REPRODUCING LANGUAGE DRIFT FOR POLICY GRADIENT FINE-TUNING IN A COLLABORATIVE TRANSLATION GAME

Yuchen Lu & Aaron Courville
Department of Computer Science and Operations Research
Universit de Montral
Montral, QC H3C3J7, Canada

ABSTRACT

In this paper, we focus on reproducing the policy gradient benchmark of the paper: *Countering Language Drift via Grounding* (Lee et al., 2019). We give an overview of the paper and the collaborative translation game proposed in the paper. We clarify some critical details including length normalization in computing reward and parameter sharing in architecture, and their effects on the reproducibility. We perform a hyper-parameter search and obtain a stronger policy gradient benchmark than the paper. However, our model still suffers from language drift with a drop in a fluency score while achieving a high communication score. We also perform the token frequency analysis and find that the token frequency induced by our model deviates from the natural language token frequency, but not necessarily toward a flatter one as claimed in Lee et al. (2019). Our experiments verify authors' claim on the inability of the policy gradient fine-tuning to counter language drift. We release our implementation on github[1] for interested readers.

1 INTRODUCTION

Recently, researchers have become interested in exploring language learning in complex, interactive environments, including goal oriented dialog (Wei et al., 2018), visual dialog (Strub et al., 2017) and emergent communications (Cao et al., 2018). Due to the difficulty of collecting labeled datasets in these settings, many works choose the following training paradigm: (1) do supervised learning on collected dataset; then (2) fine-tune with self-play to maximize expected reward. By following this training paradigm, many works have encountered the phenomenon of *language drift* (Wei et al., 2018; Lewis et al., 2017; Zhu et al., 2017). During self-play, agents adapt to each other and their shared language drifts to a point where it can become incomprehensible to human, all while steadily improving the expected reward.

Lee et al. (2019) investigate a particular solution to language drift and they propose a collaborative translation game as a canonical example for studying language drift. As is illustrated in Figure 1, there are two machine translation agents which collaborate to translate from French (Fr) to German (De), with English (En) as the pivot (or common) language between them. Agents are separately pre-trained on either a Fr→En corpus or a En→De corpus, and then are jointly fine-tuned on separate Fr→De corpus. In this setup, completing the task would require collaboration between Fr→En Agent, or Agent A, and En→De Agent, or Agent B. This setup can also use the Fr→En BLEU score as a measure of *fluency* and the Fr→En→De BLEU score as a measure of *communication*. During fine-tuning with policy gradient, the authors observe that the fluency score deteriorates while the communication score increases, suggesting language drift. To counter language drift, the authors use auxiliary rewards with a language modeling task and a grounding task. They find that merely adding a language modeling task only impose syntactic conformity, but adding a grounding task is necessary for the best task performance, since it conveys semantic information. Beyond BLEU score, the authors also propose the token frequency analysis and word recall rate by POS tag, and

[1] https://github.com/JACKHAHA363/language_drift

they show examples where the generated English is repetitive and where certain words are replaced by other words that – from the perspective of standard english – are used inappropriately.

In this paper, we mainly focus on reproducing Lee et al.'s (2019) language drift in the policy gradient methods. This is crucial because policy gradient methods are widely used in NLP for fine-tuning because of their scalability to large vocabulary size. Policy gradient methods are known to be sensitive to hyper-parameters, random seed and implementation details. It remains a question whether a highly tuned policy gradient method could avoid language drift. We are also aware of the recent development in more advanced policy gradient algorithms (Schulman et al., 2015; 2017; Wu et al., 2017), but we will only focus on a more established policy gradient method, using a learned value function as a baseline.

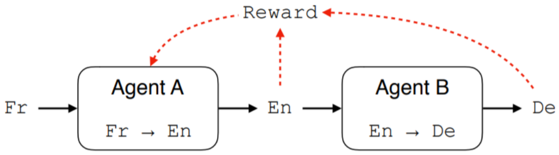


Figure 1: Translation game (Lee et al., 2019). The whole system translate French to German, with English as an immediate language. Agent A translates from French (Fr) to English (De), while Agent B translates from English (En) to German (De).

2 Preliminary

The dataset consists of N triples of $\{Fr_i, En_i, De_i\}_{i=1}^{N}$, where Fr_i^j represents the jth token of ith French sentence. For each triple, we sample from $P_A(En|Fr_i)$ to get the English translation $\widetilde{En}_i$. The loss for training Agent B is

$$L_B(i) = -\log_B(De_i|\widetilde{En}_i) \tag{1}$$

For training Agent A, we formulate the sampling of English as a sequential decision process. At each time step t, we have $s^t = (Fr_i, \widetilde{En}_i^0, ..., \widetilde{En}_i^{t-1})$ where $\widetilde{En}_i^0 =< BOS >$. The action space is the vocabulary, and the trajectory ends when we encounter $< EOS >$ or when it exceeds the length of the French input (Lee et al., 2019). The agent then gets a terminal reward $R_i = -L_B(i)$ for the trajectory $\widetilde{En}_i$. For agent A, the total training loss is

$$L_A(i) = -\sum_t (R_i - V(s^t)) \log P_A(\widetilde{En}_i^t|s^t) + \sum_t \alpha_v (R_i - V(s^t))^2 - \alpha_{ent} \sum_t H(\log P_A(\widetilde{En}_i^t|s^t))$$

where H is the entropy term to encourage exploration, and α_v, α_{ent} are two hyper-parameters. This loss is also normalized by the lengths of the translated English sentences. During evaluation, both agents use greedy decoding to get the BLEU scores. Although Lee et al. (2019) have a weight α_{pg} on the policy gradient term, we omit it because it can be absorbed in the learning rate.

3 Implementation Details

For pre-processing the dataset, we use torchtext[2] and its moses tokenizer to process the raw and unstructured data into textfile. We then use the original Byte Pair Encoding (BPE) implementation[3] to learn and apply BPE and extract the vocabularies. We start from OpenMNT[4] and use Pytorch 1.0 as the main codebase because it provides good framework for machine translation. We use NLTK[5] to compute the BLEU scores.

[2] https://github.com/pytorch/text/tree/master/torchtext

[3] https://github.com/rsennrich/subword-nmt

[4] https://github.com/OpenNMT/OpenNMT-py

[5] https://www.nltk.org/_modules/nltk/translate/bleu_score.html

	IWSTL	Multi30K
Fr→En	33.17 / 34.05	24.75 / 26.80
En→De	19.11 / 21.94	17.67 / 18. 56

Table 1: Pretraining Results. The numbers here are the BLEU scores, and the ones after the slash are from Lee et al. (2019).

We use a sequence-to-sequence model with attention (Bahdanau et al., 2014) for both agents, and both encoders and decoders are 1-layer GRU with 256 hidden size and 256 embedding size. Regarding the value function approximation, we use a separate encoder-decoder GRU with a single linear layer.

In all experiments, we use the Adam optimizer with $\beta_1 = 0.9, \beta_2 = 0.98$ and default ϵ, and use a piece-wise constant learning schedule. More details can be found in our github repository.

4 Experiment Results

4.1 Pretraining

For pre-training, we use Adam with learning rate 0.001 and batch size 64. We first train 30000 steps and then decay the learning late by 0.1 for every 10000 steps. In total, we pre-train the models for 100000 steps, and pick the best performing checkpoints on validation set. With this procedure, we achieve comparable but slightly worse BLEU scores as shown in Table 1.

4.2 Policy Gradients Fine-tuning

4.2.1 Hyper-parameter Search

We use Adam for both agents and the value network, but we use different learning rates and schedules for the two agents. For agent B, we use a learning rate of 0.001 and decay by 0.5 for every 1000 steps. For Agent A, we tune its learning rate and fix it thorough out training. We also use a fixed learning rate 0.001 for the value network. When tuning the hyper-parameters, we keep $\alpha_v = 0.5$ and only change learning rate lr and α_{ent}. We use a larger batch size 256 than pretraining. We run each configuration for 5 times and read the results at the end of training. We also run the experiment that keeps Fr→En fixed throughout training which is also done by Lee et al. (2019).

4.2.2 Results

The results are presented in Table 2. We find that both ”Pretrained” and ”Fr→En Fixed” yield comparable performance but with a minor performance drop. We find that the policy gradient results are very sensitive to the choice of learning rate. In the worst case, the policy is never able to learn anything useful, while in the best case ($lr = 0.0001, \alpha_{ent} = 0.01$), the agent can achieve a communication score of 29.82 with only a drop of 4.5 in the fluency score. Although this policy gradient benchmark is much stronger than Lee et al. (2019), the full model PG+LM+G, using language modelling task and grounding task as extra training tasks, still outperforms the policy gradient benchmark in terms of a stronger fluency score. In particular, we find that when we set $\alpha_{ent} = 0.01, lr = 0.0004$, our method is able to achieve a similar performance to the policy gradient benchmark in the paper. These results show that the language drift of the policy gradient fine-tuning is reproducible, and simply tuning hyper-parameters could not counter the language drift. Our hyper-parameter search demonstrates the reliability of authors' observation on the policy gradient fine-tuning. We also plot the curve for performance on validation set during training for different hyper-parameter configurations in the Appendix.

We follow the original paper and perform token frequency analysis. We sort the frequency of words decreasingly of each model and then compare with the sorted word frequency in the reference English. The comparison results can be found in Figure 2. Although Lee et al. (2019) claims to observe a flatter distribution for policy gradient methods, we do not observe this and the our token frequency

Methods		Configuration	Fr→En	Fr→En→De
Lee et al. (2019)	Pretrained		27.18	16.30
	Fr→En Fixed		27.18	20.96
	PG	PG	12.38 (0.67)	24.51 (1.48)
		PG+LM(All)	23.60(1.05)	27.67(0.39)
		PG+LM+G(All)	24.75(0.40)	28.08(0.73)
Ours	Pretrained		24.88	11.98
	Fr→En Fixed		24.88	20.96
	PG	$\alpha_{ent} = 0.01, lr = 0.0001$	20.41(0.75)	29.82(0.26)
		$\alpha_{ent} = 0.01, lr = 0.0002$	15.30(1.02)	28.60(0.55)
		$\alpha_{ent} = 0.01, lr = 0.0004$	11.41(0.49)	25.78(0.23)
		$\alpha_{ent} = 0.01, lr = 0.0008$	6.13(0.45)	20.74(0.74)
		$\alpha_{ent} = 0.02, lr = 0.0001$	16.04 (1.13)	28.31 (0.11)
		$\alpha_{ent} = 0.04, lr = 0.0001$	2.44(0.53)	14.54(0.74)
		$\alpha_{ent} = 0.08, lr = 0.0001$	0.00(0.00)	2.30(0.09)

Table 2: BLEU Scores Comparison. The Fr→En BLEU score stands for the fluency score, and lower score indicates more drift. The Fr→En→De BLEU indicates the communication score, the higher the better. For the policy gradient method, the number in the parenthesis is the standard deviation.

results are more skewed, which needs further investigation. However, the proposed token frequency analysis proposed can be an informative metric for language drift from this observation.

5 Reproducibilty Discussion

In general, the authors have provided enough details and the policy gradient method in this paper is straightforward to reproduce, but we would like to highlight some details and pitfalls during our attempt.

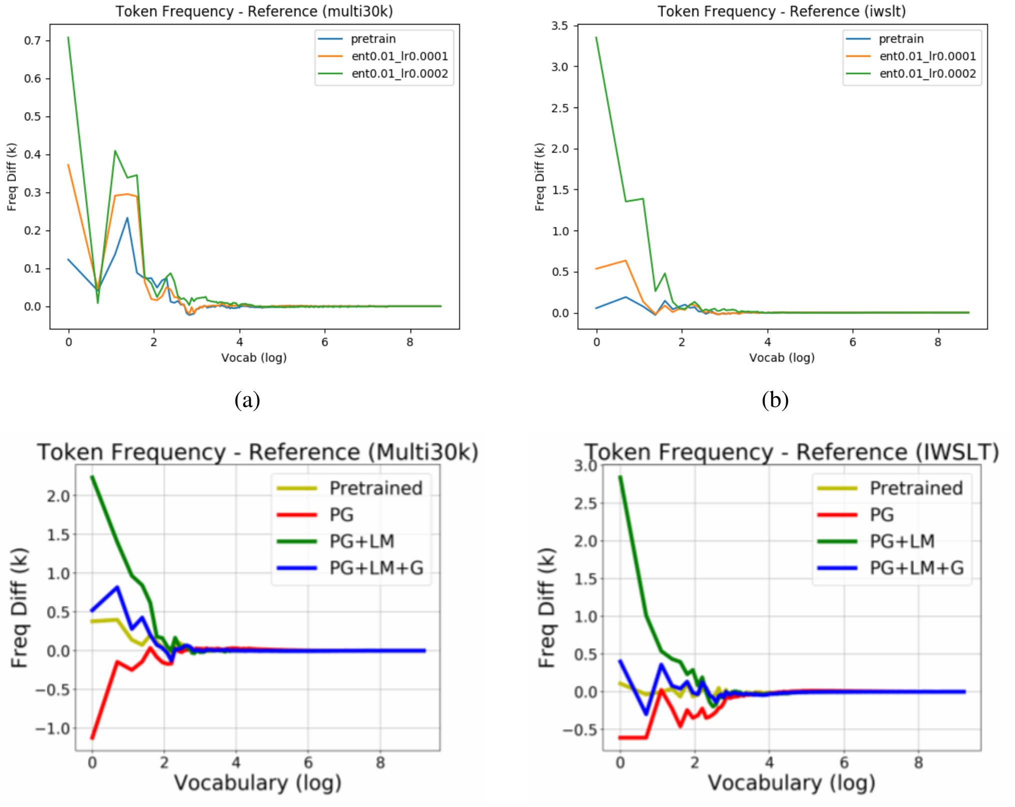
Weight Sharing While Lee et al. (2019) use a value head with 2 layers MLP, and share the encoder-decoder parameters with the policy network, We find that sharing the GRU parameters makes the training less stable – an observation that warrants further investigation.

Learning Rate Schedule Although Lee et al. (2019) have mentioned on their learning rate schedule in both pre-training and fine-tuning, they do not articulate it. As a result, we choose to use the default piece-wise constant schedule from OpenNMT, which could potentially explain the gap shown in 1. In addition we find the use of learning rate annealing would cause sub-optimal results in fine-tuning. Therefore we use a fixed learning rate for Agent A and an annealing schedule for Agent B for convergence, while Lee et al. (2019) anneal the learning rates for both agents.

Reward with Length Normalization During implementation, we noticed that in the original Eqn.1, the reward favors shorter German sentences since the reward is composed of many log-likelihood terms and each log-likelihood term is negative. After discussing with the authors on OpenReview[6], we find that the reward needs to be normalized by the length of the German sentence, which is not explicitly mentioned in the paper. We argue that this is an important detail in our experiment, because otherwise the fine-tuning is unstable.

Other Hyper-parameters Although the authors provide most of the important hyper-parameter, they miss some important ones like batch size, β of Adam etc. However, for those hyper-parameters not mentioned in paper, trying several conventional values is enough for getting a stable training.

[6] https://openreview.net/forum?id=BkMn9jAcYQ



(a)

(b)

(c)

Figure 2: Token Frequency Analysis Comparison. The first row is our reproduction, while the second row is Lee et al.'s (2019) results. Our results is comparable to the red lines in the second row, denoted as ”PG”.

6 Conclusion

In this paper we reproduce the language drift observed when using policy gradient methods in a collaborative translation game. We summarize the original paper and provide a detailed description on our implementation. By performing a hyper-parameter search on learning rate and entropy coefficient, we obtain a stronger benchmark but the policy gradient method still suffer from a drop of ”fluency” score. Our token frequency analysis also shows that language drift induces qualitatively different token frequencies, but it does not necessarily become flatter as claimed in the paper.

We describe several potential pitfalls including weight sharing, learning rate schedule and hyper-parameter choice. In particular, we find that length normalization for reward computation is crucial for stable training, which is not mentioned in the paper. We point out these details hoping that it could be useful for future attempt in reproducing this paper.

We conclude that the authors' claim on the inability of policy gradient methods to counter language drift is reproducible and reliable.

Acknowledgments

We want thank authors' response and the insightful comments of reviewers on OpenReview. We thank Nvidia and Compute Canada for providing the computing resource for this work.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Kris Cao, Angeliki Lazaridou, Marc Lanctot, Joel Z Leibo, Karl Tuyls, and Stephen Clark. Emergent communication through negotiation. *arXiv preprint arXiv:1804.03980*, 2018.

Jason Lee, Kyunghyun Cho, and Douwe Kiela. Countering language drift via grounding, 2019. URL `https://openreview.net/forum?id=BkMn9jAcYQ`.

Mike Lewis, Denis Yarats, Yann N Dauphin, Devi Parikh, and Dhruv Batra. Deal or no deal? end-to-end learning for negotiation dialogues. *arXiv preprint arXiv:1706.05125*, 2017.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

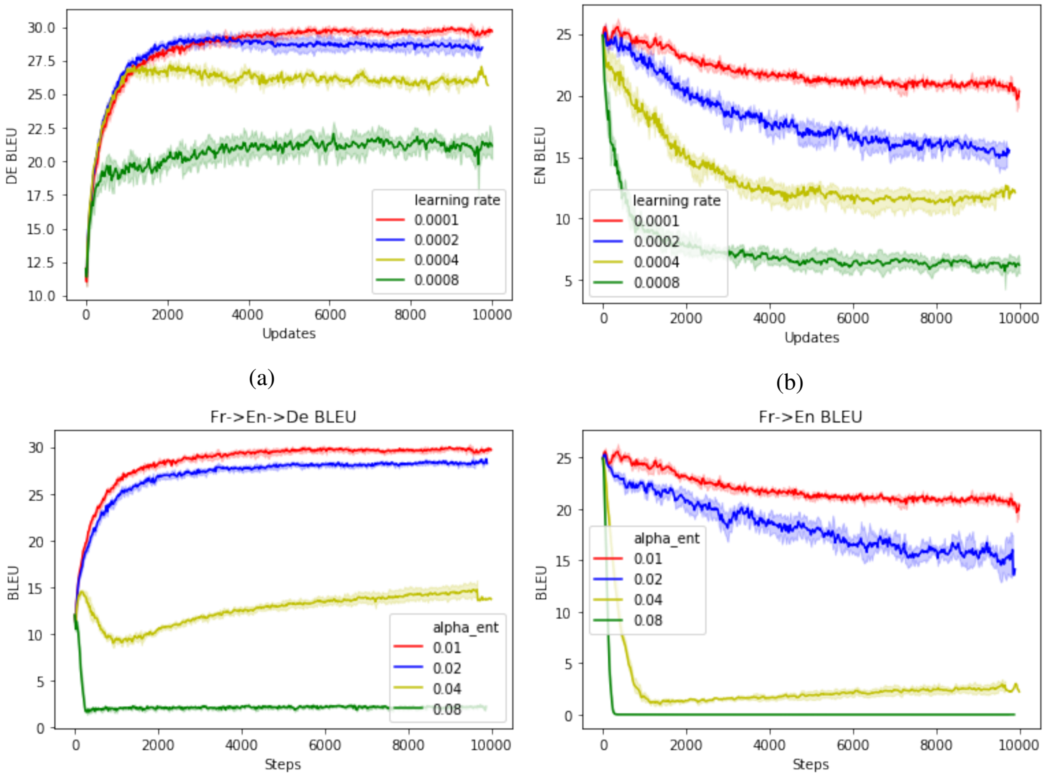
Florian Strub, Harm De Vries, Jeremie Mary, Bilal Piot, Aaron Courville, and Olivier Pietquin. End-to-end optimization of goal-driven and visually grounded dialogue systems. *arXiv preprint arXiv:1703.05423*, 2017.

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*, pp. 3844–3854, 2018.

Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In *Advances in neural information processing systems*, pp. 5279–5288, 2017.

Yan Zhu, Shaoting Zhang, and Dimitris Metaxas. Interactive reinforcement learning for object grounding via self-talking. *arXiv preprint arXiv:1712.00576*, 2017.

7 APPENDIX



(a)

(b)

(c)

(d)

Figure 3: Communication score and fluency score on validation set during fine-tuning for our hyper-parameter search.