CE}}_{pretrain}$
SIL (teacher)	$\mathcal{L}^{\text{INT}}$
SIL (student)	$\mathcal{L}^{\text{CE}}_{teacher}$
SSIL (teacher)	$\mathcal{L}^{\text{INT}} + \alpha\mathcal{L}^{\text{CE}}_{pretrain}$
SSIL (student)	$\mathcal{L}^{\text{CE}}_{teacher}$

TABLE 3.1 – Finetuning with respective training objective.

rather than drifted components.

3.2.4 SSIL: Combining SIL and S2P

S2P and SIL have two core differences: first, SIL never re-uses human pretraining data. As observed in Section 3.4.1, this design choice reduces the language modeling ability of SIL-trained agents, with a higher negative likelihood when evaluated on human corpus. Second, S2P agents merge interactive and supervised losses, whereas SIL's student never experiences an interactive loss. As analyzed in Section 3.4.3, the S2P multi-task loss induces conflicting gradients, which may trigger language drift. In this paper, we propose to combine these two approaches and demonstrate that the combination effectively minimizes their respective weaknesses. To be specific, we apply the S2P loss over the SIL teacher agent, which entails $\mathcal{L}^{\text{TEACHER}} = \mathcal{L}^{\text{INT}} + \alpha\mathcal{L}^{\text{CE}}_{pretrain}$. We call the resulting algorithm, Supervised Seeded Iterated Learning (SSIL). In SSIL, teachers can generate data close to the human distribution due to the S2P loss. At the same time, students are updated with a consistent supervised loss to avoid the potential weakness of multi-task optimization. In addition, SSIL still maintains the inductive learning bias of SIL. We list all these methods in Table 3.1 for easy comparison. We also experiment with other ways of combining SIL and S2P by mixing the pretraining data with teacher data during the imitation learning stage. We call this method *MixData*. We show the results of this approach in section 3.4.2. We find that this approach is very sensitive to the mixing ratio of these two kinds of data, and the best configuration is still not as good as SSIL.

3.3 EXPERIMENTAL SETTING

3.3.1 Translation Game

We replicate the translation game setting from Lee et al. (2019) as it was designed to study language drift. First, a *sender* agent translates French to English (Fr-En), while a *receiver* agent translates English to German (En-De). The sender and receiver are then trained together to translate French to German with English as a pivot language. For each French sentence, we sample English from the sender, send it to the receiver, and sample German from the receiver.

The task score is defined as the BLEU score between generated German translation and the ground truth (*BLEU De*) Papineni et al. (2002b). The goal is to improve the task score without losing the language structure of the intermediate English language.

3.3.2 Training Details

Most of the training and implementation details, including the metrics used follow from Chapter 2 section 2.4. We use the Moses tokenizer Koehn et al. (2007a) and learn a byte-pair-encoding(BPE) Sennrich et al. (2016a) from Multi30K with text from all languages. Then this BPE is applied to different datasets. Our vocab size for En, Fr, De is 11552, 13331, and 12124 respectively.

The sender and the receiver are pretrained on the IWSLT dataset Cettolo et al. (2012) which contains (Fr, En) and (En, De) translation pairs. We then use the Multi30k dataset (Elliott et al., 2016) to build the finetuning dataset with (Fr, De) pairs. As IWSLT is a generic translation dataset and Multi30k only contains visually grounded translated captions, we also call IWSLT task-agnostic while Multi30K task-related. We use the cross-entropy loss of German as the interactive training objective, which is differentiable w.r.t. the receiver. For the sender, we use Gumbel Softmax straight-through estimator to make the training objective also differentiable w.r.t. the sender, as in Lu et al. (2020b).

Our language model is trained with captions data from MSCOCO Lin et al. (2014). For image ranker, we use the captions in Multi30K as well as the original Flickr30K images. We use a ResNet152 with pretrained ImageNet weights to extract the image features which are then normalized.

3.3.3 Metrics for Grounding Scores

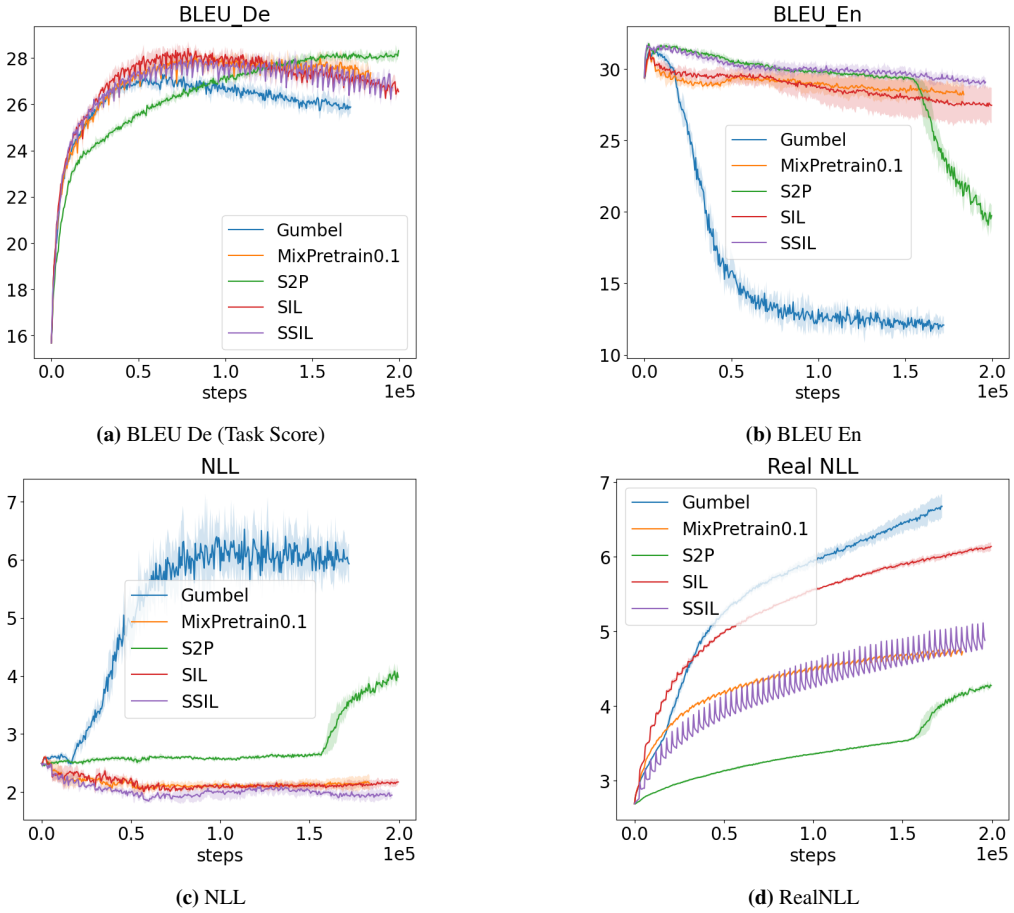
In practice, there are different kinds of language drift Lazaridou et al. (2020) (e.g. syntactic drift and semantic drift). We thus have multiple metrics to consider when evaluating language drift. We first compute English BLEU score (*BLEU En*) comparing the generated English translation with the ground truth human translation. We include the negative log-likelihood (*NLL*) of the generated En translation under a pretrained language model as a measure of syntactic correctness. In line with Lu et al. (2020b) , we also report results using another language metric: the negative log-likelihood of human translations (*RealNLL*) given a finetuned Fr-En model. We feed the finetuned sender with human task-data to estimate the model's log-likelihood. The lower is this score, the more likely the model would generate such human-like language.

3.4 EXPERIMENTS

3.4.1 S2P and SIL Weaknesses

We report the task and grounding scores of vanilla Gumbel, S2P, SIL, and SSIL in Figure 3.2. The respective best hyper-parameters can be found in the appendix. As reported by Lu et al. (2020b), vanilla Gumbel successfully improves the task score *BLEU De*, but the *BLEU En* score as well as other grounding metric collapses, indicating a language drift during the training. Both S2P and SIL manage to increase *BLEU De* while maintaining a higher *BLEU En* score, countering language drift. However, S2P has a sudden (and reproducible) late-stage collapse, unable to maintain the grounding score beyond 150k steps. On the other hand, SIL has a much higher RealNLL than S2P, suggesting that SIL has a worse ability to model human data.

SSIL seems to get the best of the two worlds. It has a similar task score *BLEU De* as S2P and SIL, while it avoids the late-stage collapse. It ends up with the highest *BLEU En*, and it improves the RealNLL over SIL, though still not as good as S2P. Also, it achieves even better NLL, suggesting that the pretrained language model favours its outputs.



(a) BLEU De (Task Score)

(b) BLEU En

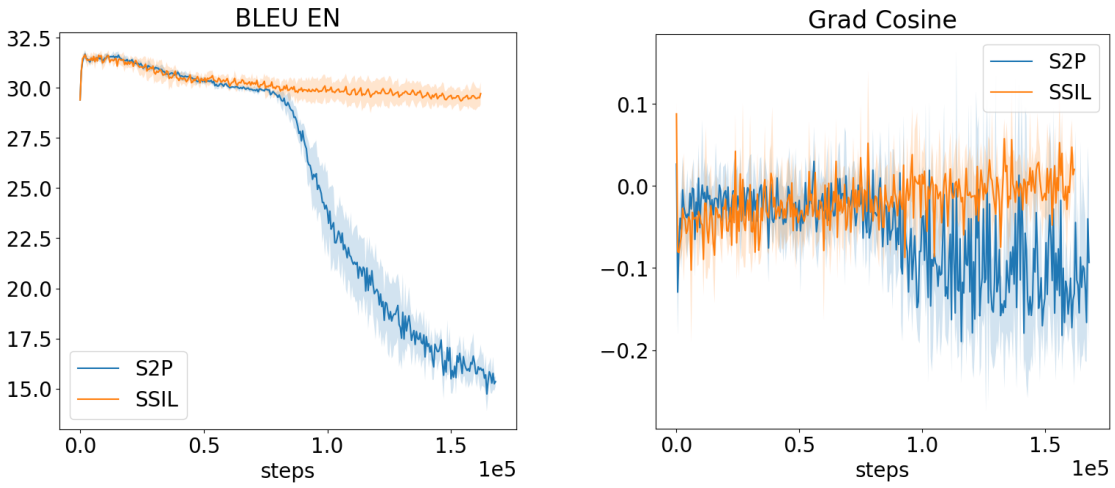
(c) NLL

(d) RealNLL

FIGURE 3.2 – Task and language metrics for Vanilla Gumbel, SIL, S2P, and SSIL in the translation game average over 5 seeds. We also show the results of mixing pretraining data in the teacher dataset (Section 3.4.2). The plots are averaged over 5 seeds with shaded area as standard deviation. Although SIL and S2P both counter language drift, S2P suffers from late collapse, and SIL has a high *RealNLL*, suggesting that its output may not correlate well with human sentences.

3.4.2 Mixing Teacher and Human Data

We also explore whether injecting pretraining data into the teacher dataset may be a valid substitute for the S2P loss. We add a subset of the pretraining data in the teacher dataset before refining the student, and we report the results in Figure 3.2 and B.2. Unfortunately, such an approach was quite unstable, and it requires heavy hyper-parameters tuning to match SSIL scores. As explained in Kirby (2001), iterated learning rely on inductive learning to remove language irregularities during the imitation step. Thus, mixing two language distributions may disrupt this imitation stage.



(a) Bleu En

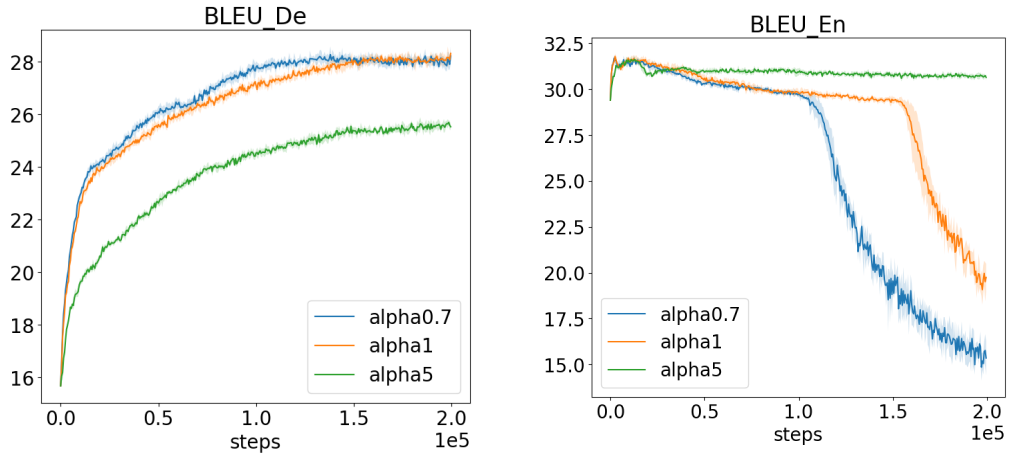
(b) Cosine Similarity

FIGURE 3.3 – Cosine similarity between the gradients issued from $\mathcal{L}^{\text{INT}}$ and $\mathcal{L}^{\text{CE}}_{pretrain}$. The collapse of the BLEU En matches the negative cosine similarity.We here set $\alpha = 0.5$ but similar values yield identical behavior as shown in Figure B.1 in Appendix.

3.4.3 Why S2P collapses ?

We investigate the potential cause of S2P late-stage collapse and how SSIL may resolve it. We firstly hope to solve this by increasing the supervised loss weight α. However, we find that a larger α only delays the eventual collapse as well as decreases the task score, as shown in Figure 3.4.

We further hypothesize that this late-stage collapse can be caused by the distribution mismatch between the pretraining data (IWSLT) and the task-related data (Multi30K), exemplified by their word frequencies difference. A mismatch between the two losses could lead to conflicting gradients, which could, in turn, make training unstable. In Figure 3.3, we display the cosine similarity of the sender gradients issued by the interactive and supervised losses $cos(\nabla_{sender}\mathcal{L}^{\text{INT}}, \nabla_{sender}\mathcal{L}^{\text{CE}}_{pretrain})$ for both S2P and SSIL for $\alpha = 0.5$ during training. Early in S2P training, we observe that the two gradients remain orthogonal on average, with the cosine oscillating around zero. Then, at the same point where the S2P *Bleu En* collapses, the cosine of the gradients starts trending negative, indicating that the gradients are pointing in opposite directions. However, SSIL does not have this trend, and the *BLEU En* does not collapse. Although the exact mechanism of how conflicting gradients trigger the language drift is unclear, current results favor our hypothesis and suggest that language drift could result from standard multi-task opti-



(a) BLEU De (Task Score)

(b) BLEU En

FIGURE 3.4 – S2P with different α. In general, one can find that for S2P there is a trade-off between grounding score and task score controlled by α. A larger α might delay the eventual collapse. However, if the α is too large, the task score will decrease significantly. As a result, even though increasing α seems to fit the intuition, it cannot fix the problem.

mization issues Yu et al. (2020); Parisotto et al. (2016); Sener and Koltun (2018) for S2P-like methods.

4 Conclusion

In this thesis, we talk about the problem of language drift in task-oriented dialogue settings and propose a method named Seeded Iterated Learning (SIL) to counter it. The method is based on the broader principle of iterated learning from cognitive sciences. It alternates between task optimization steps and student-teacher imitation learning steps to simulate iterated cultural transmission of language. We modified the iterated learning principle so that it starts from a seed model pre-trained on actual human data, and the language properties are preserved during training. Our extensive experimental study revealed that this method outperforms standard baselines both in terms of keeping a syntactic language structure and of solving the task. We also talk about the trade-offs between the language score and the task score especially in the S2P baseline. We analyzed S2P and SIL and highlighted their shortcomings. S2P experiences a late-stage collapse on the grounding score, whereas SIL has a higher negative likelihood on human corpus. Later we introduce SSIL to combine these two methods effectively. We further show the correlation between S2P late-stage collapse and conflicting gradients between task and language optimization objectives. This problem of conflicting gradients is also observed in the field of multi-task optimization (Yu et al., 2020).

The problem of language drift is not restricted to task-oriented dialogue and surfaces in many settings where language agents are trained on some external metric which is agnostic to language quality. For eg, a lot of multi-agent setups involve agents communicating with each other using language in either competitive or collaborative scenarios. For effective communication it's crucial that the language doesn't drift. The methods proposed in this thesis are fairly general and can be extended to many of these settings. Furthermore, iterated learning is a powerful and general unsupervised learning method which can be used to learn structured compositional representations from the data using the learning bottleneck. It has also seen successful applications in domains like visual-question answering (Vani et al., 2021) and multi-label object-classification (Rajeswar et al., 2021).

Bibliographie

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*, 2020.

Akshat Agarwal, Swaminathan Gurumurthy, Vasu Sharma, Mike Lewis, and Katia Sycara. Community regularization of visually-grounded dialog. In *Proc. of International Conference on Autonomous Agents and MultiAgent Systems*, 2019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proc. of of International Conference on Learning Representations*, 2015.

Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron Courville. Systematic generalization: What is required and can it be learned ? In *Proc. of International Conference on Learning Representations*, 2019.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. In *Proc. of International Conference on Learning Representations*, 2017.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*, 2018.

Mauro Cettolo, Christian Girardi, and Marcello Federico. Wit3: Web inventory of transcribed and translated talks. In *Proc. of Conference of european association for machine translation*, 2012.

Prithvijit Chattopadhyay, Deshraj Yadav, Viraj Prabhu, Arjun Chandrasekaran, Abhishek Das, Stefan Lee, Dhruv Batra, and Devi Parikh. Evaluating visual conversational agents via cooperative human-ai games. In *Proc. of AAAI Conference on Human Computation and Crowdsourcing*, 2017.

Bernard Chazelle and Chu Wang. Self-sustaining iterated learning. In *Proc. of the Innovations in Theoretical Computer Science Conference*, 2017.

Bernard Chazelle and Chu Wang. Iterated learning in dynamic social networks. *The Journal of Machine Learning Research*, 20(1):979–1006, 2019.

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 Empirical Methods in Natural Language Processing*, 2014.

Noam Chomsky. Poverty of stimulus: Unfinished business. *Transcript of a presentation given at Johannes-Gutenberg University, Mainz*, 2010.

Morten H Christiansen and Simon Ed Kirby. *Language evolution.* Oxford University Press, 2003.

Herbert H Clark and Eve V Clark. Psychology and language. 1977.

Michael Cogswell, Jiasen Lu, Stefan Lee, Devi Parikh, and Dhruv Batra. Emergence of compositional language with deep generational transmission. *arXiv preprint arXiv:1904.09067*, 2019.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537, 2011.

Gautier Dagan, Dieuwke Hupkes, and Elia Bruni. Co-evolution of language and agents in referential games. *arXiv preprint arXiv:2001.03361*, 2020.

Abhishek Das, Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. Learning cooperative visual dialog agents with deep reinforcement learning. In *Proc. of International Conference on Computer Vision*, 2017.

Jill G De Villiers, Jill De Villiers, Peter A De Villiers, and Peter A DeVilliers. *Language acquisition*. Harvard University Press, 1978.

Harm De Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. Guesswhat ?! visual object discovery through multi-modal dialogue. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Layla El Asri, Hannes Schulz, Shikhar Sharma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. Frames: a corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the SIGdial Meeting on Discourse and Dialogue*, 2017.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. Multi30k: Multilingual english-german image descriptions. In *Proc. of Workshop on Vision and Language*, 2016.

Nick C Ellis and Diane Larsen-Freeman. Language emergence: Implications for applied linguistics—introduction to the special issue. *Applied linguistics*, 27(4):558–589, 2006.

Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. Vse++: Improving visual-semantic embeddings with hard negatives. *arXiv preprint arXiv:1707.05612*, 2017.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *Proc. of Neural Information Processing Systems*, 2018.

Jianfeng Gao, Michel Galley, Lihong Li, et al. Neural approaches to conversational ai. *Foundations and Trends in Information Retrieval*, 13(2-3):127–298, 2019.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Thomas L Griffiths and Michael L Kalish. A bayesian view of language evolution by iterated learning. 2005.

Shangmin Guo, Yi Ren, Serhii Havrylov, Stella Frank, Ivan Titov, and Kenny Smith. The emergence of compositional languages for numeric concepts through iterated learning in neural agents. *arXiv preprint arXiv:1910.05291*, 2019.

Abhinav Gupta, Ryan Lowe, Jakob Foerster, Douwe Kiela, and Joelle Pineau. Seeded self-play for language learning. In *Proc. of Beyond Vision and LANguage: inTEgrating Real-world kNowledge (LANTERN)*, 2019.

Serhii Havrylov and Ivan Titov. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. In *Proc. of Neural Information Processing Systems*, 2017.

Patrick J Hayes. The second naive physics manifesto. *Formal theories of the common sense world*, 1988.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. Revisiting self-training for neural sequence generation. In *Proc. of International Conference on Learning Representations*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *Proc. of International Conference on Learning Representations*, 2017a.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparametrization with gumble-softmax. In *International Conference on Learning Representations (ICLR)*, 2017b.

Daniel Jurafsky and James H Martin. Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.

Michael L Kalish, Thomas L Griffiths, and Stephan Lewandowsky. Iterated learning: Intergenerational knowledge transmission reveals inductive biases. *Psychonomic Bulletin & Review*, 14(2):288–294, 2007.

Kyungyul Kim, ByeongMoon Ji, Doyoung Yoon, and Sangheum Hwang. Self-knowledge distillation: A simple way for better generalization. *arXiv preprint arXiv:2006.12000*, 2020.

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*, 2016.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Simon Kirby. Spontaneous evolution of linguistic structure-an iterated learning model of the emergence of regularity and irregularity. *IEEE Transactions on Evolutionary Computation*, 5(2):102–110, 2001.

Simon Kirby. Natural language from artificial life. *Artificial life*, 8(2):185–215, 2002.

Simon Kirby and Tom Griffiths. Iterated learning and the evolution of language. *Current Opinion in Neurobiology*, 28:108–114, 10 2014. doi: 10.1016/j.conb.2014.07.014.

Simon Kirby, Tom Griffiths, and Kenny Smith. Iterated learning and the evolution of language. *Current opinion in neurobiology*, 28:108–114, 2014.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June 2007a. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P07-2045.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proc. of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, 2007b.

Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. Natural language does not emerge 'naturally' in multi-agent dialog. In *Proc. of Empirical Methods in Natural Language Processing*, 2017.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. *Proc. of Internation Conference on Learning Representations*, 2018.

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language. In *Proc. of Internation Conference on Learning Representations*, 2016.

Angeliki Lazaridou, Anna Potapenko, and Olivier Tieleman. Multi-agent communication meets natural language: Synergies between functional and structural language learning. In *Proc. of the Association for Computational Linguistics*, 2020.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553): 436–444, 2015.

Jason Lee, Kyunghyun Cho, and Douwe Kiela. Countering language drift via visual grounding. In *Proc. of Empirical Methods in Natural Language Processing*, 2019.

Oliver Lemon and Olivier Pietquin. *Data-driven methods for adaptive spoken dialogue systems: Computational learning for conversational interfaces*. Springer Science & Business Media, 2012.

Olivier Lemon and Olivier Pietquin. Machine learning for spoken dialogue systems. In *Proceedings of European Conference on Speech Communication and Technologies (Interspeech)*, 2007.

Esther Levin, Roberto Pieraccini, and Wieland Eckert. A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Transactions on speech and audio processing*, 8(1):11–23, 2000.

David K. Lewis. *Convention: A Philosophical Study*. Wiley-Blackwell, 1969.

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. Deal or no deal ? end-to-end learning of negotiation dialogues. In *Proc. of Empirical Methods in Natural Language Processing*, pages 2443–2453, 2017.

Fushan Li and Michael Bowling. Ease-of-teaching and language structure from emergent communication. In *Proc. of Neural Information Processing Systems*, 2019.

Jiwei Li, Alexander H Miller, Sumit Chopra, Marc'Aurelio Ranzato, and Jason Weston. Dialogue learning with human-in-the-loop. In *Proc. of International Conference on Learning Representations*, 2016a.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep reinforcement learning for dialogue generation. In *Proc. of Empirical Methods in Natural Language Processing*, 2016b.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proc. of European Conference on Computer Vision*, 2014.

Ryan Lowe, Abhinav Gupta, Jakob Foerster, Douwe Kiela, and Joelle Pineau. On the interaction between supervision and self-play in emergent communication. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rJxGLlBtwH.

Yuchen Lu, Soumye Singhal, Florian Strub, Olivier Pietquin, and Aaron Courville. Supervised seeded iterated learning for interactive language learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3962–3970, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.325. URL https://aclanthology.org/2020.emnlp-main.325.

Yuchen Lu, Soumye Singhal, Florian Strub, Olivier Pietquin, and Aaron Courville. Countering language drift with seeded iterated learning. In *Proceedings of International Conference of Machine Learning (ICML)*, 2020b.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Proc. of International Conference on Learning Representations*, 2017.

Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

Hossein Mobahi, Mehrdad Farajtabar, and Peter L Bartlett. Self-distillation amplifies regularization in hilbert space. *arXiv preprint arXiv:2002.05715*, 2020.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002a. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002b.

Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2016.

Steven Pinker. Language acquisition. *Language: An invitation to cognitive science*, 1: 135–82, 1995.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog 1.8*, 2019.

Sai Rajeswar, Pau Rodriguez, Soumye Singhal, David Vazquez, and Aaron Courville. Multi-label iterated learning for image classification with label ambiguity. *arXiv preprint arXiv:2111.12172*, 2021.

Yi Ren, Shangmin Guo, Matthieu Labeau, Shay B. Cohen, and Simon Kirby. Compositional languages emerge in a neural iterated learning model. In *Proc. of International Conference on Learning Representations*, 2020.

Stuart Jonathan Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, England, third edition, 2016. ISBN 9781292153964. URL https://www.pearson.com/us/higher-education/program/PGM156683.html.

Jost Schatzmann, Karl Weilhammer, Matt Stuttle, and Steve Young. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The knowledge engineering review*, 21(2):97–126, 2006.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 527–538, 2018.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016a. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://www.aclweb.org/anthology/P16-1162.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016b.

Pararth Shah, Dilek Hakkani-Tur, Bing Liu, and Gokhan Tur. Bootstrapping a neural conversational agent with dialogue self-play, crowdsourcing and on-line reinforce-

ment learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 41–51, 2018.

Daniel L Silver and Robert E Mercer. The task rehearsal method of life-long learning: Overcoming impoverished data. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 90–101. Springer, 2002.

Gabriel Skantze and Anna Hjalmarsson. Towards incremental speech generation in dialogue systems. In *Proc.of the Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, 2010.

Elizabeth S Spelke. Nativism, empiricism, and the origins of knowledge. *Infant Behavior and Development*, 21(2):181–200, 1998.

Florian Strub, Harm De Vries, Jeremie Mary, Bilal Piot, Aaron Courville, and Olivier Pietquin. End-to-end optimization of goal-driven and visually grounded dialogue systems. In *Proc. of International Joint Conferences on Artificial Intelligence*, 2017.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, 2000.

Jesse Thomason, Shiqi Zhang, Raymond J Mooney, and Peter Stone. Learning to interpret natural language commands through human-robot dialog. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. What is left to be understood in atis? In *2010 IEEE Spoken Language Technology Workshop*, pages 19–24. IEEE, 2010.

Ankit Vani, Max Schwarzer, Yuchen Lu, Eeshan Dhekane, and Aaron Courville. Iterated learning for emergent systematicity in vqa. In *ICLR*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Wei Wei, Quoc Le, Andrew Dai, and Jia Li. Airdialogue: An environment for goal-oriented dialogue research. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3844–3854, 2018a.

Wei Wei, Quoc Le, Andrew Dai, and Jia Li. Airdialogue: An environment for goal-oriented dialogue research. In *Proc. of Empirical Methods in Natural Language Processing*, 2018b.

Jason D Williams, Matthew Henderson, Antoine Raux, Blaise Thomson, Alan Black, and Deepak Ramachandran. The dialog state tracking challenge series. *AI Magazine*, 35(4):121–124, 2014.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Terry Winograd. Procedures as a representation of data in a computer program for understanding natural language. Technical report, PhD thesis, Massachusets Institute of Technology, January 1971.

Qizhe Xie, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Self-training with noisy student improves imagenet classification. *arXiv preprint arXiv:1911.04252*, 2019.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

Licheng Yu, Hao Tan, Mohit Bansal, and Tamara L Berg. A joint speaker-listener-reinforcer model for referring expressions. In *Proc. of Computer Vision and Pattern Recognition*, 2017.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *arXiv preprint arXiv:2001.06782*, 2020.

Yan Zhu, Shaoting Zhang, and Dimitris Metaxas. Interactive reinforcement learning for object grounding via self-talking. *Visually Grounded Interaction and Language Workshop*, 2017.

A Appendix for Chapter 2

A.1 Complementary Theoretical Intuition for SIL and Its Limitation

We here provide a complementary intuition of Seeded Iterated Learning by referring to some mathematical tools used to study Iterated Learning dynamics in the general case. These are not rigorous proof but guide the design of SIL.

One concern is that, since natural language is not fully compositional, whether iterated learning may favor the emergence of a new compositional language on top of the initial one. In this spirit, (Griffiths and Kalish, 2005; Kalish et al., 2007) modeled iterated learning as a Markov Process and showed that vanilla iterated learning indeed converges to a language distribution that (i) is independent of the initial language distribution, (ii) depends on the student language before the inductive learning step.

Fortunately, (Chazelle and Wang, 2017) show iterated learning can converge towards a distribution close to the initial one with high probability if the intermediate student distributions remain close enough of their teacher distributions and if the number of training observations increases logarithmically with the number of iterations.

This theoretical result motivates one difference between our framework and classical iterated learning. As we want to preserve the pretrained language distribution, we do not initialize the new students from scratch as in (Li and Bowling, 2019; Guo et al., 2019; Ren et al., 2020) because the latter approach exerts a uniform prior on the space of language. At the same time, we would like to add a prior that favors natural language (e.g., favoring language whose token frequency satisfies Zipf's Law).

A straightforward instantiation of the above theoretical results is initializing new students as the pretrained model. However, we empirically observe that periodically re-setting the model to initial pretrained model would quickly saturate the task score. As a result, we just keep using the students from the last imitation learning for the beginning of the new generation and retain the natural language properties from the pretraining

checkpoint.

However, we would also point out the limitation of existing theoretical results in the context of deep learning: The theoretical iterated learning results assume the agent to be a perfect Bayesian learner (e.g., learning is inferring the posterior distribution of hypothesis given data). However, we only apply standard deep learning training procedures in our setup, which might not have this property. Because of the assumption of perfect Bayesian learner, (Chazelle and Wang, 2019) suggests using training sessions with increasing length. However, in practice, increasing k_2 may be counter-productive because of overfitting issues (especially when we have a limited number of training scenarios).

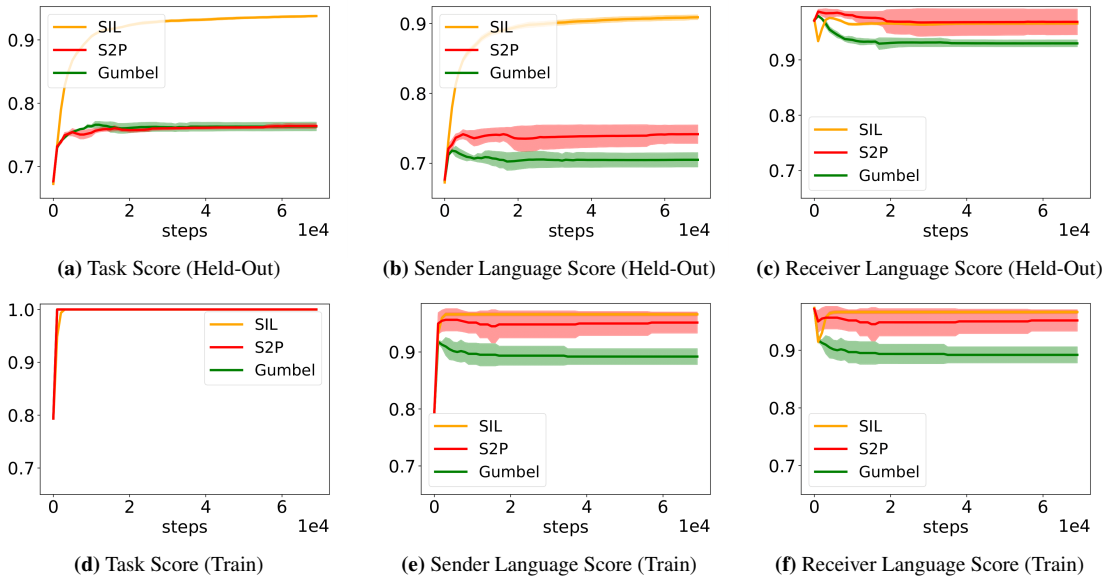
A.2 LEWIS GAME

A.2.1 Additional Plots

We sweep over different Gumbel temperatures to assess the impact of exploration on language drift. We show the results with Gumbel temperature $\tau = 1, 10$ in Fig A.2 and Fig A.1. We observe that the baselines are very sensitive to Gumbel temperature: high temperature both decreases the language and tasks score. On the other side, Seeded Iterated Learning perform equally well on both temperatures and manage to maintain both task and language accuracies even with high temperature.

A.2.2 Tracking Language Drift with Token Accuracy

To further visualize the language drift in the Lewis game, we focus on the evolution of the probability of speaking different words when facing the same concept. Formally, we track the change of conditional probability $s(w|c)$. The result is in Figure A.3.



(a) Task Score (Held-Out) **(b)** Sender Language Score (Held-Out) **(c)** Receiver Language Score (Held-Out)

(d) Task Score (Train) **(e)** Sender Language Score (Train) **(f)** Receiver Language Score (Train)

FIGURE A.1 – Complete training curves for Task score and sender grounding in Lewis Game comparing SIL vs baselines for $\tau = 10$ on the held-out dataset (bottom), and the interactive training split (bottom). We observe that the three methods reach 100% accuracy on the training task score, but their score differs on the held-out split. For SIL we use $k_1 = 1000, k_2 = k_2' = 400$.

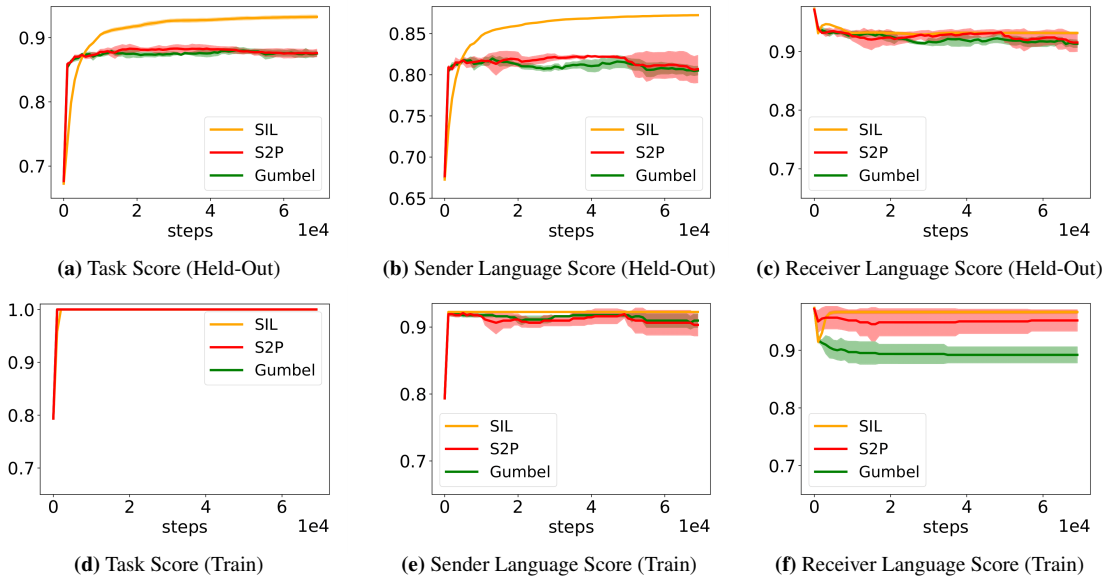
A.3 TRANSLATION GAME

A.3.1 Data Preprocessing

We use Moses to tokenize the text (Koehn et al., 2007b), and we learn byte-pair-encoding (Sennrich et al., 2016b) from Multi30K (Elliott et al., 2016) with all languages. Then we apply the same BPE to different datasets. Our vocab size for En, Fr, De is 11552, 13331, and 12124.

A.3.2 Model Details and Hyperparameters

The model is a standard seq2seq translation model with attention (Bahdanau et al., 2015). Both encoder and decoder have a single-layer GRU (Cho et al., 2014) with hidden size 256. The embedding size is 256. There is a dropout after embedding layers for both encoder and decoder For the decoder at each step, we concatenate the input and the attention context from the last step.



(a) Task Score (Held-Out) **(b)** Sender Language Score (Held-Out) **(c)** Receiver Language Score (Held-Out)

(d) Task Score (Train) **(e)** Sender Language Score (Train) **(f)** Receiver Language Score (Train)

FIGURE A.2 – Complete training curves for Task score and sender grounding in Lewis Game comparing SIL vs baselines for $\tau = 1$ on the held-out dataset (bottom), and the interactive training split (bottom). For SIL we use $k_1 = 1000, k_2 = k_2' = 400$.

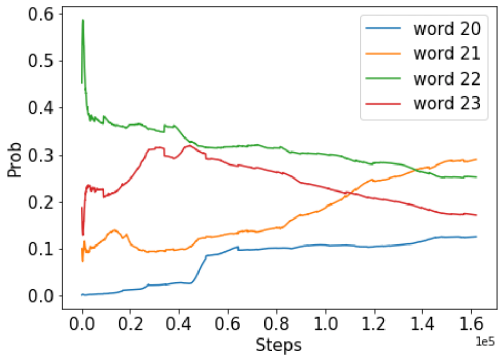


FIGURE A.3 – Change of conditional probability $s(w|c)$ where $c = 22$ and $w = 20, 21, 22, 23$. Following pretraining, $s(22|22)$ start with the highest probability. However, language drift gradually happens and eventually word 21 replaces the correct word 22.

Pretraining For Fr-En agent, we use dropout ratio 0.2, batch size 2000, and learning rate 3e-4. We employ a linear learning rate schedule with the annealing steps of 500k. The minimum learning rate is 1e-5. We use Adam optimizer (Kingma and Ba, 2014) with $\beta = (0.9, 0.98)$. We employ a gradient clipping of 0.1. For En-De, the dropout ratio is 0.3. We obtain a BLEU score of 32.17 for Fr-En, and 20.2 for En-De on the IWSLT test dataset (Cettolo et al., 2012).

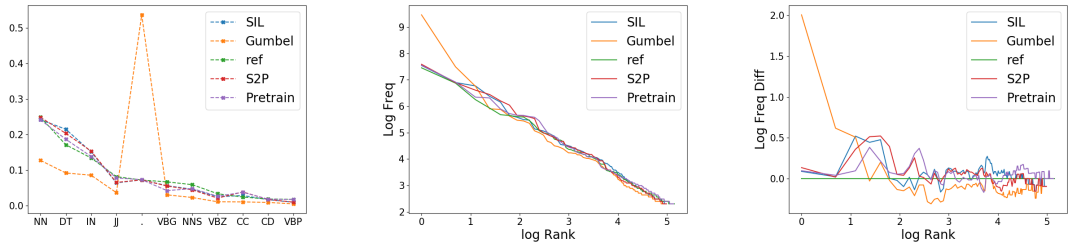
TABLE A.1 – Translation Game Results. The checkmark in "ref len" means the method use reference length to constrain the output during training/testing. ↑ means higher the better and vice versa. Our results are averaged over five seeds, and reported values are extracted for the best BLEU(De) score during training. We here use a Gumbel temperature of 0.5.

Method		ref len	BLEU↑		NLL↓	R1%↑
			De	En		
(Lee et al., 2019)	Pretrained	N/A	16.3	27.18	N/A	N/A
	PG	✓	24.51	12.38	N/A	N/A
	PG+LM+G	✓	28.08	24.75	N/A	N/A
Ours	Pretrained	N/A	15.68	29.39	2.49	21.9
	Fix Sender	N/A	22.02 ± 0.18	29.39	2.49	21.9
	Gumbel		27.11 ± 0.14	14.5 ± 0.83	5.33 ± 0.39	9.7 ± 1.2
	Gumbel	✓	26.94 ± 0.20	23.41 ± 0.50	5.04 ± 0.01	18.9 ± 0.8
	S2P($\alpha = 0.1$)		27.43 ± 0.36	19.16 ± 0.63	4.05 ± 0.16	13.6 ± 0.7
	S2P($\alpha = 1$)		27.35 ± 0.19	29.73 ± 0.15	2.59 ± 0.02	$\mathbf{23.7 \pm 0.7}$
	S2P($\alpha = 5$)		24.64 ± 0.16	$\mathbf{30.84 \pm 0.07}$	2.51 ± 0.02	23.5 ± 0.5
	NIL		$\mathbf{28.29 \pm 0.16}$	29.4 ± 0.25	$\mathbf{2.15 \pm 0.12}$	21.7 ± 0.2

Finetuning During finetuning, we use batch size 1024 and learning rate 1e-5 with no schedule. The maximum decoding length is 40, and the minimum decoding length is 3. For iterated learning, we use $k_1 = 4000$, $k_2 = 200$ and $k_2' = 300$. We set Gumbel temperature to be 5. We use a greedy sample from the teacher speaker for imitation.

A.3.3 Language Model and Image Ranker Details

Our language model is a single-layer LSTM (Hochreiter and Schmidhuber, 1997) with hidden size 512 and embedding size 512. We use Adam and a learning rate of 3e-4. We use a batch size of 256, and a linear schedule with 30k anneal steps. The language model is trained with captions from MSCOCO (Lin et al., 2014). For the image ranker, we use the pretrained ResNet-152 (He et al., 2016) to extract the image features. We use a GRU (Cho et al., 2014) with hidden size 1024 and embedding size 300. We use a batch size of 256 and use VSE loss (Faghri et al., 2017). We use Adam with a learning rate of 3e-4 and a schedule with 3000 anneal steps (Kingma and Ba, 2014).



(a) POS tag distribution. **(b)** Word Frequency Analysis **(c)** Difference of Log of Word Frequency

FIGURE A.4 – Language statistics on samples from different method.

A.3.4 Language Statistics

We here compute several linguistic statistics on the generated samples to assess language quality.

POS Tag Distribution We compute the Part-of-Speech Tag (POS Tag (Marcus et al., 1993)) distribution by counting the frequency of POS tags and normalizing it. The POS tag is sorted according to their frequencies in the reference, and we pick the 11 most common POS tags for visualization, which are:

— NN Noun, singular or mass

— DT Determiner

— IN Preposition or subordinating conjunction

— JJ Adjective

— VBG Verb, gerund or present participle

— NNS Noun, plural

— VBZ Verb, 3rd person singular present

— CC Coordinating conjunction

— CD Cardinal number

The results are shown in Figure A.4a. The peak on “period" shows that Gumbel has a tendency of repeating periods at the end of sentences. However, we observe that both S2P and

Word Frequency For each generated text, we sort the frequency of the words and plot the log of frequency vs. log of rank. We set a minimum frequency of 50 to exclude long-tail results. The result is in Figure A.4b.

Word Frequency Difference To further visualize the difference between generated samples and reference, we plot the difference between their log of word frequencies in Figure A.4c.

A.4 HUMAN EVALUATION

We here assess whether our language drift evaluation correlates with human judgment. To do so, we performed a human evaluation with two pairwise comparison tasks.

— In Task1, the participant picks the best English semantic translation while observing the French sentence.

— In Task2, the participant picks the best English translation from two candidates.

Thus, the participants are likely to rank captions mainly by their syntax/grammar quality in Task2. In contrast, they would also consider semantics in Task1, partially disentangle structural and semantic drift.

We use the validation data from Multi30K (1013 French captions) for each task and generate 4 English sentences for each French caption from the Pretrain, Gumbel, S2P, and SIL. We also retrieved the ground-truth human English caption. We then build the test by randomly sampling two out of five English captions. We gathered 22 people, and we collected about 638 pairwise comparisons for Task2 and 315 pairwise comparisons for Task1. We present the result in Table A.3 and Table A.4. I also include the binomial statistical test result where the null hypothesis is *methods are the same*, and the alternative hypothesis is *one method is better than the other one*.

Unsurprisingly, we observe that the Human samples are always preferred over generated sentences. Similarly, Gumbel is substantially less preferred than other models in both settings.

In Task 1(French provided), human users always preferred S2P and SIL over pretrained models with a higher win ratio. Oh the other hand, when French is not provided, the human users prefer the pretrain models over S2P and SIL. We argue that while the pretrained model keeps generating grammatically correct sentences, its translation effectiveness is worse than both S2P and SIL since these two models go through interactive learning to adapt to a new domain.

Finally, SIL seems to be preferred over S2P by a small margin in both tasks. However, our current ranking is not conclusive. We can see that the significance level of comparisons among Pretrain, S2P, and SIL is not smaller enough to reject the null hypothesis, especially in task 1, where we have fewer data points. In the future, we plan to have a more extensive scale human evaluation to differentiate these methods further.

TABLE A.2 – The Win-Ratio Results. The number in row X and column Y is the empiric ratio that method X beats method Y according to collected human pairwise preferences. We perform a naive ranking by the row-sum of win-ratios of each method. We also provide the corresponding P-values under each table. The null hypothesis is *two methods are the same*, while the alternative hypothesis is *two methods are different.*

TABLE A.3 – With French Sentences

	Gumbel	Pretrain	S2P	SIL	Human
Gumbel	0	0.25	0.15	0.12	0
Pretrain	0.75	0	0.4	0.4	0.13
S2P	0.84	0.6	0	0.38	0.21
SIL	0.88	0.6	0.63	0	0.22
Human	1	0.87	0.79	0.77	0
Ranking	Human(3.4), SIL(2.3), S2P(2.0), Pretrain(1.7), Gumbel(0.5)				
	P-values				
	Gumbel	Pretrain	S2P	SIL	Human
Gumbel	-	$< 10^{-2}$	$< 10^{-2}$	$< 10^{-2}$	$< 10^{-2}$
Pretrain	$< 10^{-2}$	-	0.18	0.21	$< 10^{-2}$
S2P	$< 10^{-2}$	0.18	-	0.15	$< 10^{-2}$
SIL	$< 10^{-2}$	0.21	0.15	-	$< 10^{-2}$
Human	$< 10^{-2}$	$< 10^{-2}$	$< 10^{-2}$	$< 10^{-2}$	-

TABLE A.4 – Without French Sentences

	Gumbel	Pretrain	S2P	SIL	Human
Gumbel	0	0.16	0.12	0.13	0.02
Pretrain	0.84	0	0.69	0.59	0.15
S2P	0.88	0.31	0	0.38	0.05
SIL	0.86	0.41	0.62	0	0.01
Human	0.98	0.85	0.95	0.98	0
Ranking	Human(3.8), Pretrain(2.3), SIL(1.9), S2P(1.6), Gumbel(0.4)				
	P-values				
	Gumbel	Pretrain	S2P	SIL	Human
Gumbel	-	$< 10^{-2}$	$< 10^{-2}$	$< 10^{-2}$	$< 10^{-2}$
Pretrain	$< 10^{-2}$	-	$< 10^{-2}$	0.08	$< 10^{-2}$
S2P	$< 10^{-2}$	$< 10^{-2}$	-	0.06	$< 10^{-2}$
SIL	$< 10^{-2}$	0.08	0.06	-	$< 10^{-2}$
Human	$< 10^{-2}$	$< 10^{-2}$	$< 10^{-2}$	$< 10^{-2}$	-

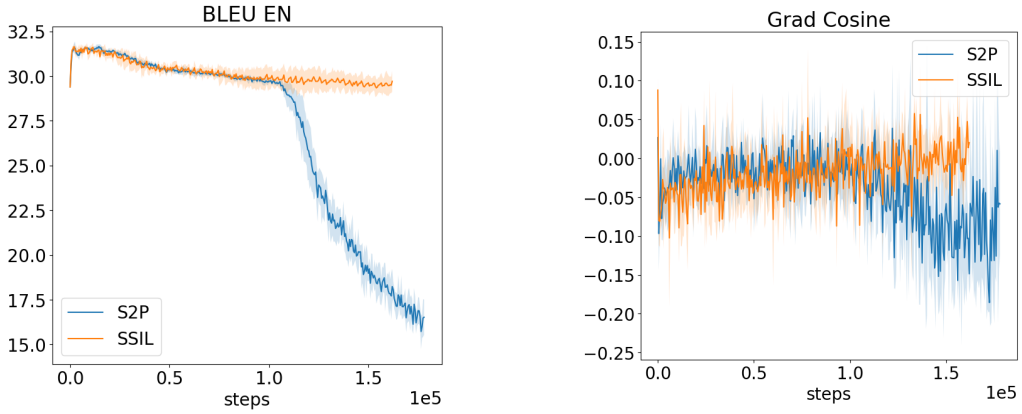
B Appendix for Chapter 3

B.1 HYPER-PARAMETERS

During finetuning, we set Gumbel temperature to be 0.5 and follow the previous work Lu et al. (2020b) for other hyperparameters, e.g. learning rate, batch size, etc. We list our hyper-parameters and our sweep: We mainly use P100 GPU for our experiments. For training 200k steps, Gumbel takes 17 hours, S2P takes 24 hours, SIL takes 18 hours and SSIL takes 24 hours. The best hyperparameters for SIL are $k_1 = 3000, k_2 = 200, k_2' = 300$. The best $alpha$ for $S2P$ is 1, while for $SSIL$ we choose $\alpha = 0.5$.

Name	Sweep
k_1	3000, 4000
k_2	200, 300, 400
k_2'	200, 300, 400
α	0, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7

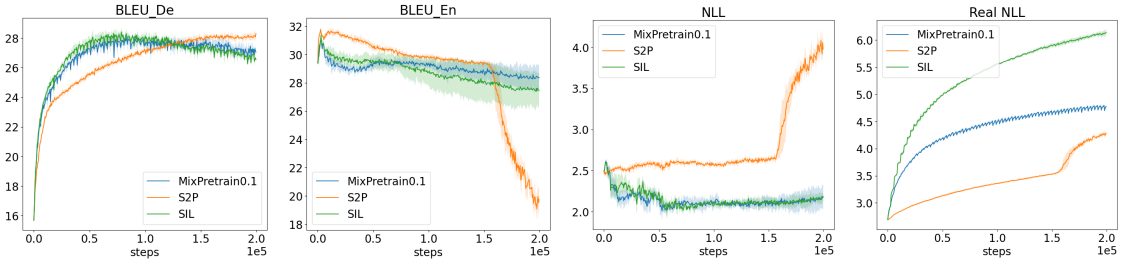
B.2 ADDITIONAL FIGURES



(a) Bleu En

(b) Cosine Similarity

FIGURE B.1 – Cosine similarity bewteen $\mathcal{L}^{\text{CE}}_{pretrain}$ and $\mathcal{L}^{\text{INT}}$ when $\alpha = 0.7$



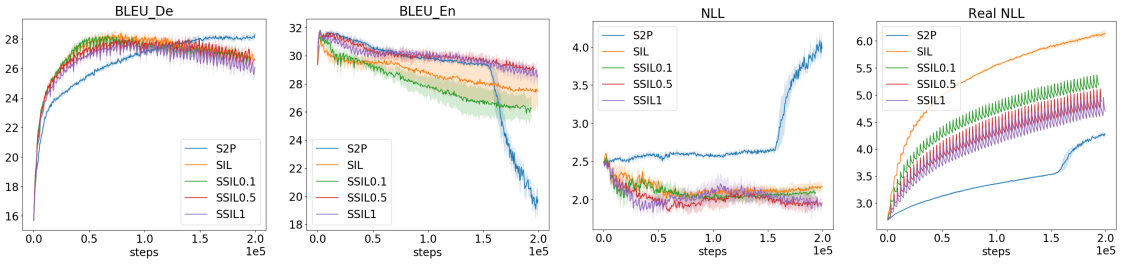
(a) BLEU De (Task Score)

(b) BLEU En

(c) NLL

(d) RealNLL

FIGURE B.2 – Mix with Pretraining data in SIL.



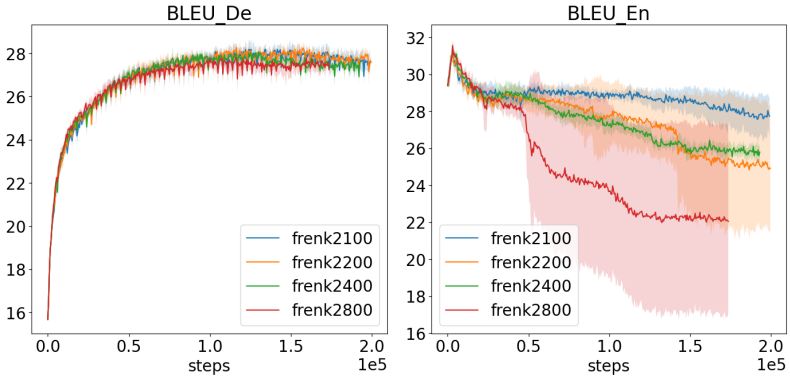
(a) BLEU De (Task Score)

(b) BLEU En

(c) NLL

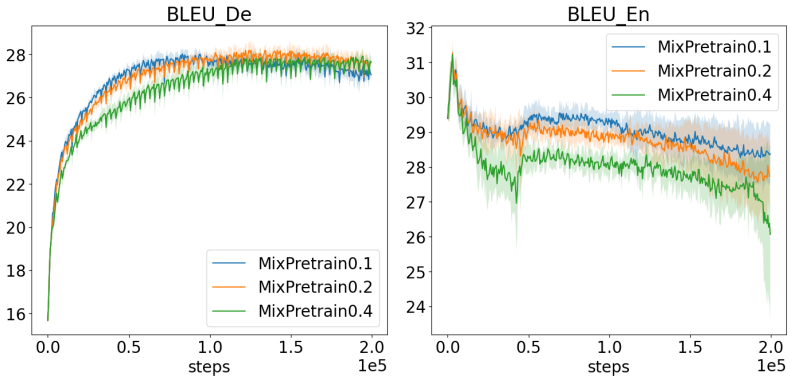
(d) RealNLL

FIGURE B.3 – SSIL with different α



(a) BLEU De (Task Score) **(b)** BLEU En

FIGURE B.4 – Effect of k_2 for MixData.$\alpha = 0.2$



(a) BLEU De (Task Score) **(b)** BLEU En

FIGURE B.5 – Effect of α for MixData. $k_2 = 100$